Universität Ulm | 89069 Ulm | Deutschland
Fakultät für Ingenieurwissenschaften, Informatik und Psychologie
Institut für Neuroinformatik
Direktor: Prof. Dr. Günther Palm

Learning in Layered Multimodal Classifier Architectures for Cognitive Technical Systems

Dissertation zur Erlangung des Doktorgrades
Doktor der Naturwissenschaften (Dr. rer. nat.)
der Fakultät für Ingenieurwissenschaften, Informatik und Psychologie
der Universität Ulm

Vorgelegt von:
Michael Glodek
aus Lübeck, Deutschland
2015

Amtierende Dekanin:
Prof. Dr. Tina Seufert

Gutachter:
Prof. Dr. Günther Palm
Prof. Dr. Thomas Martinetz
Prof. Dr. Barbara Hammer

Tag der Promotion:
24.03.2016

Abstract

Modern computer systems have changed our way of living fundamentally. They improve our effectiveness by assisting us in our work and daily tasks. However, current systems are limited to a direct input of commands. Furthermore, they are unable to take active decisions on the behalf of the user, mostly because of a lack of information about the user. Cognitive technical systems (CTS) pick up on these deficiencies by recognizing user states and the user's environment with the help of sensor data. The derived information is collected in a knowledge base and further processed by the application and the dialog management to perform the decision making.

In this thesis, new methods addressing sensor-based state recognition in the context of CTS in human-computer interaction are developed and empirically evaluated. The focus is set on large multimodal and temporal multiple classifier systems. Furthermore, the work covers the topics sequential classifiers, handling of partially-available information, and integration of sub-symbolic and symbolic information for complex state recognition. Following approaches are presented in this work: ensemble Gaussian mixture model (EGMM), conditioned hidden Markov model (CHMM), fuzzy conditioned hidden Markov model (FCHMM), hidden Markov model using graph probability densities (HMM-GPD), Markov fusion network (MFN), Kalman filter for classifier fusion and layered classifier architectures. The EGMM extends the classical GMM by the ensemble technique in order to achieve a more robust density estimation. The CHMM and the FCHMM extend the HMM by an additional causal sequence which influences the hidden states. The CHMM uses a sequence of discrete causes, whereas the FCHMM uses a sequence of causes with fuzzy memberships. Both approaches can further be utilized to the integrate symbolic information. The HMM-GPD introduces graph probability densities as observations in HMM. MFN and Kalman filter for classifier fusion are probabilistic algorithms for temporal and multimodal late fusion which are robust against sensor failures. Within this thesis, the unidirectional layered architecture (ULA) and the bidirectional layered architecture (BLA) are proposed. Both architectures recognize complex classes based on probabilistic logical rules and the temporal combination of basic patterns. Each layer recognizes patterns based on the class predictions of the underlying layer. Hence, upper layers recognize more complex patterns. The BLA additionally propagates information in the direction of the lower layers. The empirical evaluation of the proposed methods is performed on datasets for affective state and activity recognition, e.g. the Freetalk dataset, AVEC 2011, AVEC 2012, AVEC 2013 and UUlmMAD.

The EGMM proved to be more robust and accurate when compared to the conventional GMM approaches. It was shown that the selection of suitable parameters is considerably easier. Further evaluations showed that the multimodal late fusion using the CHMM outperformed the HMM on the Freetalk dataset. The HMM-GPD was studied in the field of activity recognition and showed a good view-invariant performance. The classification was performed on sequences of graphs extracted from partially occluded skeleton models. The MFN and Kalman filter for classifier fusion was studied on the AVEC datasets and achieved good results in comparision to other approaches. Furthermore, it was shown that they outperformed classic point-wise and windowed fusion approaches. A comprehensive study analyzing the ULA showed that the FCHMM is well-suited to recognize states on different layers given unsegmented sequential data. A dynamic Markov logic network implemented the probabilistic logical rules in the uppermost layer. The thesis further presents a new dataset which was recorded in order to study the BLA.

The development of a CTS brings new challenges to the recognition of user's state and his environment. The presented work identifies important properties in this area and proposes and evaluates methods tailored to this operational area.

Kurzfassung

Moderne Computer Systeme haben unser Leben von Grund auf verändert und steigern unsere Effektivität bei der Bewerkstelligung von Arbeits- und Alltagsaufgaben. Aktuelle Systeme beschränken sind jedoch auf eine direkte Eingabe von Befehlen. Des Weiteren können sie aufgrund eines Mangels an Information nicht aktiv Entscheidungen im Interesse des Nutzers treffen. Kognitiv-technische Systeme (engl. Abk.: CTS) greifen dieses Defizit auf indem sie den Zustand und die Umgebung des Nutzers mittels Sensorik erkennen. Die erkannten Daten werden in einer Wissensbasis gesammelt und von der Anwendung und dem Dialog Management zur Entscheidungsfindung verwendet.

In der vorliegenden Arbeit wurden neue Methoden zur sensorischen Zustandserkennung im Kontext von CTS für die Mensch-Maschine Interaktion entwickelt und empirisch evaluiert. Im Fokus stehen multimodale und temporale Multi-Klassifikator Systeme. Des Weiteren werden die Themenbereiche sequentielle Klassifikatoren, Behandlung von partiell-verfügbaren Informationen und Integration von sub-symbolischen und symbolischen Informationen zur Erkennung von komplexen Zuständen bearbeitet. Folgende Methoden werden in dieser Arbeit vorgestellt: Ensemble Gaussian Mixture Model (EGMM), Conditioned Hidden Markov Model (CHMM), Fuzzy Conditioned Hidden Markov Model (FCHMM), Hidden Markov Model Graph Probability Densities (HMM-GPD), Markov Fusionsnetzwerk (MFN), Kalman-Filter für die Klassifikatorfusion und geschichtete Klassifikatorarchitekturen. EGMM erweitern das klassische GMM durch die Ensemble Technik für eine robustere Dichteschätzung. CHMM und FCHMM erweitern das HMM um eine zusätzliche kausale Sequenz, welche die versteckten Zustände beeinflusst und zur Integration von symbolischen Informationen verwendet werden kann. Das FCHMM verwendet Fuzzy-Werte innerhalb der kausale Sequenz. Die HMM-GPD führen Graphdichten als HMM Observationen ein. MFN und Kalman-Filter für die Klassifikatorfusion beschreiben probabilistische Ansätze zur temporalen und multimodalen Fusion von Entscheidungen, welche robust gegenüber Sensorausfall sind. In dieser Arbeit werden die Unidirectional Layered Architecture (ULA) und Bidirectional Layered Architecture (BLA) vorgeschlagen. Beide Architekturen erkennen komplexe Klassen mittels probabilistischer logischer Regeln und der zeitlichen Abfolge von Basismustern. Jede Schicht erkennt Muster basierend auf den Vorhersagen der darunterliegenden Schicht. Die BLA propagiert zusätzliche Information in die Richtung der tieferliegenden Schichten. Die empirische Evaluation der Methoden wurde mit Datensätzen aus dem Bereich Emotions- und Aktivitätserkennung durchgeführt, unter anderem dem Freetalk Datensatz, AVEC 2011, AVEC 2012, AVEC 2013 and UUlmMAD.

Das EGMM zeigte sich im Vergleich robuster und genauer als konventionelle GMM Ansätze. Zudem wurde gezeigt, dass die Auswahl von geeigneten Parametern durch die EGMM deutlich vereinfacht wird. Die multimodale Fusion von Klassifikatorausgaben auf dem Freetalk Datensatz mit dem CHMM lieferte bessere Ergebnisse als mit dem HMM. Die HMM-GPD wurde im Bereich der Aktivitätserkennung untersucht und zeigte die gewünschte Perspektiven-Invarianz bei der Klassifikation von Graph-Sequenzen extrahiert aus teilverdeckten Skelettmodellen. Das MFN und die Kalman-Filter für die Klassifikatorfusion wurden auf den AVEC Datensätzen untersucht und erzielten im Vergleich zu anderen Ansätzen sehr gute Ergebnisse. Weitere Studien zeigten, dass die neuen Ansätze eine bessere Leistungen aufweisen als die klassischen Punkt und Fenster-Fusionenansätze und zudem robust gegenüber Sensorausfällen sind. Die Studie zur Analyse der ULA zeigte, dass das FCHMM gut geeignet ist um Nutzerzustände auf kontinuierlichen Daten und auf unterschiedlichen Schichten zu erkennen. Ein dynamisches Markov Logic Network modellierte zudem die probabilistischen logischen Regeln in der obersten Schicht. Des Weiteren führt die Arbeit einen neuen Datensatz zur Untersuchung der BLA ein.

CTS stellen neue Herausforderungen an die Erkennung von Zuständen und Umgebung des Nutzers. Die vorliegende Arbeit identifiziert wichtige Eigenschaften in diesem Bereich und entwickelt und evaluiert neue daraufhin angepasste Methoden.

Danksagung

Diese Doktorarbeit entstand am Institut für Neuroinformatik an der Universität Ulm unter der Leitung von Prof. Dr. Günther Palm und wurde finanziert durch den Sonderforschungsbereich Transregio 62 “Eine Companion-Technologie für kognitive technische Systeme”, welches durch die Deutschen Forschungsgemeinschaft (DFG) eingerichtet wurde.

Zu Beginn möchte ich mich bei Prof. Dr. Palm für die spannende Aufgabenstellung, die lehrreiche Zeit sowie die Begutachtung meiner Arbeit bedanken. Meinen Dank gilt zudem Dr. Friedhelm Schwenker, welcher meine Arbeit mitbetreute. Einen weiteren Dank möchte ich an meine Gutacher Prof. Dr. Günther Palm, Prof. Dr. Thomas Martinetz, Prof. Dr. Barbara Hammer sowie an meine Prüfer Prof. Dr. Michael Weber und Prof. Dr. Wolfgang Minker richten, welche freundlicherweise meine Arbeit bewerten. Einen besonderen Dank möchte ich an dieser Stelle ebenfalls an den Vorsitzenden des Promotionsausschusses Prof. Dr. Uwe Schöning aussprechen.

Danken möchte ich ebenfalls meinen Kollegen aus der Neuroinformatik mit denen ich besonders eng zusammenarbeiten durfte: Markus Kächele, Georg Layer, Sascha Meudt, Martin Schels, Stefan Scherer, Miriam K. Schmidt, Stephan Tschechne, Felix Heilemann und Florian Gawrilowicz. Die fachlichen Gespräche während der Mittagspause werden mir stets in Erinnerung bleiben. Einen Dank für den Zusammenhalt möchte ich an dieser Stelle ebenfalls an alle Mitarbeiter des Instituts richten. Den Mitautoren gilt mein Dank für die zahlreichen guten Ideen, konstruktiven Kritiken und tatkräftige Unterstützung. Insbesondere möchte ich an dieser Stelle Martin Schels, Stefan Scherer, Edmondo Trentin, Georg Layher und Thomas Geier danken. Allen Mitarbeitern des Sonderforschungsbereiches, allen voran Ihren Sprechern Prof. Dr. Susanne Biundo und Prof. Dr. Andreas Wendemuth, möchte ich danken für die spannenden Forschungsfragen, anregenden Unterhaltungen und engagierte Zusammenarbeit.

Darüber hinaus geht mein Dank an meine Familie. An meine Mutter, dafür, dass sie stets an mich geglaubt hat und mich immer in allen Dingen unterstützt. An meine Schwester, dafür, dass sie mich mit ihrem Schreiben inspirierte und tatkräftig zusammen mit Ihrem Freund Steve bei der Korrektur dieser Arbeit unterstützte. An dieser Stelle möchte ich auch meinen Vater gedenken, den ich nun schon seit so vielen Jahren schmerzlich vermisse, und ihm danken für alles was er mich lehrte und nun in meinen Augen verkörpert.

Table of Contents

Chapter 1

Introduction

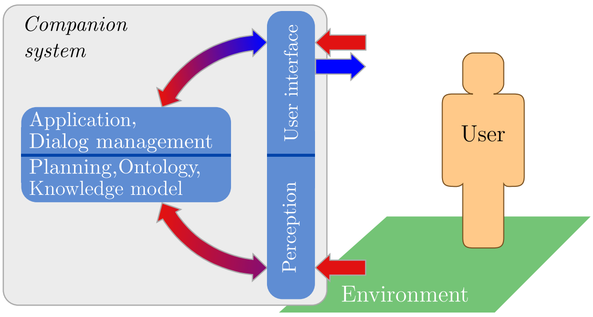
Over the last decades, the development in computer science has fundamentally changed our society. Computer systems facilitate our work, help us to organize our life and have become a major part of our spare time. However, although they are ubiquitous in our everyday life, human-computer interaction (HCI) is still limited to passive devices such as keyboard, mouse and touch events which forces computer systems to act only responsively on the basis of directly triggered events. HCI can be regarded as being rather one-directional, since computers cannot recognize our intentions, interpret ongoing actions and are not able to access the meaning of situations. The interaction is limited to services being exclusively triggered by users. Hence, computer systems are unable to instantaneously adapt their functionality to everyday demands of users. A change of this paradigm requires the development of new technologies which enable computer systems to perceive the environment, to gather useful information about the user and to make pro-active decisions on behalf of the user. Computer systems possessing these technologies are called *Companion systems* which provide the framework for this thesis.

Companion system

The development of Companion systems is subjected to a number of challenges when compared with conventional systems. One way to identify these challenges is to divide the input to a Companion system into three major categories which are virtually omnipresent in the new HCI paradigm: (1) the explicit user input, e.g. keyboard or multimodal instructions by gesture and speech; (2) the implicit user input (Schmidt, 2000), e.g. emotion or disposition (Scherer et al., 2012a); and (3) the recognition of the user's environment including the context of use (Dey and Abowd, 1999; Honold et al., 2013), e.g. location, activities or the state and manipulation of objects nearby. These categories show that Companion systems must have a strong perception and, therefore, should be regarded as *cognitive technical systems* (CTS) (Gamrad, 2011). However, it has to be pointed out that the concrete

Cognitive technical system

Figure 1.1. Decomposition of a Companion system. The user interface allows explicit inputs (red arrow) and provides outputs to the user (blue arrow). The implicit input (red arrow) is realized by the perception which is monitoring the user's state and the situative context from the environment. The system is decomposed into a peripheral building block, i.e. user interface and perception, and an inner building block which realizes higher cognitive abilities. The bidirectional arrows represent the information flow between these building blocks. For a detailed description please refer to the text.


Companion system
User interface
Application, Dialog management
Planning,Ontology, Knowledge model
Perception
User
Environment

manifestation of the input categories strongly depends on the application at hand.

In order to recognize the events of interest, Companion systems must implement several requirements at once. Events of interest are often characterized by having a high variability and by being only weakly represented in observable features. Furthermore, these events are likely to have a temporal dimension and to be present in multiple modalities. On major difficulty is that unexpected events may occur in an open world scenario. This can lead to wrong recognitions since these events are unknown to the Companion system. Real-world scenarios can even lead to situations in which CTS have to deal with missing data or even with sensor failures which further complicates the recognition of the desired events.

A deeper analysis can be performed by taking an alternative view in which a conceived Companion system is decomposed into a set of basic modules in order to examine the system's information flow (Glodek et al., 2015a). Figure 1.1 depicts the internal information flow within the system and the information flow between the user, the environment and the system. The system itself is organized in two building blocks: (1) a peripheral building block comprising a user interface module and a perception module and (2) an inner building block comprising a knowledge model module and a decision making module which realize planning, ontologies, the controlling application and the dialog management. The red input arrow in the lower right represents the information entering the perception module of the first building block, i.e. the implicit user input. The perception module is connected to the knowledge module by a bi-directional arrow which indicates that information should be exchanged in both directions in a seamless manner. Information is passed towards the knowledge module such that it can be processed by abstract symbolic artificial intelligence approaches. However, information should also be passed in the reverse direction such that the perception can benefit from the derived context utilizing qualified priors. On the upper right, red and blue arrows represent the conventional explicit input and output of the system. The commands which are combined and interpreted by the user interface module are, if necessary, proceeded to the inner decision making module which then adapts the knowledge model, planning and dialog management accordingly. The bi-directional arrow indicates that the output should be tailored to the user and his current input preference. Therefore, these arrows represent not only the fusion of information but also a fission in which high-level knowledge is transferred back to the periphery (Honold et al., 2012). The implementation of the bi-directional information flow inside the system is challenging because each of the four modules faces characteristic requirements: (1) perception has to be *omni-present and recognized classes must have a sufficient level of abstraction*, (2) knowledge has to *handle the perceived information dynamically, and still be persistent for further processing*, (3) decision making has to be *pro-active* and (4) user interface has to be *concise (effective) and instantaneously available.* The bi-directional arrows represent the transformation of information from one module to another, while still meeting the requirements of the individual modules. This work focuses on pattern recognition and the seamless bidirectional exchange of information with the knowledge module in the context of CTS perception (Glodek et al., 2015a).

1.1. Research Questions

In the following, three research questions will be formulated which are of central importance to this work:

(i) How to combine multimodal and temporal probabilistic classifier results?

(ii) How to handle missing information in sequential classifiers?

(iii) How to create high performance classifiers for complex classes?

The remainder of this section will clarify and elucidate the three questions in greater detail.

(i) ***How to combine multimodal and temporal probabilistic classifier results?***
The first question addresses a classic topic of pattern recognition, namely late classifier fusion in which decisions from multiple classifiers are combined to increase the over-all performance. In general, late classifier fusion is done by independently classifying the same set of classes which are directly observable and combining the resulting decisions subsequently. In CTS, patterns often have a natural expansion over time such that late classifier fusion needs to be extended to temporal domain, i.e. the combination needs to be performed with decisions from multiple modalities or feature views and multiple time steps. A large number of approaches have been proposed to combine non-temporal classifier decisions. In case of crisp classifier decisions, the combination is commonly performed using majority vote, i.e. the decision occurring most frequently is selected (Kuncheva, 2004, page 112ff.). However,

the classifier fusion techniques based on crisp decisions have their limitations since the uncertainties associated with the classifier decisions are disregard (Thiel, 2010). In contrast, probabilistic classifier outputs allow more sophisticated combinations, such as the sum or product rule (Kittler et al., 1998). More advanced combination approaches are adaptive fusion mappings (Schwenker et al., 2006) or Dempster-Shafer classifier fusion (Szczot et al., 2012). However, these approaches are still missing a unified formulation for temporal classifier fusion. Dietrich et al. (2004) were among the first groups that systematically extended classic fusion techniques to temporal fusion. They focused on the question whether fusion over time should be performed before or after the classic classifier fusion.

The research question is addressed by presenting and analyzing novel multimodal and temporal fusion techniques. A special focus is set on probabilistic methods and the handling of missing classifier decisions. Furthermore, the incorporation of measured uncertainty and the applicability to real-world scenarios plays an important role.

(ii) ***How to handle missing information in sequential classifiers?***

The development of CTS which are deployed in real-world scenarios demands for techniques which allow the system to deal with multiple kinds of internal exceptions. Exceptions can take place on almost all stages of the system's architecture. However, especially the acquisition of information before classifier fusion can be affected. For instance, a complete modality may fail such that alternative modalities need to take over. This can for example concern the audio modality in case the person remains silent or is located in a noisy environment, or the video modality in case a monitored person leaves the camera's field of view. In these circumstances, the classification has to permanently or temporarily resign from features of one entire modality. However, features can also become partially unavailable, e.g. persons may become partially occluded due to the view-point of the camera. These incomplete features vectors represent a real challenge for classification. Complete loss of feature vectors can be handled at fusion level and, therefore, is addressed by the first research question. Incomplete feature vectors can be efficiently dealt with at the stage of classification. This case is addressed by this research question.

The most common strategy is to discard incomplete feature vectors and to use only the feature vectors which are complete. Classification of sequences which are composed of complete and lost feature vectors can be performed by repairing the sequence on feature or on decision-level with the help of past and future information. The reconstruction on feature-level can be done using a Markov chain which models hidden causes emitting the feature vectors (Peursum et al., 2004; Chen et al., 2006). The reconstruction on decision-level is, as already mentioned, addressed by the first research question. However, such kind of reconstruction can compensate only short-time losses. An alternative strategy to handle this issue is to create a fixed-length feature vector representation by clustering a large variable-sized collection of available feature vectors using the so-called bag-of-words approach (Fei-Fei and Perona, 2005). Again, this approach tends to wash out and discard valuable feature information. The best treatment of incomplete feature vectors in a sequence is to avoid the loss of valuable information by not dealing with the incompleteness before classification. Instead, the classification algorithm has to use what is available and to disregard just the missing parts.

To do so, the starting point has to be a sequential classifier which is capable of propagating information temporarily. In addition, the classifier must be able to represent incomplete feature vectors. Such an algorithm handles the issue at classification-level which is reasonable since some classes do not depend on the observation of complete feature vectors. As a consequence, the number of sensors can be reduced and an increased classifier performance can be expected. Furthermore, such a classifier would allow the training on datasets also containing incomplete feature vectors.

(iii) ***How to create high performance classifiers for complex classes?***
Companion systems which adapt to the user's demands and attitudes are in need of high-level knowledge to perform their decision making. However, simple stand-alone classes, e.g. detected single objects or the presence of a face in an image, are insufficient and cannot be regarded as useful information for further processing in the inner building block. Since the decision making module utilizes methods from symbolic artificial intelligence (AI), it is essential to recognize complex classes with high-level relations from which new information can be derived. An example of a useful recognition could be "Person A is currently sitting in front of the home-entertainment-system drinking a cup of coffee while having breakfast". The "Person A" is grounded to a distinct individual and is related to his location and actions. The entities, their relations and already available prior knowledge can be associated to perform further valuable reasoning (Möller and Neumann, 2008). However, from a pattern recognition perspective it is still not clear how to detect such complex patterns. It is worth noting that sub-patterns which are directly observable, such as "Drinking coffee", "Reading newspaper" and "Eating toast", compose complex categories, such as "Having breakfast".
Technically, the composition of observable sub-patterns to complex patterns has to account for uncertainty and needs to be performed in multiple moderate steps. This procedure can be regarded as a seamless transition from sub-symbolic knowledge to symbolic knowledge.

The recognition of complex patterns bears many challenges: A robust detection of sub-patterns can benefit from multiple modalities, temporal fusion and the preservation of uncertainties. In addition, it is important to use sequential classifiers to account the temporal nature of patterns in real-world scenarios. On higher levels, a smooth shift to symbolic AI methods is inevitable. Pattern recognition approaches at this level would require datasets with a large number of samples of complex patterns. However, this is practically infeasible because complex patterns have a high variability in their appearances. Symbolic AI methods on the other hand often rely on crisp spatial or temporal relations. Such a crisp logical framework is less suitable in the given application since the recognition of basic patterns are typically afflicted with uncertainty. A good choice is the use of a probabilistic logical framework which allows complex classes to recover from wrongly recognized subpatterns (Domingos et al., 2006). It is even thinkable that information which was extracted at the level of complex patterns can be propagated back to the recognizers closer to the sensors in order to enhance the overall performance.

1.2. Related Work

Companion systems need to perceive the situative context and the user state in order to correctly assess the condition the user is in. The user state refers to the collection of states restricted to the user, e.g. emotion, common knowledge or body pose without any relations to the environment and the system. As opposed to that, the situative context deals with spatial and temporal relations between the user and the environment. Examples in the spatial domain are relations between user and objects, e.g. a user can interact with an object or a user expresses an emotion towards an object (disposition), but also relations between two objects itself, e.g. one object is part of another. The temporal domain relates events to each other. For example, a complex activity can be composed of a sequence of shorter actions, or a change of emotion or knowledge can be triggered by information which was passed to the user. The user state and the situative context are both stored in the knowledge model for further processing.

The research questions are going to be addressed in the context of Companion systems and, hence, with a special focus on the recognition of the user state and situative context. Therefore, the first research question is going to be examined taking the application of emotion recognition in HCI. The second research question is addressed by using the example of action recognition, whereas the last research question is concerned about the recognition of user preferences.

In the following, an overview about the related work which has been achieved regarding these three topics is given.

1.2.1. Emotion Recognition in Human-Computer Interaction

The research on emotion has evolved over a large period of time (Cannon, 1927; Ekman, 1972; Russell, 1980; Mehrabian, 1996; Marsella and Gratch, 2009). One of the first approaches to explain the origin of emotions is the James-Lange theory which was proposed in 1884 (Cannon, 1927). Almost a century later, Ekman (1972) published his theory of universals and cultural differences in facial expression of emotions. According to this theory, humans have a pan-cultural set of emotions which is expressed according to cultural display rules. The genesis of an emotion can be divided into three stages: elicitation in which different events may induce a different emotion depending on the cultural background; the presence of a universal emotion; and the display of this particular emotion which again depends on the cultural background. In the years that follow, advanced models to structure emotions were developed. Russell (1980) proposed a two-dimensional circumplex model in which emotions are spanned in the pleasure-arousal space. Later the two-dimensional model was extended by a third dimension — the well-known pleasure-arousal-dominance model (Mehrabian, 1996). Ambady and Rosenthal (1992) showed that even short observation snippets of expressive behavior allow accurate social perception. The study showed that humans are able to efficiently communicate social information via unintended and unconscious cues. The categorization and studies on emotion perception gave important evidences which motivate the application of automated affective state recognition in HCI.

The automated recognition of emotions in HCI is a challenging task (Palm and Glodek, 2013). The target emotional categories must be present in the actual Companion application. The dataset recorded for training must be accurately annotated. Robust features must be extracted from the modalities of the dataset, and the trained classifiers must be robust and provide a measure for the uncertainty associated with the classification. Furthermore, The situative context of the subject should be inferred in order to assess the user's disposition.

First approaches of automated emotion recognition restricted themselves to unimodal and acted datasets (Kanade et al., 2000; Tian et al., 2001; Lee et al., 2004; Wagner et al., 2007; Maganti et al., 2007). The accuracies on these datasets range between 70% to 80% which is close to the human emotion recognition performance (Kanade et al., 2000; Burkhardt et al., 2005; Bänziger and Scherer, 2007). Later approaches were evaluated on multimodal datasets with naturalistic emotions which are clearly more demanding with respect to annotation and pattern recognition (Douglas-Cowie et al., 2003; Campbell et al., 2006a; Martin et al., 2006; Douglas-Cowie et al., 2011; Strauß et al., 2008; McKeown et al., 2010; Schuller et al., 2007; Zeng et al., 2009). The accuracies of these approaches vary strongly depending on the application, the dataset and the annotation. In the naturalistic setting, it is generally difficult to obtain a ground truth annotation since there is no exact and indisputable definition of emotion. Labels are strongly related to the dataset such that distinct emotional categories are often insufficiently covered, e.g. "Disgust" or "Fear". The annotation is further complicated by weakly expressed emotions and unexpected events which appear only once in a dataset or are not covered by the annotation schema (Douglas-Cowie et al., 2011; Strauß et al., 2008).

However, that kind of datasets also offer many opportunities, e.g. multiple modalities, temporal evolvement and context information (Scherer et al., 2012a; Metallinou et al., 2012; Wöllmer et al., 2010; Wendt et al., 2008; Batliner et al., 2008). Cowie et al. (2001) argued that the multi-faceted categories of emotions can be captured by artificial systems using the two-dimensional representation of "Activation" and "Valence". Furthermore, the authors pointed out that emotions occur in multiple temporal granularities, e.g. expression, attitude, mood or trait. Vinciarelli et al. (2008) stated that social signals are indispensable in human interaction and, therefore, are an essential part of next-generation computing. The machine analysis of a social interactions can be realized by observing multiple behavioral cues which together compose a social signal. This topic was further addressed by Scherer et al. (2012a); Scherer (2011) who proposed to investigate the user's disposition towards the interacting system, e.g. involvement, agreement, understanding.

In the next step of the Companion system development, the output of the automated emotion recognition needs to be integrated into the processing of the knowledge and decision making modules. However, so far the role allocation between humans and the emotionally aware CTS is undefined. This issue makes it difficult to perform an importance rating of emotional categories. Hence, the behavior towards such a CTS is commonly studied with the help of a Wizard-of-Oz experiment (Kelley, 1984; Frommer et al., 2012; Walter et al., 2011; Bertrand et al., 2011; Strauß et al., 2008). In a Wizard-of-Oz experiment, the participant interacts with a sophisticated computer system without knowing that it is remote-controlled by an investigator, the so-called wizard. Datasets which are derived based on recordings of such Wizard-of-Oz experiments are counted to be the most challenging ones and, therefore, are going to be addressed in this work.

1.2.2. Human Action Recognition

Action recognition is a vivid research field in pattern recognition offering a multitude of difficulties (Poppe, 2010; Ahad et al., 2008; Turaga et al., 2008; Layher et al., 2012). Sheikh et al. (2005) identified three main challenges: view-invariance, execution rate and anthropometry of actors. These challenges can be solved on different stages of processing. The anthropometry is ideally addressed on an early feature extraction stage, while the execution rate is most efficiently handled during classification. To achieve view-invariance, the feature extraction and the classification need to efficiently interplay with each other to gain as much information as possible from the sequential data representing an action, as already pointed out in Section 1.1. However, only few methods have been proposed which address view-invariance in such a way.

According to Poppe (2010), image-based representations for human action recognition can be divided into global and local representations. The global representation is obtained in a top-down fashion starting with a localization of the person and a subsequent encoding of the corresponding region. One approach is to extract fixed-length features from the human silhouette, e.g. edges and optical flow. Bobick and Davis (2001) proposed to extract the so-called motion-energy by cumulating the differences of binary silhouettes between frames. Chen et al. (2006) used a fixed-length feature vector derived from a star skeleton, i.e. an optimal fit of lines having a common origin in the contour of a human. Global features require a robust detection of the region of interest. Finer structures are often disregarded and the application to recordings in which the region of interest is partially occluded is very limited.

Alternatively, local image representations make use of a collection of independent patches which are sampled from an image. Sampling can be done densely over the complete image (Poppe, 2010) or using an interest point detector, e.g. the scale space extrema detector (Cheung and Hamarneh, 2007) or the 3D Harris Detector (Gorelick et al., 2007). The patches in the image are then used to extract features such as histogram of gradients (Dalal and Triggs, 2005) or histogram of optical flow (Lucas et al., 1981). Therefore, features are extracted from patches of interest without knowing whether they actually belong to the human and despite a possible partial occlusion of the human. Hence, the problem of occlusion is not solved but stalled, since it has to be handled by the classifiers operating on the features. Thus, the occlusion problem is often addressed by recording a dataset which has a large number of samples such that the machine learning algorithm can learn a sufficient statistic on the complex feature distribution.

Apart from the global and local image representations, Poppe (2010) further outlines the application-specific representation in which rich image features are extracted especially for the domain of action recognition. These features are commonly given by skeleton models which are composed of joint locations and angles. In case the extraction is performed based on a plain RGB image, the resulting the skeleton model is typically limited to two-dimensional coordinates. Alternatively, a more sophisticated motion capturing system can be utilized which provides view-invariant three-dimensional joint locations. However, such a system is in general expansive and obtrusive. A good compromise, is a three-dimensional skeleton model derived using a depth-map camera, e.g. the Kinect camera[1]. Depth-map cameras are not view-invariant due to their single viewpoint (Shotton et al., 2013). However, skeleton models can be extracted from the visible part of the subject and occluded parts can be marked as such. The biggest benefit is that these cameras are available on the retail market.

Many experiments treat view-invariance as being solved by making use of fully observable (or at least advantageously recorded) data (Schuldt et al., 2004; Tran and Sorokin, 2008).

[1] The Kinect™ camera is an input device developed by Microsoft® which captures human body poses.

Nonetheless, there is some literature addressing the topic of view-invariance in human action recognition with new features and/or classification techniques (Ahad, 2011). Ben-Arie et al. (2002) proposed a method for activity recognition which is based on multi-dimensional indexing and voting. Each of the tracked body parts acts as an expert which independently votes for an activity. The votes are then averaged and evaluated over time using sequencing. The use of body parts to group independent experts allows a native handling of occlusions, which albeit was not addressed in the study. However, the overall architecture does not rely on standard classifiers, and neither on a probabilistic framework which makes it difficult to predict the performance with respect to occlusion and view-invariance. Peursum et al. (2004) introduced a modified hidden Markov model (HMM) for action recognition using discrete observations based on a star skeleton representation. In case the pose estimation fails to extract a valid star skeleton, the observation is completely discarded. The HMM deals with the incomplete sequence by omitting the emission probabilities in case observations are missing. However, star skeletons which were marked as being invalid might still contain valuable information which could have been exploited. One can argue that the extraction of data was not performed efficiently.

1.2.3. Complex Pattern Recognition

In literature, complex class recognition usually focuses on a particular problem rather than evaluating a generic approach (Turaga et al., 2008). Hence, the comparison of proposed approaches is not straight forward, also because the addressed problem definition often relies on a dataset which is not publicly available. Nonetheless, the state-of-the-art in this area has evolved and the progress made so far is encouraging. Rao et al. (2002) proposed to recognize high-level action classes by extracting the curvature of a hand trajectory. The curvature is fragmented into instants and intervals. An instant is an entity that occurs for only one frame and represents an important change in the motion characteristics. An interval represents the time period between two instants. The view-invariant representation is then classified using an algorithm matching the current observed curvature to a set of known curvatures. Wrede et al. (2004) developed a cognitive vision system for scene analysis which makes use of an active memory. The system has the ability to relate objects to actions. The observed concepts, e.g. hand, typing, keyboard, are stored in a volatile memory. The content of the memory is processed by additional algorithms, e.g. a consistency validation, which would reject the recognition of certain actions in case no supporting objects were found in the scenery. An approach to recognize office activities using the context of manipulated objects was proposed by Biswas et al. (2007). The context is modeled using a Markov logic network (MLN) which operates on multiple weak feature sources, i.e. object information, body pose and body movement. The combination of the observed features in the MLN resolves ambiguities and, thus, provides a more robust recognition. Tran and Davis (2008) presented a system for the surveillance of an outdoor parking lot. Despite the noisy observations, the system recognizes humans and cars with the goal to derive high-level information using an MLN, e.g. people shaking hands, trunks get loaded or a person gets into a car. Kembhavi et al. (2010) developed a system for scene understanding which utilizes an MLN to combine image analysis and reasoning. The key concept is to assign a functionality to zones in the scene which were derived in a pre-processing segmentation step, e.g. roads, sidewalks, crosswalks, pedestrian entrances or bus-stops. The MLN input is given by the segmented zones, tracked objects over time, e.g. bus, car or pedestrian, and world knowledge in the form of rules, e.g. “Cars stop at crosswalk in order to let people pass”. Oliver et al. (2004) proposed a multi-layered architecture to recognize office activities based on multiple sources. The lowermost layer recognizes basic patterns. Each following layer performs the classification based on the output of the proceeding layer. Hence, with each layer the temporal granularity and the level of abstractness increases such that more complex classes are recognized. Gehrig et al. (2011) utilizes a similar multi-layered approach to detect human intentions based on six activities and motion primitives. Domain knowledge is applied (by the means of labeled ground truth to learn transition probabilities) either to recognize motion, activity or both.

1.3. Organization of the Thesis

The remainder of this manuscript is structured as follows.
Chapter 2 provides a thorough compendium of the work which constitutes the basis of this thesis. The focus is set on probabilistic models, multiple classifier systems and support vector machines (SVM). Relevant probabilistic models for this work are: Gaussian mixture model (GMM), graph probability density (GPD), hidden Markov model (HMM), latent-dynamic conditional random field (LDCRF), Kalman filter and Markov logic network (MLN). Furthermore, an introduction to the terminology and basics of multiple classifier systems is given. This is followed by a detailed presentation of the layered HMM (LHMM) for complex class recognition as proposed by (Oliver et al., 2004). Finally, an introduction to fuzzy-in fuzzy-out multi-class SVM is provided.

Chapter 3 presents the contributions being made based on the basics presented in the previous chapter. First, the ensemble GMM (EGMM) is introduced which provides more robust estimations of probability densities than classic approaches with the help of an ensemble of GMM. Thereafter, the conditional HMM (CHMM) and fuzzy conditional HMM (FCHMM) are described which both extend the standard HMM by a sequence of outer causes influencing the hidden states. The FCHMM additionally features fuzzy sequences of outer causes. Following this, HMM using GPD (HMM-GPD) are proposed which model sequences of variable-sized graphs as observations. The CHMM using GPD (CHMM-GPD) further allows the training of a unified model which shares hidden states between classes. The Markov fusion network (MFN) and Kalman filter for classifier fusion are proposed to combine a continuous stream of multimodal classifier decisions. This is followed by a presentation of the unidirectional layered architecture (ULA) and bidirectional layered architecture (BLA). The ULA is an extension of the LHMM which is still limited to an unidirectional information flow towards the upper layers. The BLA further allows a back-propagation of information from the upper layers recognizing complex classes to the lower layers operating on data derived from the sensors. Subsequently, an alternative fuzzy-in fuzzy-out multi-class SVM is presented which is called the inequality constraint multi-class fuzzy-in fuzzy-out SVM (IC-MC-F^2-SVM).

A number of empirical studies were conducted in order to evaluate the proposed approaches. The empirical evaluations have a strong focus on real-world datasets addressing among others emotion recognition in HCI, action recognition and user preference recognition. Chapter 4 introduces the datasets and evaluation criteria. The results of the empirical evaluations are then presented in Chapter 5. Each evaluation of a study is followed by a short summary which shortly recapitulates and interprets the results.

The discussion of the proposed methods and their empirical evaluations are provided in Chapter 7. Finally, Chapter 7.1 provides a conclusion to the work done which also addresses the three research questions raised in this introduction. The last chapter further gives an outlook on interesting aspects of possible future work.

Chapter 2

Basics

The introduction expounded the requirements, context and challenges to be addressed in this work. The approaches which will be proposed in the subsequent chapter rely on a large number of basic methods which are going to be presented in this chapter. The work puts a strong focus on probabilistic graphical models. Consequently, this chapter will begin by discussing the concepts of probability theory, graphical models and the most frequently utilized models of the thesis, i.e the Gaussian mixture model (GMM) and the hidden Markov model (HMM). Following this, a brief outline of the HMM-related Kalman filter and the Markov logic network (MLN) is given. Then the scope is broadened to alternative classifier approaches and architectures, starting with the concepts of multiple classifier systems and layered HMM (LHMM). Finally, the state-of-the-art regarding fuzzy-in fuzzy-out support vector machines is presented.

2.1. Probability Theory

In our (real) world only a few laws are absolute — in most cases outcomes and predictions of events can be considered as being rather uncertain (Bloch et al., 2001; Palm, 2012). Several mathematical theories have been developed to describe the uncertainty of such events (Thiel, 2010, page 41ff.; Pearl, 1988;Shafer, 1986). One of the most elementary theories is the "frequentist" setting which is built on the assumption that experiments are reproducible such that a statistical evaluation of the occurrences of an event can be performed (Bortz and Schuster, 2010; Bishop, 2006). Such kind of probabilities are also often referred to as "objective probabilities" because they are well-founded on a multitude of observations. However, it is not always possible to observe an event frequently enough to setup sufficient statistics. For instance, it is not possible to create a statistic on an outcome of a sport game using the frequentist setting, since it depends on factors which were never observed before, e.g. the combination of players on the field or the physical state of the players.

Nonetheless, human experts are still able to give a subjective prediction on how likely outcomes of a unreproducible event may be (Casella and Berger, 2001). The most popular paradigm based on “subjective probabilities” is the Bayesian theory (Koller and Friedman, 2009), although alternative approaches such as the Dempster-Shafer theory or the possibility theory have attracted a reasonable amount of interest in recent times (Dempster, 1967; Heinsohn and Socher-Ambrosius, 1999; Dubois, 2006).

In the next section, a short introduction to the frequentist setting will be given which is followed by an overview of the Bayesian theory with a special focus on graphical models, i.e. Bayesian networks and Markov networks.

2.1.1. Frequentist Setting

The *frequentist setting* is based on the counting of observed events. Let x and y be two random variables each one modeling an event. possible outcomes can be assigned to the random variables x or y from a closed set of states $\{x_1, \ldots, x_{M_\mathsf{x}}\}$ and $\{y_1, \ldots, y_{M_\mathsf{y}}\}$. The joint probability of observing a certain assignment of the two random variables depends on the number of observed outcomes in the past

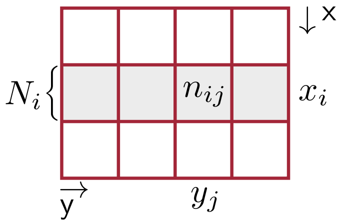
$$p(\mathsf{x} = x_i, \mathsf{y} = y_j) = \frac{n_{ij}}{N} \tag{2.1}$$

where n_{ij} denotes the number of observed outcomes $(x_i,\ y_j)$ and $N = \sum_{i=1}^{M_\mathsf{x}} \sum_{j=1}^{M_\mathsf{y}} n_{ij}$ denotes the total number of all observations. In order to ensure that this probability is unbiased, the observed outcomes need to be drawn from the same experiment and need to be independent from each other. In other words, the observations have to be *independent and identical distributed* (i.i.d.). This property holds for all statistical approaches in this work and, therefore, is generally assumed as being given. Furthermore, it is mandatory that a sufficient number of outcomes have been observed for each random variable to ensure sufficient statistics. In the given setting, the conditioning of the random variable y being in state y_j given an outcome x_i is obtained by normalizing the distribution of events

$$p(\mathsf{y} = y_j | \mathsf{x} = x_i) = \frac{n_{ij}}{N_i} \tag{2.2}$$

where $N_i = \sum_{j=1}^{M_\mathsf{y}} n_{ij}$. Figure 2.1 depicts the counting of observed events in the frequentist setting. Each observed outcome (x_i, y_j) adds one to the element n_{ij} of the matrix which was initialized with zeros. The divisor N_i which is used for the normalization is obtained by summing the elements along the line of x_i which is highlighted in gray.

Figure 2.1. Counting of events in the frequentist setting. Each observed event x_i and y_j of the random variables x and y increments the corresponding entry n_{ij} (Figure adapted from Bishop (2006), page 13).



Two rules can be derived based on the probabilities defined so far, i.e. the sum rule and the product rule. The *sum rule* which is also often referred to as *marginalization* is given by

Sum rule

$$p(\mathsf{x} = x_i) = \sum_{j=1}^{M_y} p(\mathsf{x} = x_i, \mathsf{y} = y_j). \tag{2.3}$$

The rule eliminates the effect of random variable y. In this setting, the matrix shown in Figure 2.1 reduces to a vector. The *product rule* can be used to derive the joint distribution with the help of a conditioned probability:

Product rule

$$p(\mathsf{x} = x_i, \mathsf{y} = y_j) = p(\mathsf{y} = y_j | \mathsf{x} = x_i) p(\mathsf{x} = x_i). \tag{2.4}$$

By applying the product rule and due to symmetry, the following equations can be obtained

$$\begin{aligned} && p(\mathsf{x}, \mathsf{y}) &= p(\mathsf{x}, \mathsf{y}) && (2.5) \\ \Leftrightarrow && p(\mathsf{y}|\mathsf{x}) p(\mathsf{x}) &= p(\mathsf{x}|\mathsf{y}) p(\mathsf{y}) && (2.6) \\ \Leftrightarrow && p(\mathsf{y}|\mathsf{x}) &= \frac{p(\mathsf{x}|\mathsf{y}) p(\mathsf{y})}{p(\mathsf{x})}. && (2.7) \end{aligned}$$

Although this equation looks similar to the Bayes' theorem which will be introduced in the next section it is still technically fundamentally different because of the frequentist setting. More details about this topic can be found in (Vapnik, 1999; Koller and Friedman, 2009; Bishop, 2006).

2.1.2. Bayesian Probability Theory

The starting point of the *Bayesian probability theory* is the assumption that the degree of uncertainty regarding states in the world can be assigned by experts using a subjective score within the interval of $[0, 1]$, where values close to one means that an event is very certain to occur and values close to zero indicate that the occurrence of an event is almost impossible. The key idea of the Bayesian theory is that hypotheses can be updated in the light of new data with the help of so-called priors. In doing so, the Bayesian theory follows precisely the same calculus as the one of the frequentist setting (Bishop, 2006, page 21ff.). The most famous application of the Bayesian probability theory is the *Bayesian theorem.*

Bayes' theorem

Let M be a model which is capable to generate data D. The theorem states that

$$p(\mathsf{M}|\mathsf{D}) = \frac{p(\mathsf{D}|\mathsf{M})p(\mathsf{M})}{p(\mathsf{D})}. \tag{2.8}$$

The *likelihood* $p(\mathsf{D}|\mathsf{M})$ describes how likely D originates from M. The quantities $p(\mathsf{M})$ and $p(\mathsf{D})$ are *priors* and represent the probability of the model itself, i.e. before taking the data into account, and the probability of observing the data alone (Koller and Friedman, 2009, page 19). The prior $p(\mathsf{D})$ is located in the divisor and normalizes the distribution. The left-hand term of the Bayes' theorem describes the so-called *posterior* distribution $p(\mathsf{M}|\mathsf{D})$.

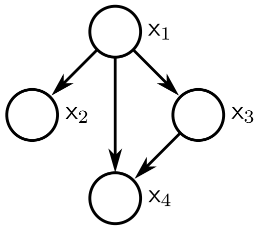
In machine learning, it is often desirable to derive the posterior of different models in order to compare them and to select the model with the highest probability. However in many cases, it is sufficient to compute only the model likelihood $p(\mathsf{D}|\mathsf{M})$ with disregard of the model and data priors (Koller and Friedman, 2009). The goal of machine learning is to find the most probable model which fits to a given dataset. This is usually achieved by selecting suitable parameters for a fix configurable model. The procedure is called *maximum likelihood* (ML), because the fitting of the parameters intends to increase the likelihood. Finding the parameters which directly maximize the posterior distribution is referred to as the *maximum a posterior* (MAP) approach. The posterior is obtained by weighting the likelihood using the prior $p(\mathsf{M})$ and normalizing the distribution using the data prior $p(\mathsf{D})$. However, in some applications it is favorable to omit the normalization, because of the extensive computational effort associated with evaluation of the data prior, e.g. in continuous state space. Fortunately, many applications are concerned only with finding the most likely model, rather than the exact posterior probability.

The theory of probability requires that all possible outcomes of a random variable are defined from the beginning. This requirement is called *closed world assumption.* An open world scenario can be defined with the help of a special event, e.g. a model for all uncovered events or a so-called *background model.* Alternative approaches to model uncertainty, such as the the Dempster-Shafer theory (Shafer, 1976), provide a more intuitive framework for the *open world assumption.* However, the Dempster-Shafer theory requires the power set of all states to be evaluated.

2.1.3. Bayesian Networks

Graphical models reduce the complexity of joint distributions by decomposing them into a set of smaller probability distributions. Hence, the number of parameters required to describe the distribution is reduced (Pearl, 1988; Heckerman, 1998; Bishop, 2006; Koller and Friedman, 2009). The idea of *Bayesian networks* is to exploit causal dependencies

Figure 2.2. Distribution decomposed by a Bayesian network. The joint distribution given by $p(\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, \mathrm{x}_4)$ can be decomposed to factors of conditioned probability distributions $p(\mathrm{x}_1)p(\mathrm{x}_2|\mathrm{x}_1)p(\mathrm{x}_3|\mathrm{x}_1)p(\mathrm{x}_4|\mathrm{x}_1, \mathrm{x}_3)$.



between random variables to obtain a sparser representation. These dependencies are usually visualized using a directed graph. An important restriction which is often considered for computational reasons is to restrain from directed cycles. Such graphs are called *directed acyclic graph* (DAG) what entails that the nodes are ordered (Bishop, 2006, page 362ff.).

For instance, in order to represent the joint distribution $p(\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, \mathrm{x}_4)$ a matrix of the size M^4 is needed where the number of states M is assumed to be the equal for all random variables without loss of generality. A possible decomposition of the joint distribution can for example be given by

$$p(\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, \mathrm{x}_4) = p(\mathrm{x}_1)p(\mathrm{x}_2|\mathrm{x}_1)p(\mathrm{x}_3|\mathrm{x}_1)p(\mathrm{x}_4|\mathrm{x}_1, \mathrm{x}_3). \tag{2.9}$$

This alternative description of the joint distribution has a significantly reduced complexity of $M + M^2 + M^2 + M^3$. The graphical model which corresponds to Equation 2.9 is depicted in Figure 2.2.

Literature provides a number of algorithms to perform inference in graphs which differ in efficiency and exactness depending on the graph structures. Exact inference can be achieved using a message passing algorithm, called the sum-product algorithm, which is applicable to graphs having a tree-structure (Bishop, 2006, page 402ff.). Due to the structure, the probabilities can be derived in two passes in which the messages are propagated from one end of the network to the other end and back, i.e. forward and backward passes. The sum-product algorithm can be generalized to the so-called “loopy” belief propagation which can be applied to graphs with cycles. The algorithm ignores that messages may be propagated to parts of the network which were already visited by the message passing (Koller and Friedman, 2009).

Naïve Bayes classifier

A popular Bayesian network which will be used in the following sections to exemplify the differences between graphical models is the *naïve Bayes classifier* (NBC). The Bayes classifier decomposes the distribution $p(\mathbf{x}, \mathrm{y})$ into a likelihood $p(\mathbf{x}|y)$ and a prior over classes $p(y)$:

$$p(y, \mathbf{x}) = p(\mathbf{x}|y)p(y). \tag{2.10}$$

The likelihood describes the dependency of the observed feature vector $\mathbf{x}$ given the class

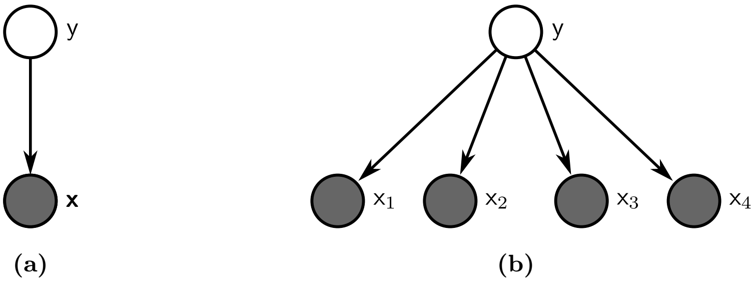


Figure 2.3. Graphical representation of the Bayes classifier and naïve Bayes classifier. (a) The Bayes classifier has only a single random variable modeling the observed feature vector $\mathbf{x}$ which is conditioned to the class random variable y. (b) The naïve Bayes classifier treats the elements of the feature vector $\mathbf{x}$ independently. The feature vector is composed of four elements, resulting in four random variables x_1, x_2, x_3 and x_4 where each depends on the class random variable y.

label random variable y. Figure 2.3a depicts the graphical model of the Bayes classifier in which observed variable is depicted by a filled circle. The observation vector of the random variable $\mathbf{x}$ is conditioned to the class label random variable y. However, this approach has the drawback that the modeling of the distribution $p(\mathbf{x}, \mathsf{y})$ requires the estimation of a large number of parameters. Especially, the observed feature vector increases the complexity of the model. Therefore, it is beneficial to use a more rigid decomposition of the distribution in case not enough samples are available for learning. This in turn leads to a more stable and robust classifier. Such a decomposition is performed by the NBC (Mitchell, 2005) which treats the elements of the observed feature vector as being independent. Hence, each element x_m with $m \in \{1, \ldots, M\}$ of the observation vector $\mathbf{x}$ is independently conditioned to the class label random variable y. The distribution of the NBC is given by

$$p(y, x_1, \ldots, x_M) = p(y) \prod_{m}^{M} p(x_m|y). \tag{2.11}$$

The graphical model of the NBC with $M = 4$ is shown in Figure 2.3b. The likelihood is maximized by optimizing the parameters of the model with respect to the data. Consider a dataset $\mathcal{T} = \{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ of discrete observations $\mathbf{x}^{(n)} = (x_1^{(n)}, \ldots, x_M^{(n)})$ and class labels $y^{(n)}$. Then the multinomial distribution of the random variable x_m conditioned to $\mathsf{y} = y$ is given by

$$p(\mathsf{x}_m = x_m|\mathsf{y} = y) = \mu_{m x_m y} \tag{2.12}$$

with the additional assumption $\sum_{x_m \in \mathsf{x}_m} \mu_{m x_m y} = 1$. The individual samples are treated as being i.i.d. such that the probability of the observations given a class is given by

$$p(\mathcal{T}_{ym}|\mathsf{y}=y) = \prod_{n=1}^{N_y} \mu_{mx_m^{(n)}y} \tag{2.13}$$

where the set $\mathcal{T}_{ym}$ contains only samples of the random variable x_m associated with the label $\mathsf{y} = y$ which are indexed by $n = 1, \dots, N_y$. In order to determine the maximum likelihood of the parameters, the objective function is created by taking the logarithm of Equation 2.13. The logarithmic function has no effect on the optimization process since it is monotone. Furthermore, the constraint $\sum_{x\in\mathsf{x}_m} \mu_{mxy} = 1$ is added using the Lagrange multiplier λ:

$$L(\boldsymbol{\mu}) = \sum_{n=1}^{N}\sum_{m=1}^{M} \ln \mu_{mx_m^{(n)}y^{(n)}} + \sum_{\hat{y}\in\mathsf{y}}\sum_{m=1}^{M} \lambda_{ym}\Big(\sum_{\hat{x}\in\mathsf{x}_m} \mu_{m\hat{x}\hat{y}} - 1\Big). \tag{2.14}$$

The maximum is determined by taking the derivative of the function and setting it equal to zero. In the second step the equation has to be rearranged to make the parameter the subject:

$$\frac{\partial L(\boldsymbol{\mu})}{\partial \mu_{mx_my}} = \underbrace{\sum_{n=1}^{N} \delta_{x_m^{(n)}=x_m}\delta_{y^{(n)}=y}\frac{1}{\mu_{mx_m^{(n)}y^{(n)}}}}_{N_{mx_my}/\mu_{mx_my}} + \lambda_{ym} = 0 \tag{2.15}$$

$$\Leftrightarrow \qquad \mu_{mx_my} = -\frac{N_{mx_my}}{\lambda}. \tag{2.16}$$

The Lagrange multiplier is determined by substituting the outcome back into Equation 2.15 which results in $\lambda_{ym} = -N_{ym}$. The final parameter is then given by

$$\mu_{mx_my} = \frac{N_{yx_mm}}{N_y}. \tag{2.17}$$

Classification is performed by evaluating the likelihoods $p(\hat{x}_m|y)$ for all possible labels y. The joint distribution is then obtained by multiplying the outcome with the corresponding prior $p(y)$. The prior can either be learned from the data, e.g. $\gamma_y = p(\mathsf{y} = y) = N_y/N$, or set manually.

2.1.4. Markov Networks

Markov networks which are also often referred to as *Markov random fields* (MRF) describe joint distributions using cliques of random variables. The random variables contained in a clique are assumed to be correlated. Without loss of generality, the joint distribution can be regarded as a factorization of functions based on the variables of maximal cliques (Bishop, 2006, page 385ff.)

$$p(\boldsymbol{\mathsf{x}}=\mathbf{x}) = \frac{1}{Z}\prod_{c\in\mathcal{C}}\phi_c(\mathbf{x}_c) \tag{2.18}$$

Potential function

where $\mathcal{C}$ denotes the set of cliques, $\mathbf{x}_c$ the states of the random variables being in the clique c, $\phi_c(\cdot)$ the *potential function* for the clique c, and Z the normalization constant required to ensure that the probabilities sum (or integrate) up to one. The quantity of Z which is also often referred to the *partitioning function* is given by

Partition function

$$Z = \sum_{\mathbf{x}\in\boldsymbol{\mathsf{x}}}\prod_{c\in\mathcal{C}}\phi_c(\mathbf{x}_c). \tag{2.19}$$

Since the potential function has to be strictly positive, the most popular representation of the function is based on the exponential function:

$$\phi_c(\mathbf{x}_c) = \exp\big(-E(\mathbf{x}_c)\big) \tag{2.20}$$

Feature function

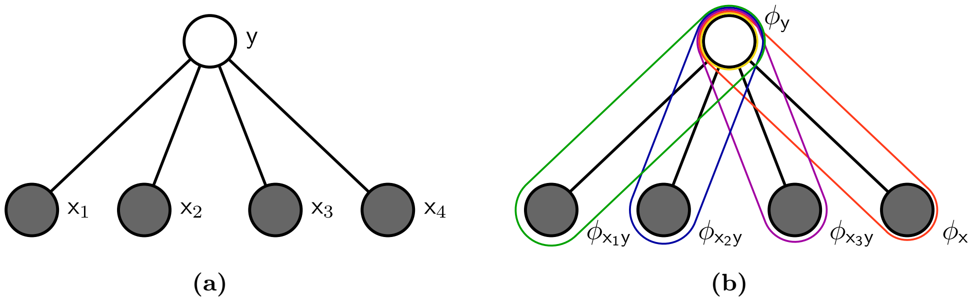
where $-E(\cdot)$ can be regarded as an energy function (Bishop, 2006, page 387ff.). A different notation is based on so-called *feature functions* which are defined for every possible state of the random variables (Sutton and McCallum, 2007; Richardson and Domingos, 2006). A feature function $f_{\hat{\mathbf{x}}}(\mathbf{x})$ for a particular state configuration $\hat{\mathbf{x}}$ returns a real-value. However, often an additional restriction to binary feature function is made such that the function returns one in case the state configuration is met and zero otherwise. The *log-linear model* utilizing feature functions is given by

Log-linear model

$$p(\boldsymbol{\mathsf{x}}=\mathbf{x}) = \frac{1}{Z}\exp\Bigg(\sum_{c\in\mathcal{C}}\sum_{\hat{\mathbf{x}}\in c} w_{\hat{\mathbf{x}}} f_{\hat{\mathbf{x}}}(\mathbf{x})\Bigg). \tag{2.21}$$

where $w_{\hat{\mathbf{x}}} \in \mathbb{R}_{\geq 0}$ weights the respective functions and is defined for each state configuration $\hat{\mathbf{x}}$. The sum over $\hat{\mathbf{x}} \in c$ iterates over all possible configurations of the random variables contained in the clique c. There are infinite parameter weighting settings which describe the same distribution, since the parameters can take any real value. Hence, a regularization term is often added to a model in order to constrain the parameter space. The log-linear model utilizing feature functions is well-suited for the definition of *template models*.

The graphical representation of Markov networks is given by nodes corresponding to random variables and links connecting pairs of nodes according to the feature functions (Bishop, 2006, page 383ff.; Koller and Friedman, 2009, page 103ff.). However, in contrast to Bayesian networks the links are undirected and describe a correlation between the connected ran-



(a) **(b)**

Figure 2.4. Naïve Bayes classifier interpreted as a Markov network. (a) Classic representation of Markov network. (b) Markov network with highlighted cliques.

dom variables. Figure 2.4a shows the NBC introduced in the previous section in form of a Markov network. Since the Bayesian network representation is more restrictive than the Markov network representation, the transformation of the NBC into a Markov network is quite straightforward and the number of parameters remains the identical. The corresponding log-linear model describing the NBC is given by

$$
\begin{aligned}
p(y, x_1, \dots, x_m) &= \frac{1}{Z}\phi_{\mathsf{y}}(y) \prod_{m=1}^{M} \phi_{\mathsf{x}_m\mathsf{y}}(x_m, y) && (2.22)\\
&= \frac{1}{Z} \exp\big(\sum_{l \in \mathsf{y}} \tilde{\gamma}_l f_y(l)\big) \prod_{m=1}^{M} \exp\big(\sum_{l \in \mathsf{y}} \sum_{s \in \mathsf{x}_m} \tilde{\mu}_{msl} f_{x_m,y}(s, l)\big) && (2.23)\\
&= \frac{1}{Z} \exp\big(\tilde{\gamma}_y + \sum_{m=1}^{M} \tilde{\mu}_{m x_m y}\big). && (2.24)
\end{aligned}
$$

The links representing the cliques ϕ_y and $\phi_{\mathsf{x}_m\mathsf{y}}(x_m, y)$ of Equation 2.22 are highlighted in Figure 2.4b. Equation 2.23 shows the log-linear model using the feature functions where $\tilde{\gamma}_y$ and $\tilde{\mu}_{m x_m y}$ are parameters to weight the configuration of a clique of random variables. The feature functions can be omitted because of the clear structure of the network. Equation 2.24 shows the representation without feature functions which will be used in the following.

In order to derive the optimal parameters for $\tilde{\gamma}_y$ and $\tilde{\mu}_{m x_m y}$ an additional constraint is added which restricts Z to be one. As a result, the representation of the probability distribution can be simplified such that it takes a similar form to Equation 2.14:

$$
\begin{aligned}
p(y, x_1, \dots, x_m) &= \underbrace{\frac{1}{\sum_{\hat{\mathbf{x}} \in \mathbf{x}} \sum_{\hat{y} \in \mathsf{y}} \prod_{m=1}^{M} \mu_{m \hat{x}_m \hat{y}} \cdot \gamma_{\hat{y}}}}_{1/Z=1/1} \cdot \exp\big(\ln \gamma_y + \sum_{m=1}^{M} \ln \mu_{m x_m y}\big) && (2.25)\\
&= \gamma_y \cdot \prod_{m=1}^{M} \mu_{m x_m y} && (2.26)
\end{aligned}
$$

where the denominator in Equation 2.25 can be interpreted as the normalization constant

Z which is set equal to one. In the next step, the logarithmic function is applied and additional constraints are added using the Lagrange multipliers λ and κ such that the objective function is given by

$$L(\boldsymbol{\gamma}, \boldsymbol{\mu}) = \sum_{n=1}^{N} \ln \gamma_{y^{(n)}} + \sum_{m=1}^{M} \ln \mu_{m x_m^{(n)} y^{(n)}} +$$

$$\sum_{\hat{y} \in \mathsf{y}} \sum_{m=1}^{M} \lambda_{ym} \Big(\sum_{\hat{x} \in \mathsf{x}_m} \mu_{m \hat{x} \hat{y}} - 1 \Big) + \kappa \Big(\sum_{\hat{y} \in \mathsf{y}} \gamma_{\hat{y}} - 1 \Big). \tag{2.27}$$

The parameters are determined analogously as described in the previous section.

Conditional random field

While graphical networks are generally used to perform inference or parameter learning on joint distributions, Lafferty et al. popularized in 2001 the so-called *conditional random field* (CRF). Again, a good showcase to explain the CRF is the NBC (Sutton and McCallum, 2007). The idea is to condition the labels to the observations by replacing the normalization constant Z using the function $Z(x_1, \ldots, x_m)$. The new function depends on the observations such that the distribution is written as

$$p(\mathsf{y}=y|x_1, \ldots, x_m) = \frac{1}{Z(x_1, \ldots, x_m)} \exp\left(\tilde{\gamma}_y + \sum_{m=1}^{M} \tilde{\mu}_{m x_m y} \right). \tag{2.28}$$

The normalization function is obtained by marginalizing over the labels y:

$$Z(x_1, \ldots, x_m) = \sum_{\hat{y} \in \mathsf{y}} \exp\left(\tilde{\gamma}_{\hat{y}} + \sum_{m=1}^{M} \tilde{\mu}_{m x_m y} \right). \tag{2.29}$$

In general, the conditioning can be performed similarly to Bayesian networks. However, the application of Markov networks in classifiers experiences a renaissance due to CRF.

Markov networks have many advantages compared to Bayesian networks: They are defined by cliques of correlating random variables which are easier to be identified than the causal dependencies of Bayesian networks. Furthermore, the restriction of the Bayesian network to acyclic graphs is no longer given. Hence, Markov networks can be defined using any structure including loops. However, the softening of the constraints comes at the expense of general parameter learning and inference which might no longer be exact and have an increased computational cost.

Markov property

The name of the Markov network originates from the so-called *Markov property* which relates to the fact that random variables are influenced by a limited number of other random variables, i.e. the random variables being in the same clique or, more general, in the Markov blanket (Freno and Trentin, 2011; Bishop, 2006, page 383ff). The MRF is a popular choice in image processing where each pixel is often modeled by a random variable being connected to adjacent pixel random variables (Hassner and Sklansky, 1980; Geman and Geman, 1984; Diebel and Thrun, 2006). However, the Markov property is not limited to Markov networks. For example, in temporal template models such as the HMM, the state of a random variable at time step $t+1$ is influenced by the distribution over the random variable of time step t.

2.1.5. Factor Graphs

As already mentioned, the classical representation of the Markov network is disadvantageous because the decomposition of a clique is ambiguous. A more qualified representation is the *factor graph* (Bishop, 2006, page 399ff.; Sutton and McCallum, 2007 ;Koller and Friedman, 2009, page 123ff.) which is a bi-partite graph where one type of nodes denote variables and a second type of nodes represent factors. In the following, variables are depicted by circular nodes and factors by squared nodes. The edges connecting the random variables with the factors indicate which variables are included in the factor. Figure 2.5 shows the NBC as a factor graph. The alternative representation explicitly depicts the factors $\phi_{\mathsf{x}_1\mathsf{y}}$, $\phi_{\mathsf{x}_2\mathsf{y}}$, $\phi_{\mathsf{x}_3\mathsf{y}}$, $\phi_{\mathsf{x}_4\mathsf{y}}$ and ϕ_{y} by the squared nodes. This information could not be deduced from a classic Markov network representation which was already shown in Figure 2.4.

Factor graphs can also be utilized to represent Bayesian networks. Figure 2.6a shows the Bayesian graphical model of the joint distribution $p(\mathsf{x}_1, \mathsf{x}_2, \mathsf{x}_3) = p(\mathsf{x}_1)p(\mathsf{x}_2)p(\mathsf{x}_3|\mathsf{x}_1, \mathsf{x}_2)$. The corresponding factor graph is shown in Figure 2.6b. As to be seen, a graphical Bayesian network can be unambiguously transformed into factor graph.

Figure 2.7a shows the Markov network which derived by so-called "marrying" the Bayesian directed links of Figure 2.6a (Bishop, 2006, page 390ff.). However, the resulting Markov network cannot be unambiguously transformed to a unique factor graph. Besides the true underlying decomposition shown in Figure 2.6b, Figure 2.7b and Figure 2.7c show two further valid factorizations. Figure 2.7b shows the factorization $\phi_1(\mathsf{x}_1, \mathsf{x}_2)\phi_2(\mathsf{x}_2, \mathsf{x}_3)\phi_3(\mathsf{x}_3, \mathsf{x}_1)$, whereas the factorization of Figure 2.7c is given by $\phi_1(\mathsf{x}_1)\phi_2(\mathsf{x}_2, \mathsf{x}_3)\phi_3(\mathsf{x}_1, \mathsf{x}_2, \mathsf{x}_3)$. All three interpretations of the Markov network are valid. Hence, the factor graph representation is

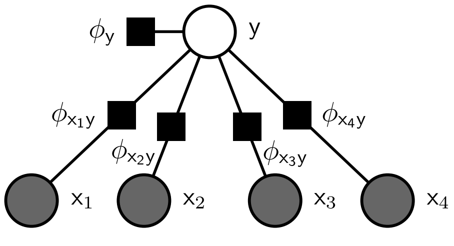

ϕ_{y}
y
$\phi_{\mathsf{x}_1\mathsf{y}}$
$\phi_{\mathsf{x}_4\mathsf{y}}$
$\phi_{\mathsf{x}_2\mathsf{y}}$
$\phi_{\mathsf{x}_3\mathsf{y}}$
x_1
x_2
x_3
x_4

Figure 2.5. Naïve Bayes classifier depicted as a factor graph. Random variables are depicted by circular nodes, whereas the squared nodes represent factors, or in case of Markov networks the cliques. The links connecting the random variables with factors indicate, which variables are included in the factor.

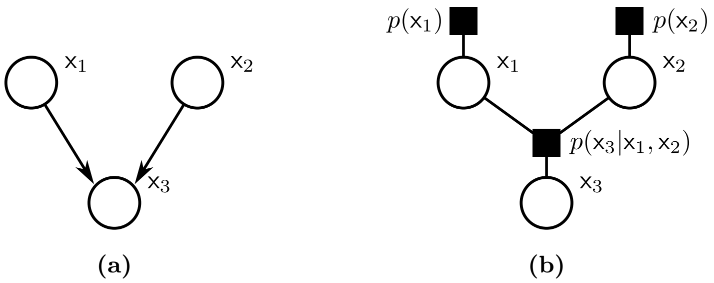


Figure 2.6. Conversion from a Bayesian network to a factor graph. The example shows how the transformation of the Bayesian network results in a unique factor graph.

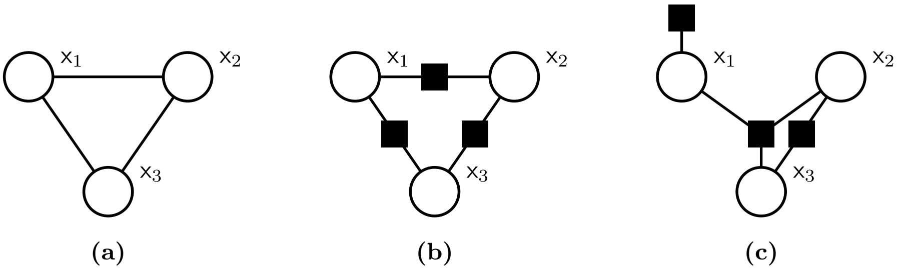


Figure 2.7. Conversion from a Markov network to a factor graph. The Markov network (a) can lead to different representations. Both factor graphs (b) and (c) are valid interpretations of the Markov network (a).

preferable, because it reveals the true structure of the Markov network. However, additional constraints, e.g. weight regularization, are still not being represented by the factor graph.

2.2. Gaussian Mixture Model

So far, the introduced formalizations were based on discrete random variables. However, in many applications it is necessary to use random variables which are defined over real-valued states (Bishop, 2006, page 78–126; Parzen, 1962). One of the most popular models for a random variable with real-valued states is the Gaussian distribution (Gauß, 1809). The distribution is described by a mean value and a standard deviation:

$$\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{\det(\boldsymbol{\Sigma})^{1/2}} \exp\Big(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu}) \Big) \tag{2.30}$$

where $\boldsymbol{\Sigma}$ denotes the covariance matrix, $\boldsymbol{\mu}$ the mean and D the dimensionality of the random variable $\mathbf{x}$. Although the Gaussian distribution turned out to be a fundamental model for myriad of observations in nature, the expressiveness of the density is still limited.

Alternative approaches are kernel density and nearest-neighbor estimators (Bishop, 2006, page 120–127), or the *Gaussian mixture model* (GMM) which makes use of a superposition of Gaussians. The GMM is frequently utilized in this thesis because of its advantageous properties. The GMM parameter learning is a well-known probabilistic formalism and, hence, used in various other approaches of this work. Furthermore, the GMM can be extended to realize more complex densities than real-valued density estimation. The basic idea is that a discrete state k of a latent random variable k selects a Gaussian distribution. Hence, the probability is described by

$$p(\mathbf{x}|\mathsf{k}=k)=\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k). \tag{2.31}$$

The probability of the latent random variable itself which can be regarded as a prior is given by $p(\mathsf{k}=k \mid \nu_k)$ where the variable which follows the short vertical bar denotes the parameter defining the probability. The density of an observation $\mathbf{x}$ is given by the marginal distribution of the conditional and prior probability:

$$\mathcal{G}(\mathbf{x} \mid \theta)= \sum_{k=1}^{K} p(\mathsf{k}=k)p(\mathbf{x}|\mathsf{k}=k) \tag{2.32}$$

$$= \sum_{k=1}^{K} \nu_k \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{2.33}$$

where K denotes the number of mixture components, $\theta = \{\nu_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$ denotes the set containing the GMM parameters. and the weights ν_k satisfy the condition $\sum_{k=1}^{K} \nu_k = 1$, $\nu_k > 0 \; \forall k = 1, \ldots, K$. As already mentioned, the GMM is able to describe densities which are more complex than the densities of a single Gaussian model. However, with respect to parameter learning, such GMM do not have a unique solution in contrast to a single Gaussian model. That means that the fitting of parameters can lead to different outcomes. The GMM parameter are determined using the well-known *expectation-maximization* (EM) *algorithm* (Dempster et al., 1977) in which each iteration is composed of two steps, i.e. the expectation (E-step) and the maximization (M-step). The algorithm is given by the following listing (Bishop, 2006, page 435ff.):

EM-algorithm

Input: A dataset $\mathcal{T} = \{\mathbf{x}^{(n)} \; : \; \mathbf{x}^{(n)} \in \mathbb{R}^D, n = 1, \ldots, N\}$ and the number of mixture components K.

1. Initialize $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$ and ν_k, e.g. by using k-means clustering and compute the log-likelihood of the model:

$$\ln \; p(\{\mathbf{x}^{(n)}\}_{n=1}^{N} \mid \theta) = \sum_{n=1}^{N} \ln \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k). \tag{2.34}$$

$$= LL(\theta) \tag{2.35}$$

2. (E-step) Estimate the conditional probability for each mixture component k to be responsible for the observation $\mathbf{x}^{(n)}$:

$$p(\mathsf{k} = k|\mathbf{x}^{(n)}) = \frac{\nu_k \mathcal{N}(\mathbf{x}^{(n)} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \nu_j \mathcal{N}(\mathbf{x}^{(n)} \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} = \gamma^{(n)}(k). \tag{2.36}$$

3. (M-step) Adapt the means and the covariance matrices of the Gaussians and the mixing parameters to maximize the likelihood:

$$\nu_k^{\text{new}} = \frac{N_k}{N}, \tag{2.37}$$

$$\boldsymbol{\mu}_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma^{(n)}(k) \mathbf{x}^{(n)} \tag{2.38}$$

$$\boldsymbol{\Sigma}_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma^{(n)}(k) (\mathbf{x}^{(n)} - \boldsymbol{\mu}_k)(\mathbf{x}^{(n)} - \boldsymbol{\mu}_k)^{\mathrm{T}} \tag{2.39}$$

where $N_k = \sum_{n=1}^{N} p(\mathsf{k} = k|\mathbf{x}^{(n)}) = \sum_{n=1}^{N} \gamma^{(n)}(k)$.

4. Update the parameters. Calculate the log-likelihood and check for stopping criteria, e.g. maximal number of iterations or a threshold based on the relative change of the last two log-likelihoods. In case stopping criteria were met, prepare the output. Otherwise return to step 2.

Output: The set of GMM parameter $\theta = \{\nu_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$.
The density of a new data sample $\hat{\mathbf{x}}$ is computed by evaluating:
$\mathcal{G}(\hat{\mathbf{x}}, \theta) = \sum_{k=1}^{K} \nu_k \mathcal{N}(\hat{\mathbf{x}} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ where $\theta = \{\{\nu_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}\}$.

The EM algorithm for GMM is parameterized by the number of mixture components, the shape of the covariance matrices, e.g. full or diagonal, the number of iterations and the stopping criteria. A configuration which is too restrictive leads to a density function which does not appropriately represents the underlying data. Conversely, choosing parameters which allow to many degrees of freedom results in convergence problems. The algorithm is sensitive to singularities, i.e. a mixture component gets responsible for a single observation, which leads to an infinitesimal variance and a likelihood diverging to infinity. Such singularities can be avoided by resetting the afflicted Gaussian using a heuristic or using a regularization term on the variance (Bishop, 2006; Ormoneit and Tresp, 1996). Hence, model parameters should be defined by experts or by model selection. However, the latter has the obvious drawback that most of the models will get rejected although they potentially render a good approximation of the underlying density.

2.3. Hybrid Joint Distribution

In the previous sections, joint distributions were based on either discrete or continuous random variables, or on joint distributions with continuous variables in which the discrete random variables were latent, i.e. the variables were not observed. The modeling of joint distributions in which discrete and continuous random variables are both observed will be addressed in this section. In this work, the joint distribution of observed discrete and continuous random variables, to which we will refer to as *hybrid joint distribution*, is modeled using a GMM extended by an additional discrete random variable.

The sample $\mathbf{x}$ is given by a tuple $\mathbf{x} = (\mathring{x}, \tilde{\mathbf{x}})$ where $\mathring{x}$ represents a discrete observation of a finite set and $\tilde{\mathbf{x}}$ an observed continuous vector of real numbers. The density of the continuous variable is modeled using a GMM which is selected based on the discrete state $\mathring{x}$:

$$
\begin{aligned}
p(\mathsf{x} = \mathbf{x} \mid \theta) &= p(\mathring{\mathsf{x}} = \mathring{x}, \tilde{\mathsf{x}} = \tilde{\mathbf{x}} \mid \theta) && (2.40)\\
&= \sum_{k=1}^{K} \nu_{\mathring{x}k} \mathcal{N}(\tilde{\mathbf{x}} \mid \boldsymbol{\mu}_{\mathring{x}k}, \boldsymbol{\Sigma}_{\mathring{x}k}) && (2.41)
\end{aligned}
$$

where the parameters of the model are $\theta = \{\nu_{\mathring{x}k}, \boldsymbol{\mu}_{\mathring{x}k}, \boldsymbol{\Sigma}_{\mathring{x}k}\}_{k=1,\mathring{x}\in\mathring{\mathsf{X}}}^{K}$. The parameters are maximized using the EM-algorithm by first determining the responsibilities $p(\mathsf{k} = k|\mathbf{x}^{(n)})$ of a given dataset $\mathcal{T} = \{\mathbf{x}^{(n)}\}_{n=1}^{N}$. For i.i.d. data samples, the function to be maximized is given by

$$
\ln p(\mathcal{T} \mid \theta) = \sum_{n=1}^{N} \ln \sum_{k=1}^{K} \nu_{\mathring{x}^{(n)}k} \mathcal{N}(\tilde{\mathbf{x}} \mid \boldsymbol{\mu}_{\mathring{x}^{(n)}k}, \boldsymbol{\Sigma}_{\mathring{x}^{(n)}k}). \tag{2.42}
$$

In order to determine the maximum likelihood of the mixing parameter $\nu_{\mathring{x}k}$ with $k = 1, \dots, K$ a Lagrange multiplier is added which ensures that the parameters will sum up to one. Taking the derivative and setting the outcome equal zero results in

$$
\begin{aligned}
\frac{\partial \ln p(\mathcal{T} \mid \theta)}{\partial \nu_{\mathring{x}l}} &= \frac{\partial}{\partial \nu_{\mathring{x}l}} \sum_{n=1}^{N} \delta_{\mathring{x}^{(n)}=\mathring{x}} \ln \sum_{k=1}^{K} \nu_{\mathring{x}k} \mathcal{N}(\tilde{\mathbf{x}}^{(n)} \mid \boldsymbol{\mu}_{\mathring{x}k}, \boldsymbol{\Sigma}_{\mathring{x}k}) - \kappa \left(\sum_{k=1}^{K} \nu_{\mathring{x}k} - 1 \right) && (2.43)\\
&= \sum_{n=1}^{N} \delta_{\mathring{x}^{(n)}=\mathring{x}} \underbrace{\frac{\nu_{\mathring{x}l} \mathcal{N}(\tilde{\mathbf{x}}^{(n)} \mid \boldsymbol{\mu}_{\mathring{x}l}, \boldsymbol{\Sigma}_{\mathring{x}l})}{\sum_{k=1}^{K} \nu_{\mathring{x}k} \mathcal{N}(\tilde{\mathbf{x}}^{(n)} \mid \boldsymbol{\mu}_{\mathring{x}k}, \boldsymbol{\Sigma}_{\mathring{x}k})}}_{\gamma^{(n)}(l)} - \kappa = \mathbf{0}. && (2.44)
\end{aligned}
$$

Analogously to the EM-algorithm for the GMM, the new variable $\gamma^{(n)}(l)$ describes the responsibility of a Gaussian for the l^{th} mixing component and the n^{th} data sample. The update formula of the weight $\nu_{\mathring{x}l}$ is derived by first determining the Lagrange multiplier κ. The outcome is then substituted back into the original equation and rearranged to obtain $\nu_{\mathring{x}l}$ which results in

$$
\nu_{\mathring{x}l} = \frac{\sum_{n=1}^{N} \delta_{\mathring{x}^{(n)}=\mathring{x}} \gamma^{(n)}(l)}{\sum_{n=1}^{N} \sum_{k=1}^{K} \delta_{\mathring{x}^{(n)}=\mathring{x}} \gamma^{(n)}(k)}. \tag{2.45}
$$

The update formula of the mean $\boldsymbol{\mu}_{\mathring{x}l}$ is obtained similarly by following transformations:

$$\frac{\partial \ln p(\mathcal{T} \mid \theta)}{\partial \boldsymbol{\mu}_{\mathring{x}l}} = \frac{\partial}{\partial \boldsymbol{\mu}_{\mathring{x}l}} \sum_{n=1}^{N} \delta_{\mathring{x}^{(n)}=\mathring{x}} \ln \sum_{k=1}^{K} \nu_{\mathring{x}k} \mathcal{N}(\tilde{\mathbf{x}}^{(n)} \mid \boldsymbol{\mu}_{\mathring{x}k}, \boldsymbol{\Sigma}_{\mathring{x}k}) \tag{2.46}$$

$$= \sum_{n=1}^{N} \delta_{\mathring{x}^{(n)}=\mathring{x}} \underbrace{\frac{\nu_{\mathring{x}l} \mathcal{N}(\tilde{\mathbf{x}}^{(n)} \mid \boldsymbol{\mu}_{\mathring{x}l}, \boldsymbol{\Sigma}_{\mathring{x}l})}{\sum_{k=1}^{K} \nu_{\mathring{x}k} \mathcal{N}(\tilde{\mathbf{x}}^{(n)} \mid \boldsymbol{\mu}_{\mathring{x}k}, \boldsymbol{\Sigma}_{\mathring{x}k})}}_{\gamma^{(n)}(l)} \boldsymbol{\Sigma}_{\mathring{x}l}^{-1} (\tilde{\mathbf{x}}^{(n)} - \boldsymbol{\mu}_{\mathring{x}l}) \tag{2.47}$$

$$= \sum_{n=1}^{N} \delta_{\mathring{x}^{(n)}=\mathring{x}} \gamma^{(n)}(l) \boldsymbol{\Sigma}_{\mathring{x}k}^{-1} (\tilde{\mathbf{x}}^{(n)} - \boldsymbol{\mu}_{\mathring{x}k}) = \mathbf{0} \tag{2.48}$$

$$\Leftrightarrow \qquad \boldsymbol{\mu}_{\mathring{x}l} = \frac{\sum_{n=1}^{N} \delta_{\mathring{x}^{(n)}=\mathring{x}} \gamma^{(n)}(l) \tilde{\mathbf{x}}^{(n)}}{N_{\mathring{x}l}} \tag{2.49}$$

where $N_{\mathring{x}l} = \sum_{n=1}^{N} \delta_{\mathring{x}^{(n)}=\mathring{x}} \gamma^{(n)}(l)$. Finally, the update formula of the covariance matrix $\boldsymbol{\Sigma}_{\mathring{x}l}$ is derived by

$$\boldsymbol{\Sigma}_{\mathring{x}l} = \frac{\sum_{n=1}^{N} \delta_{\mathring{x}_t^{(n)}=\mathring{x}} \gamma^{(n)}(l) (\tilde{\mathbf{x}}^{(n)} - \boldsymbol{\mu}_{\mathring{x}l})(\tilde{\mathbf{x}}^{(n)} - \boldsymbol{\mu}_{\mathring{x}l})^{\mathrm{T}}}{N_{\mathring{x}l}}. \tag{2.50}$$

It is important to emphasize that a single discrete random variable as described here can encode an arbitrary number of discrete random variables using the Cartesian product. An alternative model for observed hybrid random variables is the so-called random forest which was proposed by Breiman (2001).

2.4. Graph Probability Density

So far, the distributions presented were limited to an ordered set of discrete or continuous variables. However, there are particular applications in which this representation is insufficient and distributions of more complex structures are required (Cook and Holder, 2006). A prominent example of such structures are graphs.

Let $G = (\mathcal{V}, \mathcal{E})$ be a graph with a set of vertices $\mathcal{V}$ defined over a fixed universe and a set of edges $\mathcal{E}$. An edge is given by a tuple of two vertices $\mathbf{e}_m = (v_m, w_m)$ where $v_m, w_m \in \mathcal{V}$ and m indexes all edges in the graph $\mathcal{E} = \{\mathbf{e}_1 \dots \mathbf{e}_M\}$. To augment graphs with weights, additional functions are defined. Let $\mathbf{f}_{\mathcal{V}}(\cdot)$ be a function providing a vector of weights attached to a vertex and $\mathbf{f}_{\mathcal{E}}(\cdot, \cdot)$ be a function rendering a vector of weights attached to an edge.

Graphs have a flexible structure which makes it challenging to extract relevant and frequent features. Especially the large variability of graphs complicates the search of an optimal model which avoids underfitting and overfitting. Trivial approaches transform graphs into a fixed-length feature vector, e.g. by counting vertices and edges or by determining the maximal fan-in and fan-out. Classification of graph can then be performed using a standard classifier. A clear drawback of this approach is that the extraction of fixed-length feature vectors will give only a faint excerpt of the actual graph structure. A more sophisticated approach aims at finding characteristic sub-patterns which appear frequently in a set of graphs by using a classifier being inspired by the Apriori algorithm which was proposed by Inokuchi et al. (2003). Gärtner et al. (2007) proposed the use of kernel methods to map a graph to a real-valued vector. They suggested two graph kernels, namely the walk kernel (WK) and the cyclic pattern kernel (CPK). The first kernel maps directed graphs to sequences of vertices along edges which correspond to walks in graphs. Since no restrictions are made regarding loops, the length of the walks can be infinite such that approximations may be necessary, e.g. to a short walk. For two undirected graphs, the CPK is defined as the cardinality of the intersections of the graphs' cyclic and tree patterns. An alternative kernel which extends marginalized kernels based on random walks from sequences to graph has been proposed by Kashima et al. (2003). Besides the attempt to merely classify graphs, an alternative approach is to learn a probability distribution for a set of graphs. The estimation of a graph probability density is a challenging since the structure of graphs has to be parameterized in many different dimensions, e.g. number of vertices and edges, fan-in and fan-out, weights on nodes or edges, number of cliques and so forth. Classic approaches make use of a fixed-length intermediate representation such that a standard density estimator can be applied, e.g. a GMM (Schmidt and Schwenker, 2011). However, Trentin and Di Iorio (2009, 2007a,b) proposed a novel concept called *graph probability density* (GPD) which shifts the focus from generative models to predictive models, i.e. discriminating classifiers. In the following, this new approach will be presented in detail.

The distribution $p(\mathsf{y}|G)$ provides the probability for a class random variable y given an observed graph G. According to the Bayes' theorem, the probability of y given a graph G can be derived using the likelihood and the priors

$$p(\mathsf{y} = y|G) = \frac{p(G|\mathsf{y} = y \shortmid \lambda_y)p(\mathsf{y} = y)}{p(G)} \tag{2.51}$$

where λ_y is the set of model parameters describing the distribution of graphs of class y. The key idea of the new approach is to consider the edges as being i.i.d. such that the graph gets decomposed into its main elements. In other words, each individual vertex depends only on its neighboring vertices which corresponds to an implicit Markov assumption in the vertices space. Such assumptions are frequently used in probabilistic graphical models and several other statistical machine learning approaches.

The GPD is defined by $p(G|\mathsf{y} \shortmid \lambda_\mathsf{y}) = p(\mathsf{e}_1, \ldots, \mathsf{e}_M|\mathsf{y} \shortmid \lambda_\mathsf{y}) = \prod_{m=1}^{M} p(\mathsf{e}_m|\mathsf{y} \shortmid \lambda_\mathsf{y})$ where $p(\mathsf{e}_m = \mathbf{e}_m|\mathsf{y} \shortmid \lambda_\mathsf{y})$ denotes the probability of the edge $\mathbf{e}_m \in \mathcal{E}$ given the model of λ_y. In case the graph is composed of weighted and labeled vertices and edges, an edge is described by a high-dimensional vector $\mathbf{x}_m = (v_m, w_m, \mathbf{f}_\mathcal{V}(v_m), \mathbf{f}_\mathcal{V}(w_m), \mathbf{f}_\mathcal{E}(v_m, w_m))$ with discrete and continuous elements. This vector can be modeled by the hybrid joint distributions which were presented in the previous Section 2.3. Inserting the likelihood using the independence assumption $p(G|\mathsf{y} \mid \lambda_{\mathsf{y}}) = \prod_{m=1}^{M} p(\mathbf{x}_m|\mathsf{y} \mid \lambda_{\mathsf{y}})$ into the Equation 2.51 results in

$$p(\mathsf{y}|\bigcap_{m=1}^{M} \mathbf{x}_m) = \frac{\prod_{m=1}^{M} p(\mathbf{x}_m|\mathsf{y} \mid \lambda_{\mathsf{y}})p(\mathsf{y})}{p(\bigcap_{m=1}^{M} \mathbf{x}_m)}. \tag{2.52}$$

The new formalization can handle arbitrary-sized graphs indifferent of whether they are cyclic or acyclic and directed or undirected.

Hence, the input to GPD is given by a set of tuples $\mathbf{x} = \{(\mathring{x}_m, \tilde{\mathbf{x}}_m)\}_{m=1}^{M}$ where each tuple corresponds to a hybrid joint distribution. The discrete state $\mathring{x}_m$ identifies a unique edge with the help of the labeled vertices. The vector $\tilde{\mathbf{x}}_m \in \mathbb{R}^D$ represents the weights associated to the edge and the corresponding vertices. Without lose of generality, the quantity M indexes the fix number of edges in the actual graph. The density over the set $\mathbf{x}$ is then defined by:

$$\begin{aligned} p(\mathbf{x} \mid \lambda) &= \prod_{m=1}^{M} p(\mathring{x}_m, \tilde{\mathbf{x}}_m \mid \theta_m) && (2.53) \\ &= \prod_{m=1}^{M} \sum_{k=1}^{K} \nu_{\mathring{x}_m k} \mathcal{N}(\tilde{\mathbf{x}}_m \mid \boldsymbol{\mu}_{\mathring{x}_m k}, \boldsymbol{\Sigma}_{\mathring{x}_m k}) && (2.54) \end{aligned}$$

where m indexes the discrete edge identifier $\mathring{x}_m$. In order to derive the parameters $\lambda = \{\theta_m\}_{m=1}^{M}$ for a given dataset $\{\mathbf{x}^{(n)}\}_{n=1}^{N}$, the log-likelihood of the distribution

$$\ln p(\{\mathbf{x}^{(n)}\}_{n=1}^{N} \mid \lambda) = \sum_{n=1}^{N} \ln \prod_{m=1}^{M} p(\mathring{x}_m^{(n)}, \tilde{\mathbf{x}}_m^{(n)} \mid \theta_m) \tag{2.55}$$

needs to be maximized. This is achieved by determining the derivative of the target parameter and setting the outcome equal to zero:

$$\begin{aligned} \frac{\partial \ln p(\{\mathbf{x}^{(n)}\}_{n=1}^{N} \mid \lambda)}{\partial \theta_m} &= \frac{\partial}{\partial \theta_m} \sum_{n=1}^{N} \ln \prod_{m=1}^{M} p(\mathring{x}_m^{(n)}, \tilde{\mathbf{x}}_m^{(n)} \mid \theta_m) && (2.56) \\ &= \sum_{n=1}^{N} \sum_{m=1}^{M} \frac{\partial}{\partial \theta_m} \ln p(\mathring{x}_m^{(n)}, \tilde{\mathbf{x}}_m^{(n)} \mid \theta_m) = 0. && (2.57) \end{aligned}$$

The update formula of the mixing component $\nu_{\mathring{x}l}$ is derived by introducing a term containing the Lagrange multiplier κ which ensures that all components of an edge sum up to one:

$$\begin{aligned} \frac{\partial \ln p(\{\mathbf{x}^{(n)}\}_{n=1}^{N} \mid \lambda)}{\partial \nu_{\mathring{x}l}} = \frac{\partial}{\partial \nu_{\mathring{x}l}} & \sum_{n=1}^{N} \sum_{m=1}^{M} \delta_{\mathring{x}_m^{(n)} = \mathring{x}} \ln \sum_{k=1}^{K} \nu_{\mathring{x}k} \mathcal{N}(\tilde{\mathbf{x}}_m^{(n)} \mid \boldsymbol{\mu}_{\mathring{x}k}, \boldsymbol{\Sigma}_{\mathring{x}k}) - \\ & \kappa\Bigg(\sum_{k=1}^{K} \nu_{\mathring{x}k} - 1\Bigg) && (2.58) \end{aligned}$$

$$= \sum_{n=1}^{N} \sum_{m=1}^{M} \delta_{\mathring{x}_m^{(n)}=\mathring{x}} \underbrace{\frac{\nu_{\mathring{x}l}\mathcal{N}(\tilde{\mathbf{x}}_m^{(n)} \mid \boldsymbol{\mu}_{\mathring{x}l}, \boldsymbol{\Sigma}_{\mathring{x}l})}{\sum_{k=1}^{K} \nu_{\mathring{x}k}\mathcal{N}(\tilde{\mathbf{x}}_m^{(n)} \mid \boldsymbol{\mu}_{\mathring{x}k}, \boldsymbol{\Sigma}_{\mathring{x}k})}}_{\widetilde{\gamma}^{(n)}(\mathring{x},l)} - \kappa = 0 \tag{2.59}$$

where $\delta_{\mathring{x}_m^{(n)}=\mathring{x}}$ is one in case the equation $\mathring{x}_m^{(n)} = \mathring{x}$ holds true and zero otherwise. The new variable $\widetilde{\gamma}^{(n)}(\mathring{x}, l)$ describes the responsibility of the indexed Gaussian to have generated the edge encoded by $\mathring{x}$ and the mixture component l. Re-arranging the formula to κ and substituting the result back into the Equation 2.59 gives

$$\nu_{\mathring{x}l}^{\text{new}} = \frac{\sum_{n=1}^{N} \sum_{m=1}^{M} \delta_{\mathring{x}_m^{(n)}=\mathring{x}} \widetilde{\gamma}^{(n)}(\mathring{x}, l)\nu_{\mathring{x}l}}{\sum_{n=1}^{N} \sum_{m=1}^{M} \sum_{k=1}^{K} \delta_{\mathring{x}_m^{(n)}=\mathring{x}} \widetilde{\gamma}^{(n)}(\mathring{x}, k)\nu_{\mathring{x}k}}. \tag{2.60}$$

Analogously, the update formulas of the mean and covariance matrix are given by

$$\boldsymbol{\mu}_{\mathring{x}l}^{\text{new}} = \frac{\sum_{n=1}^{N} \sum_{m=1}^{M} \delta_{\mathring{x}_m^{(n)}=\mathring{x}} \widetilde{\gamma}^{(n)}(\mathring{x}, l)\tilde{\mathbf{x}}_m^{(n)}}{\sum_{n=1}^{N} \sum_{m=1}^{M} \delta_{\mathring{x}_m^{(n)}=\mathring{x}} \widetilde{\gamma}^{(n)}(\mathring{x}, l)} \tag{2.61}$$

$$\boldsymbol{\Sigma}_{\mathring{x}l}^{\text{new}} = \frac{\sum_{n=1}^{N} \sum_{m=1}^{M} \delta_{\mathring{x}_m^{(n)}=\mathring{x}} \widetilde{\gamma}^{(n)}(\mathring{x}, l)(\tilde{\mathbf{x}}_m^{(n)} - \boldsymbol{\mu}_{\mathring{x}l})(\tilde{\mathbf{x}}_m^{(n)} - \boldsymbol{\mu}_{\mathring{x}l})^{\mathrm{T}}}{\sum_{n=1}^{N} \sum_{m=1}^{M} \delta_{\mathring{x}_m^{(n)}=\mathring{x}} \widetilde{\gamma}^{(n)}(\mathring{x}, l)}. \tag{2.62}$$

The proposed GPD has many advantageous characteristics: Due to the Markov property, the density is applicable to variable-sized graphs. Furthermore, the approach can be applied to directed and undirected graphs which can further have labeled or weighted vertices or edges. Conversely, the GPD can be used only for discriminative purposes and is restricted to graphs which are small enough to ensure numerical stable estimations. Each possible edge and labeling has to be present in the database to guarantee that a proper density function is learned.

2.5. Hidden Markov Model

Up to this point, probability densities of fixed-length and variable-length random variables have been discussed. In this section, another characteristic by which patterns can be discriminated shall be addressed: their manifestation over time. A popular approach to model sequences of observations is the *hidden Markov model* (HMM) (Koller and Friedman, 2009; Rabiner, 1989).

The HMM bases on the assumption that an observed sequence $\mathbf{X} \in \mathbb{R}^{D\times T}$ was generated by a hidden sequence of discrete states $\mathbf{w} \in \Omega_{\mathsf{w}}^{T}$ where D is the length of the observed features, T is the number of discrete time steps of the sequence, and Ω_{w}^{T} denotes the set of all possible sequences of the random variables $(\mathsf{w}_1, \mathsf{w}_2, \ldots, \mathsf{w}_T) = \mathbf{w}$. While the observations $\mathbf{x}_t$ are treated as being independent from each other given a hidden state, the hidden random

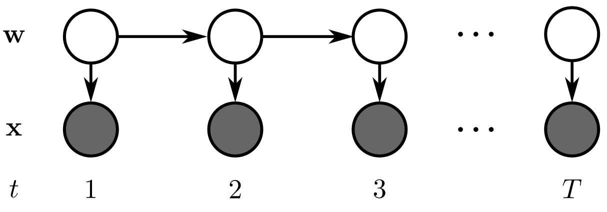


Figure 2.8. Graphical representation of the HMM. The Markov chain w is composed of a sequence of hidden random variables w_t. Each hidden random variable w_t, except the one of first first time step, depends on its predecessor w_{t-1}. Furthermore, each hidden random variable emits an observation $\mathbf{x}_t$.

variables w capture the sequential dependencies by a Markov chain $p(\mathsf{w}_t = w_t|\mathsf{w}_{t-1} = w_{t-1})$. Each observation $\mathbf{x}_t$ is emitted by the corresponding hidden state w_t according to the probability distribution $p(\mathbf{x}_t|\mathsf{w}_t = w_t \shortmid \theta_{w_t})$. The graphical model representing the HMM decomposition of the probability distribution is shown in Figure 2.8. The probability of a sequence $\mathbf{X}$ for an HMM λ is given by

$$
\begin{aligned}
p(\mathbf{X} \shortmid \lambda) &= \sum_{\mathbf{w} \in \Omega_{\mathsf{w}}^T} p(\mathbf{X}, \mathbf{w} \shortmid \lambda) && (2.63)\\
&= \sum_{\mathbf{w} \in \Omega_{\mathsf{w}}^T} p(\mathsf{w}_1 = w_1 \shortmid \boldsymbol{\pi}) \prod_{t=2}^{T} p(\mathsf{w}_t = w_t|\mathsf{w}_{t-1} = w_{t-1} \shortmid \mathbf{A}) \cdot \\
&\quad \prod_{t=1}^{T} p(\mathbf{x}_t|\mathsf{w}_t = w_t \shortmid \boldsymbol{\theta}) && (2.64)
\end{aligned}
$$

where $\lambda = \{\mathbf{A}, \boldsymbol{\pi}, \boldsymbol{\theta}\}$ denotes the set of model parameters. The probability is divided into three main components: the initial probability, the transition probability and the emission probability. The initial probability $p(\mathsf{w}_1 = w_1 \shortmid \boldsymbol{\pi})$ specifies the probability of an initial hidden state and is defined by $\boldsymbol{\pi}$ with a number of $|\Omega_{\mathsf{w}}|$ parameters. The Change from one hidden state of time step $t-1$ to another hidden state of the time step t is described by the transition probability $p(\mathsf{w}_t = w_t|\mathsf{w}_{t-1} = w_{t-1} \shortmid \mathbf{A})$ specified by $\mathbf{A}$ with a number of $|\Omega_{\mathsf{w}}| \times |\Omega_{\mathsf{w}}|$ parameters. To reduce the number of parameters, the transition matrix is often restricted to the so-called *left-to-right* matrix in which the hidden states are ordered and only transitions to the succeeding or the same states are permitted while transitions to preceding nodes are not allowed. The model parameters of the observations are given by $\boldsymbol{\theta}$. In the following, parameters are omitted whenever not explicitly needed for the sake of clarity. The HMM can be seen as one of the simplest nontrivial examples of a dynamic Bayesian network created according to a *template model* principle (Koller and Friedman, 2009, page 203). For each time step, a hidden random variable is instantiated such that the unrolled model compiles enough time slices to match the number of observations.

Template model

Three problems for HMM

Rabiner and Schafer (1978) prominently defined "*three basic problems for HMM*":

1. *Given an HMM λ: How likely has a sequence of observations $\mathbf{X}$ been generated by λ?*
   This probability can be determined using the Equation 2.64 which marginalizes over all possible combinations of the Markov chain w. This computationally expensive procedure can be circumvented by using the forward algorithm which is based on dynamic programming. The forward algorithm will be introduced in Section 2.5.1.

2. *Given a set of sequences $\mathcal{T} = \{\mathbf{X}^{(n)}\}_{n=1}^{N}$: What are the parameters having the highest likelihood?* The problem is generally referred to as parameter learning and can be solved by the EM algorithm (Dempster et al., 1977). The EM algorithm is based on the forward and backward algorithms with an additional maximization step and will be presented in Section 2.5.2.

3. *Given an HMM λ and a sequence $\mathbf{X}$: What is the most likely sequence of hidden states?* Different solutions were proposed depending on the definition of "most likely sequence". The Viterbi algorithm is probably the most popular solution to this question. Since this question has only marginal relevance for this thesis, the interested reader is referred to (Bishop, 2006, page 629).

Classification using HMM is performed by assigning a class y to a sequence of observations $\mathbf{X}$. This is realized in the same manner as for the NBC introduced in Section 2.1.3. In the first step, an HMM λ_{y} is learned for each class y using a corresponding set of observed sequences. A new data sample $\hat{\mathbf{X}}$ is then tested by evaluating $p(\hat{\mathbf{X}} \mid \lambda_{\mathsf{y}})$ for all classes. The probability distribution given a new data sample is then obtained by

$$p(\mathsf{y} = y|\hat{\mathbf{X}}) = \frac{p(\hat{\mathbf{X}} \mid \lambda_y)p(\mathsf{y} = y)}{\sum_{\hat{y}\in\Omega_{\mathsf{y}}} p(\hat{\mathbf{X}} \mid \lambda_{\hat{y}})p(\mathsf{y} = \hat{y})} \tag{2.65}$$

where the likelihood $p(\hat{\mathbf{X}} \mid \lambda_y)$ is determined with the help of inference and the class prior $p(\mathsf{y})$ is obtained from data or generated by hand (Bishop, 2006, page 38ff.).

In the following, we discuss the algorithms to perform inference and parameter learning.

2.5.1. Inference

Forward algorithm

To infer the likelihood that a sequence $\mathbf{X}$ was generated by the HMM λ it is necessary to marginalize over all possible sequences of hidden states. At the first glance, this marginalization appears to be computationally expensive. However, by utilizing dynamic programming the complexity can be reduced to $O(T \cdot |\Omega_{\mathsf{w}}|^2)$. For each time step t and hidden state j the so-called forward variable is evaluated (Bishop, 2006, page 411ff. and 618ff.):

$$\alpha_t(j) = p(\mathbf{X}_{1..t}, \mathsf{w}_t = j) \tag{2.66}$$

where $\mathbf{X}_{1..t}$ denotes the selection of observations starting from the first time step to the time step t. The recursive formulas using dynamic programming are given by

$$\alpha_t(j) = p(\mathbf{x}_t|\mathsf{w}_t = j) \sum_{i\in\Omega_{\mathsf{w}}} \alpha_{t-1}(i)p(\mathsf{w}_t = j|\mathsf{w}_{t-1} = i) \tag{2.67}$$

$$\alpha_1(j) = p(\mathbf{x}_1|\mathsf{w}_1 = j)p(\mathsf{w}_1 = j). \tag{2.68}$$

Hence, the probability of a sequence $\mathbf{X}$ given an HMM λ can be derived by performing a sum over the forward variable of the last time step:

$$p(\mathbf{X} \mid \lambda) = \sum_{i \in \Omega_{\mathsf{w}}} \alpha_T(i). \tag{2.69}$$

The learning of parameters which will be addressed in the next section requires two additional probability distributions, i.e. the responsibility $\gamma_t(j) = p(\mathsf{w}_t = j|\mathbf{X})$ of hidden state j at time step t given the complete sequence $\mathbf{X}$, and the probability $\xi_{t-1,t}(i,j) = p(\mathsf{w}_{t-1} = i, \mathsf{w}_t = j|\mathbf{X})$ of the transition from state i in time step $t-1$ to the hidden state j in time step t given the complete sequence $\mathbf{X}$. In order to determine these probabilities, the backward variables $\beta_t(j) = p(\mathbf{X}_{t+1..T}|\mathsf{w}_t = j)$ are required which are derived similarly to the forward variable:

Backward algorithm

$$\beta_T(j) = 1 \tag{2.70}$$

$$\beta_t(j) = \sum_{i \in \Omega_{\mathsf{w}}} \beta_{t+1}(i) p(\mathbf{x}_{t+1}|\mathsf{w}_{t+1} = i) p(\mathsf{w}_{t+1} = i|\mathsf{w}_t = j) \tag{2.71}$$

$$\beta_0(\cdot) = \sum_{i \in \Omega_{\mathsf{w}}} \beta_1(i) p(\mathbf{x}_1|\mathsf{w}_1 = i) p(\mathsf{w}_1 = i). \tag{2.72}$$

The responsibility of hidden state j at time step t given the observations is obtained by

$$\gamma_t(j) = p(\mathsf{w}_t = j|\mathbf{X}) = \frac{p(\mathbf{X}|\mathsf{w}_t = j)p(\mathsf{w}_t = j)}{p(\mathbf{X})} \tag{2.73}$$

$$= \frac{p(\mathbf{X}_{1..t}, w_t = j)p(\mathbf{X}_{t+1..T}|\mathsf{w}_t = j)}{p(\mathbf{X})} \tag{2.74}$$

$$= \frac{\alpha_t(j)\beta_t(j)}{p(\mathbf{X})} \tag{2.75}$$

and the probability of the transition from state i at time step $t-1$ to state j at time step t given the observations is obtained by

$$\xi_{t-1,t}(i,j) = p(\mathsf{w}_{t-1} = i, \mathsf{w}_t = j|\mathbf{X}) \tag{2.76}$$

$$= \frac{p(\mathbf{X}_{1..t-1}, w_{t-1} = i)p(\mathsf{w}_t = j|\mathsf{w}_{t-1} = i)}{p(\mathbf{X})}. \tag{2.77}$$

$$p(\mathbf{x}_t|\mathsf{w}_t = j)p(\mathbf{X}_{t+1..T}|\mathsf{w}_t = j)$$

$$= \frac{\alpha_{t-1}(i)p(\mathsf{w}_t = j|\mathsf{w}_{t-1} = i)p(\mathbf{x}_t|\mathsf{w}_t = j)\beta_t(j)}{p(\mathbf{X})}. \tag{2.78}$$

The forward-backward algorithm tends to be numerically unstable for long sequences of $\mathbf{X}$ because of probabilities which take values below the machine precision. Therefore, the variables are commonly normalized using scaling factors $p(\mathbf{X}_{1..t}) = \sum_{j \in \mathsf{w}_t} \alpha_t(j)$ for each time step:

$$\widehat{\alpha}_t(j) = p(\mathsf{w}_t = j|\mathbf{X}_{1..t}) = \frac{\alpha_t(j)}{p(\mathbf{X}_{1..t})}. \tag{2.79}$$

Furthermore, it is advisable to perform the calculations in logarithmic space. A detailed description of the normalization can be found in (Bishop, 2006, page 627ff.).

2.5.2. Parameter Learning

The parameters of the HMM λ given a set of sequences $\mathcal{T} = \{\mathbf{X}^{(n)}\}_{n=1}^{N}$ can be derived using the EM-algorithm (Rabiner, 1989; Dempster et al., 1977). The posterior distributions of the latent variables $p(\mathbf{w}|\mathbf{X} \shortmid \lambda^{\text{old}})$ are determined in the E-step as already described in the previous section. The M-step then maximizes the parameters in λ with respect to the expectation of the complete-data log-likelihood

$$Q(\lambda, \lambda^{\text{old}}) = \sum_{n=1}^{N} \sum_{\mathbf{w} \in \Omega_{\mathsf{w}}} p(\mathbf{w}|\mathbf{X}^{(n)} \shortmid \lambda^{\text{old}}) \ln p(\mathbf{X}^{(n)}, \mathbf{w} \shortmid \lambda). \tag{2.80}$$

The equation is expanded by replacing the joint distribution $p(\mathbf{X}^{(n)}, \mathbf{w}|\lambda)$ with the Equation 2.64 and the quantities $p(\mathbf{w}|\mathbf{X}^{(n)} \shortmid \lambda^{\text{old}})$ with $\gamma_t(i)$ and $\xi_{t-1,t}(i,j)$ of Equation 2.75 and 2.78. The likelihood is then given by

$$\begin{aligned} Q(\lambda, \lambda^{\text{old}}) = & \sum_{n=1}^{N} \sum_{w_1 \in \Omega_{\mathsf{w}}} \gamma_1^{(n)}(w_1) \ln \pi_{w_1} + \sum_{t=2}^{T} \sum_{w_{t-1} \in \Omega_{\mathsf{w}}} \sum_{w_t \in \Omega_{\mathsf{w}}} \xi_{t-1,t}^{(n)}(w_{t-1}, w_t) \ln a_{w_{t-1} w_t} + \\ & \sum_{t=1}^{T} \sum_{w_t \in \Omega_{\mathsf{w}}} \gamma_t^{(n)}(w_t) \ln p(\mathbf{x}_t^{(n)}|\mathsf{w}_t = w_t \shortmid \boldsymbol{\theta}) \qquad (2.81) \\ = & \sum_{n=1}^{N} Q^{(n)}(\lambda, \lambda^{\text{old}}) \qquad (2.82) \end{aligned}$$

where a_{w_{t-1},w_t} is an element of the matrix $\mathbf{A}$, π_{w_1} an element of the vector $\boldsymbol{\pi}$ and $\boldsymbol{\theta}$ holds the observation parameters. In the M-step, each holds arameters is maximized separately. To keep the notation uncluttered, the evaluation of the expected log-likelihood is first outlined only for a single data sample $\mathbf{X}^{(n)}$, i.e. $Q^{(n)}(\lambda, \lambda^{\text{old}})$, and later expanded to the complete dataset.

In order to maximize the probability distribution of the initial hidden state $p(\mathsf{w}_1 = w_1 \shortmid \boldsymbol{\pi})$, an additional constraint $\sum_{i \in \Omega_{\mathsf{w}}} \pi_i = 1$ with the Lagrange multiplier κ is added. Taking the derivative with respect to π_{w_1} and setting the outcome equal to zero results in

$$\frac{\partial Q^{(n)}(\lambda, \lambda^{\text{old}})}{\partial \pi_{w_1}} = \frac{\partial}{\partial \pi_{w_1}} \sum_{i \in \Omega_{\mathsf{w}}} \gamma_1^{(n)}(i) \ln \pi_i - \kappa \Bigg(\sum_{i \in \Omega_{\mathsf{w}}} \pi_i - 1 \Bigg) = 0 \tag{2.83}$$

$$\Leftrightarrow \quad \frac{\gamma_1^{(n)}(w_1)}{\pi_{w_1}} - \kappa = 0 \tag{2.84}$$

$$\Leftrightarrow \quad \gamma_1^{(n)}(w_1) - \pi_{w_1} \kappa = 0. \tag{2.85}$$

The quantity of the Lagrange multiplier κ is derived using the constraint $\sum_{i \in \Omega_{\mathsf{w}}} \pi_i = 1$:

$$\sum_{i\in\Omega_{\mathrm{w}}} \gamma_1^{(n)}(i) - \pi_i\kappa = 0 \tag{2.86}$$

$$\Leftrightarrow \qquad \sum_{i\in\Omega_{\mathrm{w}}} \kappa\pi_i = \sum_{i\in\Omega_{\mathrm{w}}} \gamma_1^{(n)}(i) \tag{2.87}$$

$$\Leftrightarrow \qquad \kappa = \sum_{i\in\Omega_{\mathrm{w}}} \gamma_1^{(n)}(i). \tag{2.88}$$

The update formula is obtained by substituting the outcome back into Equation 2.85:

$$\frac{\gamma_1^{(n)}(w_1)}{\pi_{w_1}} - \kappa = \frac{\gamma_1^{(n)}(w_1)}{\pi_{w_1}} - \sum_{i\in\Omega_{\mathrm{w}}} \gamma_1^{(n)}(i) = 0 \tag{2.89}$$

$$\Leftrightarrow \qquad \pi_{w_1} = \frac{\gamma_1^{(n)}(w_1)}{\sum_{i\in\Omega_{\mathrm{w}}} \gamma_1^{(n)}(i)}. \tag{2.90}$$

The final update formula is obtained by transferring the finding to the complete dataset. Hence, the initial parameter is maximized by

$$\pi_{w_1} = \sum_{n=1}^{N} \frac{\gamma_1^{(n)}(w_1)}{\sum_{i\in\Omega_{\mathrm{w}}} \gamma_1^{(n)}(i)}. \tag{2.91}$$

The update formula for elements of the transition matrix **A** is obtained by again adding a regularization term with a Lagrange multiplier κ and setting the derivative with respect to a_{vw} equal to zero:

$$\begin{aligned}\frac{\partial Q(\lambda,\lambda^{\mathrm{old}})^{(n)}}{\partial a_{vw}} &= \frac{\partial}{\partial a_{vw}} \sum_{t=2}^{T} \sum_{i\in\Omega_{\mathrm{w}}} \sum_{j\in\Omega_{\mathrm{w}}} \xi_{t-1,t}^{(n)}(i,j) \ln a_{ij} - \\ &\qquad \kappa \sum_{i\in\Omega_{\mathrm{w}}} \sum_{j\in\Omega_{\mathrm{w}}} a_{ij} - 1 = 0\end{aligned} \tag{2.92}$$

$$\Leftrightarrow \quad \sum_{t=2}^{T} \frac{\xi_{t-1,t}^{(n)}(v,w)}{a_{vw}} - \kappa = 0 \tag{2.93}$$

$$\Leftrightarrow \sum_{t=2}^{T} \xi_{t-1,t}^{(n)}(v,w) - a_{vw}\kappa = 0. \tag{2.94}$$

Again the update formula is obtained by first determining the quantity of the Lagrange multiplier and then substituting the outcome back into the original equation:

$$\sum_{t=2}^{T} \frac{\xi_{t-1,t}^{(n)}(v,w)}{a_{vw}} - \kappa = \sum_{t=2}^{T} \frac{\xi_{t-1,t}^{(n)}(v,w)}{a_{vw}} - \sum_{j\in\Omega_{\mathrm{w}}} \xi_{t-1,t}^{(n)}(v,j) = 0 \tag{2.95}$$

$$\Leftrightarrow \qquad a_{vw} = \frac{\sum_{t=2}^{T} \xi_{t-1,t}^{(n)}(v,w)}{\sum_{t=2}^{T} \sum_{j\in\Omega_{\mathrm{w}}} \xi_{t-1,t}^{(n)}(v,j)}. \tag{2.96}$$

The final update formula for the complete dataset is then given by

$$\Leftrightarrow \qquad a_{vw} = \frac{\sum_{n=1}^{N}\sum_{t=2}^{T}\xi_{t-1,t}^{(n)}(v,w)}{\sum_{n=1}^{N}\sum_{t=2}^{T}\sum_{j\in\Omega_{\mathsf{w}}}\xi_{t-1,t}^{(n)}(v,j)}. \tag{2.97}$$

The parameters of the observation probability $p(\mathbf{x}_t|\mathsf{w}_t = w_t \mid \boldsymbol{\theta})$ are derived by solving the derivative of the individual hidden state i

$$\frac{\partial Q^{(n)}(\lambda,\lambda^{\text{old}})}{\partial\theta_i} = \frac{\partial}{\partial\theta_i}\sum_{t=1}^{T}\sum_{j\in\mathsf{w}_t}\gamma_t^{(n)}(j)\ln p(\mathbf{x}_t|\mathsf{w}_t = j \mid \boldsymbol{\theta}) = 0. \tag{2.98}$$

In literature, multiple models for observations have been proposed. Discrete random variables are modeled using the multinomial distribution, whereas continuous random variables are often modeled using GMM. In the following, the integration of these probability distributions into the HMM will be presented.

2.5.3. Discrete Observed Variables

Sequences of observed discrete random variables are modeled using one *multinomial distribution* for each hidden state i. Each distribution is defined over a set of states Ω_{x} such that the probability distribution is given by

Multinomial distribution

$$p(\mathsf{x}_t = x|\mathsf{w}_t = i \mid \boldsymbol{\theta}) = \nu_{ix}. \tag{2.99}$$

The update formula is derived by introducing an additional constraint which ensures that the probabilities of the states sums up to one. Applying the logarithm and taking the derivative which is then set equal to zero leads to

$$\nu_{ix} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{T}\gamma_t^{(n)}(i)\delta_{x_t^{(n)}=x}}{\sum_{n=1}^{N}\sum_{t=1}^{T}\gamma_t^{(n)}(i)} \tag{2.100}$$

where $\delta_{x_t^{(n)}=x}$ is equal one in case $x_t^{(n)} = x$ holds true and zero otherwise.

2.5.4. Continuous Observed Variables

One of the most popular models for continuous observations in HMM is the GMM. Analogously to the multinomial distribution, one GMM is defined for each hidden state i:

$$p(\mathbf{x}_t = \mathbf{x}_t|\mathsf{w}_t = i \mid \boldsymbol{\theta}) = \sum_{k=1}^{K} z_{ik}\mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_{ik},\boldsymbol{\Sigma}_{ik}). \tag{2.101}$$

The maximization of the GMM parameters is performed similarly to the EM-algorithm for GMM presented in Section 2.2. However, in the current setting the HMM responsibility $\gamma_t^{(n)}(i)$ is insufficient. An additional responsibility $\widehat{\gamma}_t^{(n)}(i,k)$ is required which expresses the probability of the k^{th} Gaussian in the i^{th} GMM to have emitted the data sample $\mathbf{x}_t^{(n)}$. Again, the parameters are maximized by taking the derivative of Equation 2.85 with respect to the desired parameter and setting the outcome equal to zero. The update formula of the mixing component is given by

$$z_{ik} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{T}\gamma_t^{(n)}(i)\widehat{\gamma}_t^{(n)}(k)}{\sum_{n=1}^{N}\sum_{t=1}^{T}\sum_{l}^{K}\gamma_t^{(n)}(i)\widehat{\gamma}_t^{(n)}(l)} \tag{2.102}$$

and the update formulas of the mean and covariance matrix are given by

$$\boldsymbol{\mu}_{ik} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{T}\gamma_t^{(n)}(i)\widehat{\gamma}_t^{(n)}(i,k)\mathbf{x}_t^{(n)}}{\sum_{n=1}^{N}\sum_{t=1}^{T}\gamma_t^{(n)}(i)\widehat{\gamma}_t^{(n)}(i,k)} \tag{2.103}$$

and

$$\boldsymbol{\Sigma}_{ik} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{T}\gamma_t^{(n)}(i)\widehat{\gamma}_t^{(n)}(k)(\mathbf{x}_t^{(n)}-\boldsymbol{\mu}_{ik})(\mathbf{x}_t^{(n)}-\boldsymbol{\mu}_{ik})^{\mathrm{T}}}{\sum_{n=1}^{N}\sum_{t=1}^{T}\gamma_t^{(n)}(i)\widehat{\gamma}_t^{(n)}(k)}. \tag{2.104}$$

2.6. Latent-Dynamic Conditional Random Field

The plain structure of the HMM can directly be transfered into the Markov network representation. Morency et al. (2007) extended this representation and developed the *latent-dynamic conditional random field* (LDCRF) based on the concept of CRF introduced in Section 2.1.4. The LDCRF performs a mapping from the sequence of hidden states $\boldsymbol{\mathsf{w}}$ and observations $\mathbf{X}$ to a sequence of labels $\mathbf{y}$. In a first step, the conditioned probability distribution is defined by the decomposition:

$$p(\boldsymbol{\mathsf{y}}|\mathbf{X} \shortmid \lambda) = \sum_{\boldsymbol{\mathsf{w}} \in \boldsymbol{\mathsf{w}}} p(\boldsymbol{\mathsf{y}}|\boldsymbol{\mathsf{w}} = \mathbf{w}, \mathbf{X} \shortmid \lambda)p(\boldsymbol{\mathsf{w}} = \mathbf{w}|\mathbf{X} \shortmid \lambda) \tag{2.105}$$

where $\boldsymbol{\mathsf{w}}$ denotes the sequence of hidden states, $\mathbf{X}$ denotes the sequence of observations and $\boldsymbol{\mathsf{y}}$ denotes the sequence of labels.

Figure 2.9 shows the graphical model of the LDCRF. The observed variables are depicted by nodes colored in dark gray. They are given by the sequence $\mathbf{X}$. The nodes colored in lighter gray represent random variables which are observed in parameter learning but are unknown in case inference. They are given by the sequence of labels $\mathbf{y}$. The LDCRF model is restricted to a disjoint set of hidden states associated with each class label in order to keep parameter learning and inference tractable. Let $\Omega_{\boldsymbol{\mathsf{w}}}^{y}$ be the set of hidden states being

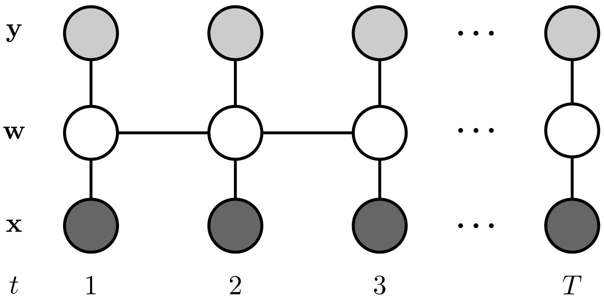


Figure 2.9. Graphical representation of the LDCRF. Analogously to the HMM, the hidden states are connected to a Markov chain $\mathsf{\mathbf{w}}$. Each hidden state w_t is further connected to an observation $\mathbf{x}_t$ and to a label state y_t.

associated with the class label y. As a result, the LDCRF model simplifies to

$$p(\mathsf{\mathbf{y}}|\mathbf{X} \shortmid \lambda) = \sum_{\mathbf{w} \in \mathsf{\mathbf{w}} \wedge w_t \in \Omega_{\mathsf{w}}^{y_t}} p(\mathsf{\mathbf{w}} = \mathbf{w}|\mathbf{X} \shortmid \lambda). \tag{2.106}$$

The probability distribution is given by the energy function:

$$\begin{aligned} p(\mathbf{w}|\mathbf{X}, \lambda) = \frac{1}{Z(\mathbf{X})} \exp \big(& \sum_{k=1}^{K} \theta_k \sum_t f_k(w_t, \mathbf{x}_t, t) + \\ & \sum_{l=1}^{L} \theta_{l+K} \sum_t g_l(w_{t-1}, w_t, t) \big) \end{aligned} \tag{2.107}$$

where the partition function depends on the sequence of observations and the set $\lambda = \{\boldsymbol{\theta}\}$ contains all model parameters. The first feature function $f_k(w_t, \mathbf{x}_t, t)$ is the so-called state function. If the observations are defined by a discrete random variable, the function is reduced to return binary values. However, if the observations are modeled using continuous random variables, the function has a real-valued range. The second feature function $g_l(w_{t-1}, w_t, t)$ is the transition function in which all combinations of the discrete states w_{t-1} and w_t are indexed by l. In order to perform parameter learning from a dataset, the following objective function needs to be minimized

$$E(\lambda) = \sum_{n}^{N} p(\mathbf{y}^{(n)}|\mathbf{X}^{(n)}) - \frac{1}{2\sigma^2}||\boldsymbol{\theta}||^2. \tag{2.108}$$

The last term of the function regularizes the growth of the parameters. Generally, gradient descent is performed in order to derive the optimal parameter setting (Morency et al., 2007; Sutton and McCallum, 2007). An algorithm comparable to the EM-algorithm can be used to perform inference since the structure of the LDCRF allows exact inference. Introducing labels which influence the states of the hidden random variables opens many interesting fields of applications. However, modeling the distribution with a Markov network limits the prospect to directly transfer achievements made in the field of Bayesian network HMM to the LDCRF. Furthermore, the model is significantly restricted by the fact that each class is assigned to a distinct set of hidden states.

2.7. Kalman Filter

The Kalman Filter (Kalman, 1960) is a popular algorithm to enhance the quality of noisy measurements over time. It finds application in the field of navigation and object tracking (Blackman and Popoli, 1999; Bar-Shalom and Li, 1993). Instead of calculating a moving average over the measurements, the Kalman filter models explicitly the measurement noise to weight the influence of measurements (Bishop, 2006, page 635ff.). The Kalman filter is based on a Markov chain which, in contrast to HMM, is realized by continuous latent variables. The model has the same tree-like structure as the HMM such that exact inference and parameter learning can also be performed efficiently. However, for our purposes it will be sufficient to discuss the inference and to refer the interested reader to the large range of literature, e.g. (Bar-Shalom and Li, 1993; Bishop, 2006; Koller and Friedman, 2009).

Inference is done by deriving the predicted state estimate $p(\mathbf{w}_{t+1}|\mathbf{X}_{1..t})$ based on the past observations $\mathbf{X}_{1..t}$ where $\mathbf{w}_{t+1}$ denotes a continuous multi-dimensional latent random variable. To obtain an efficient inference algorithm in which the functional form of the continuous variable $\mathbf{w}_t$ is preserved the Gaussian function is a tractable choice (Bishop, 2006, page 636). Hence, the probability distribution of the Markov chain and the emission probability are modeled by

$$p(\mathbf{w}_t|\mathbf{w}_{t-1}) = \mathcal{N}(\mathbf{w}_t \mid \mathbf{A}\mathbf{w}_{t-1}, \mathbf{Q}) \tag{2.109}$$

$$p(\mathbf{x}_t|\mathbf{w}_t) = \mathcal{N}(\mathbf{x}_t \mid \mathbf{H}\mathbf{w}_t, \mathbf{R}) \tag{2.110}$$

where $\mathbf{A}$ is the state transition model, $\mathbf{Q}$ the noise model, $\mathbf{H}$ the measurement matrix and $\mathbf{R}$ the error covariance. In order to predict an enhanced measurement, the latent state and the associated certainty for $p(\mathbf{x}_t|\mathbf{w}_t)$ have to be derived. The respective mean $\boldsymbol{\mu}_{t+1}$ and the covariance $\boldsymbol{\Sigma}_{t+1}$ are obtained in two steps using the current observation $\mathbf{x}_{t+1}$ and the last mean $\boldsymbol{\mu}_t$ and covariance $\boldsymbol{\Sigma}_t$. First, the mean $\hat{\boldsymbol{\mu}}_{t+1}$ and the covariance $\hat{\boldsymbol{\Sigma}}_{t+1}$ of the predicted state estimate are obtained by

$$\hat{\boldsymbol{\mu}}_{t+1} = \mathbf{A}\boldsymbol{\mu}_t + \mathbf{B}\mathbf{u} \tag{2.111}$$

$$\hat{\boldsymbol{\Sigma}}_{t+1} = \mathbf{A}\boldsymbol{\Sigma}_t\mathbf{A}^{\mathrm{T}} + \mathbf{Q} \tag{2.112}$$

where $\mathbf{B}$ is the control-input model and $\mathbf{u}$ is the control vector. The latent random variable must not necessarily have the same dimensionality as as the actual observations e.g. it cannot only model the velocity buy also the acceleration. In the second step the updated state distribution is derived by first computing a set of auxiliary variables:

$$\gamma = (\mathbf{x}_{t+1} - \mathbf{H}\hat{\boldsymbol{\mu}}_{t+1}) \tag{2.113}$$

$$\mathbf{S} = (\mathbf{H}\hat{\mathbf{\Sigma}}_{t+1}\mathbf{H}^{\mathrm{T}} + \mathbf{R}) \tag{2.114}$$

$$\mathbf{K}_{t+1} = \hat{\mathbf{\Sigma}}_{t+1}\mathbf{H}^{\mathrm{T}}\mathbf{S}^{-1} \tag{2.115}$$

where $\boldsymbol{\gamma}$ denotes the innovation and $\mathbf{S}$ the corresponding covariance matrix. The so-called Kalman gain $\mathbf{K}_{t+1}$ affects the impact of new measurements to the prediction. A low Kalman gain smoothes the measurements, whereas a high Kalman gain keeps close track to the measurements. The final mean and covariance matrix of the predicted state estimate are then determined with

$$\boldsymbol{\mu}_{t+1} = \hat{\boldsymbol{\mu}}_{t+1} + \mathbf{K}_{t+1}\boldsymbol{\gamma} \tag{2.116}$$

$$\mathbf{\Sigma}_{t+1} = (\mathbf{I} - \mathbf{K}_{t+1}\mathbf{H})\hat{\mathbf{\Sigma}}_{t+1}. \tag{2.117}$$

Within this work, the parameter learning for Kalman filter has only been addressed marginally. For further details on this topic, the interested reader is referred to the literature (Bar-Shalom and Li, 1993; Kalman, 1960).

2.8. Markov Logic Network

The recognition of patterns by analyzing the compliance to logical rules requires a close coupling of symbolic and sub-symbolic knowledge. It is advantageous to make use of a less rigid logic representation in order to handle real-world scenarios afflicted with uncertainty and to realize a smooth transition from pattern recognition to symbolic AI (Domingos, 2006). In literature, a couple of promising approaches to probabilistic logic were proposed (de Salvo Braz et al., 2008; De Raedt, 2008; Pasula and Russell, 2001). Among these approaches, the Markov logic network (MLN) which combines probability and first-order logic has attracted considerable attention (Richardson and Domingos, 2006). The MLN can been regarded as a template model for constructing Markov networks based on first-order logic. According to Richardson and Domingos (2006), a first-order knowledge base can be regarded as a set of hard constraints on the set of possible worlds. In case a world violates even one formula, it has a probability of zero. The idea of MLN is to soften these constraints: When a world violates one formula in the knowledge base it is less probable, but not impossible. An MLN $M_{\mathcal{L},\mathcal{C}}$ is defined by a set of formulas F_i and their corresponding weights w_i, i.e. $\mathcal{L} = \{(F_1, w_1), (F_2, w_2), \ldots, (F_{|\mathcal{L}|}, w_{|\mathcal{L}|})\}$, and a set of constants $\mathcal{C} = \{c_1, c_2, \ldots, c_{|\mathcal{C}|}\}$. The sets $\mathcal{L}$ and $\mathcal{C}$ represent the template model to create the Markov network which is done according to the following two rules:

1. Create a binary random variable for each theoretical grounding of the predicates based on the constants in $\mathcal{C}$. In case the node is indeed grounded, the binary node equals one if the atom is *true* and zero otherwise.

2. Create a binary feature function for each formula F_i and all possible configuration of grounded predicates involved in the formula. The weight of the feature function is given by w_i.

An example of an MLN in the "Pet owner" domain is given in Table 2.1. In this table, the rows specify the weights, the first-order logic formulas, the descriptions in natural language and the function identifiers. Formula F_1 describes the strength by which a thing y is owned by x, or in other words how much something is owned. The second formula F_2 expresses the prior of being a pet in the world. Formula F_3 specifies that a pet is never an owner but is always owned. The last formula F_4 constitutes that owners are happy. The weight of formula F_3 is set to infinity in order to express that this proposition always holds. The remaining weights can only be interpreted with respect to each other in an instantiated network. However, the formula F_1 which is weighted by 2.0 has a higher probability than formula F_2 which is weighted by $-{}^1\!/_2$. Hence, it is less likely to be a pet than to be an owner.

The first rule to create the MLN generates all possible groundings of the predicates based on the constants $\mathcal{C} = \{\texttt{bob}, \texttt{cat}\}$, e.g. $\texttt{pet}$(bob), $\texttt{pet}$(cat) or $\texttt{own}$(cat,cat). The second rule creates the feature functions connecting these predicates according to the formulas, e.g. formula F_4 generates four feature functions: $\texttt{own}$(bob,cat) ⇒ $\texttt{ishappy}$(bob), $\texttt{own}$(bob,bob) ⇒ $\texttt{ishappy}$(bob), $\texttt{own}$(cat,bob) ⇒ $\texttt{ishappy}$(cat) and $\texttt{own}$(cat,cat) ⇒ $\texttt{ishappy}$(cat). Figure 2.10 shows the factor graph of the grounded Markov network given the two constants. A

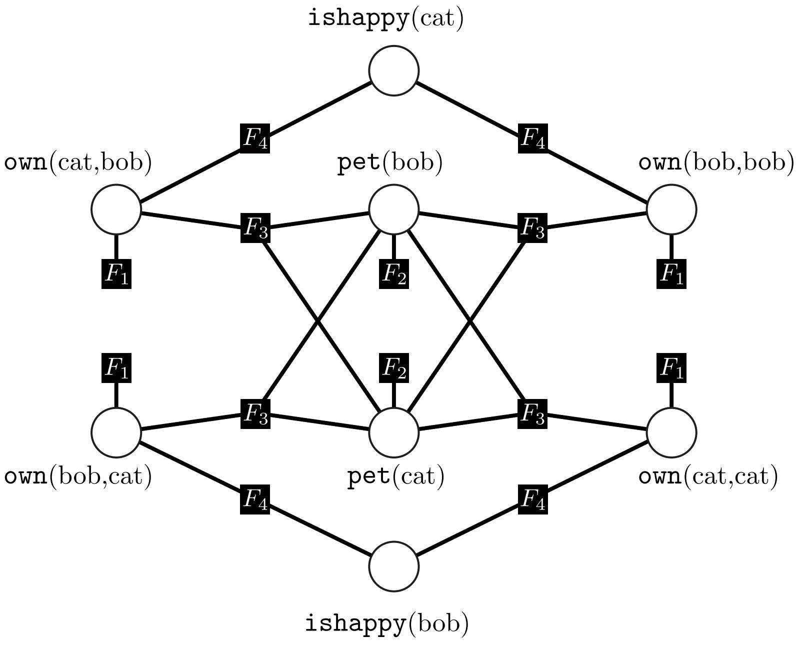


Figure 2.10. Grounded MLN of the "Pet owner" domain. The MLN is grounded according to the formulas and weights of Table 2.1 using the constants $\mathcal{C} = \{\texttt{bob}, \texttt{cat}\}$.

Table 2.1. MLN description of the "Pet owner" domain. A template model of an ungrounded MLN is given by weighted first-order logic formulas. Textual descriptions and indentifiers for formulas are additionally provided.

Weight	First-order logic	Description	ID
2.0	$\texttt{own}(x, y)$	Prior of owning things	F_1
$-{}^{1}/_{2}$	$\texttt{pet}(x)$	Prior of being a pet	F_2
∞	$\texttt{own}(x, y) \Rightarrow \neg\texttt{pet}(x) \wedge \texttt{pet}(y)$	Ownership is subjected to non-pets	F_3
${}^{4}/_{5}$	$\texttt{own}(x, y) \Rightarrow \texttt{ishappy}(x)$	To own things makes one happy	F_4

Table 2.2. MLN prediction on the predicates in the "Pet owner“ domain. The tables shows the probabilities assigned to all possible predicates in case the predicate `pet`(bob) has been set to *false*.

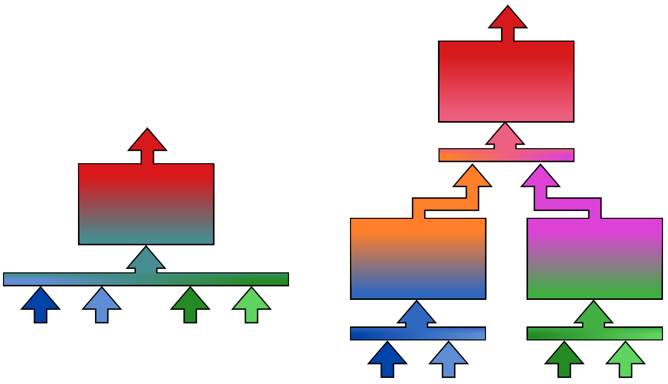
`ishappy`(bob)	73.79%	`ishappy`(cat)	50.01%
`own`(bob,bob)	0%	`own`(bob,cat)	61.67%
`own`(cat,bob)	0%	`own`(cat,cat)	0%
`pet`(cat)	76.26%		

query to the network is carried out by setting observed predicates to either *true* or *false* and subsequently performing the inference, e.g. using Gibbs sampling (Bishop, 2006), Table 2.2 shows the outcome of a query to the MLN in which the predicate `pet`(bob) has been set to *false*. The table shows that the predicate `pet`(cat) is *true* with a probability of 76.26% and that the cat is owned by bob with a probability of 61.67%. Furthermore, bob is likely to be happy with a probability of 73.79%, whereas the probability of the cat to be happy is undeceive with being close to 50%.

The dynamic Markov logic network (DMLN) extends the concept of the MLN by enrolling a template model over time, analogously to the HMM. Geier and Biundo (2011) proposed a new approximate online inference technique for the DMLN which is more computationally efficient. The DMLN updates only selected new nodes and reuses messages emitted by older nodes. The DMLN will be utilized later on in the empirical evaluations.

2.9. Multiple Classifier Systems

The creation of robust and accurate classifiers has always been in the focus of pattern recognition. Solutions range from sophisticated non-linear discriminant functions to heuristics to solve non-separable classifier problems, but also include large systems based on multiple classifiers, so-called multiple classifier systems (MCS) (Kuncheva, 2004; Kittler et al., 1998). MCS have been designed for a large number of purposes, other than the objectives robustness and performance. A common task is the combination of recognition outcomes from different modalities, e.g. audio and video channels, or the fusion of recognition outcomes over time (Dietrich, 2004). Other MCS solve multi-class problems or make use of hierarchical or layered architectures to recognize complex classes. Combiners in MCS



(a) Early fusion (b) Mid-level fusion (c) Late fusion

Figure 2.11. Early, mid-level and late fusion schemes utilized in MCS architectures.

can be categorized by the time at which information from multiple sources are brought together. A distinction is made between early, mid-level and late fusion approaches (Valstar et al., 2007; Sanderson and Paliwal, 2004). An overview of these three concepts is depicted in Figure 2.11.

Early fusion

Early fusion describes the procedure to concatenate multiple input feature vectors to a single high-dimensional feature vector. The assumption is that the combined input feature vectors are dependent to each other and that a classifier operating on the concatenated feature vector can learn correlations between them. The early fusion is illustrated in Figure 2.11a. Two input feature vectors which are given by the blue and green arrows are combined to a new single feature vector which is represented by the mixed colored thin block. The resulting new feature vector is then fed into a classifier depicted by the larger block. The classifier maps the concatenated feature vector to the final output depicted by the red arrow. Early fusion is promising for low-dimensional features which can complement each other and which, taken by themselves, are not discriminative enough to solve the recognition problem. However, the drawback of this approach is the danger of running into the so-called *curse of dimensionality.* in which the number of data samples is to low with respect to their dimensionality. As a result, the space of possible separating classifier functions can contain trivial solutions which will not generalize (Bishop, 2006; Vapnik, 1999; Schels et al., 2013a). We will not regard the early fusion as a full member of MCS. However, this kind of information fusion is frequently part of large MSC architectures.

Late fusion

In contrast to early fusion, the idea of *late fusion* is to create multiple individual classifiers for the identical classes which are then combined using a fusion rule, e.g. voting, averaging or the product rule (Kittler et al., 1998; Thiel, 2010, page 133ff.; Giacinto and Roli, 2002; Schels et al., 2013c). Advanced classifier fusion rules are realized by learning a mapping to the final class, e.g. using a linear associative memory, decision template, pseudo inverse or NBC (Schels et al., 2010; Schwenker et al., 2010, 2006). The late fusion is illustrated

in Figure 2.11c. Two features which are depicted by the blue and green arrows are used independently to classify the same red-colored target class. The classifier outcomes are then combined to a final decision by some fusion rule. The objective is to increase the accuracy and robustness by combining classifiers outputs which are *diverse*, i.e. the classifiers outputs do not agree on falsely classified samples (Kuncheva, 2004).
Classifier diversity can be achieved in many ways, e.g. different feature views, features from different modalities, bagging, different base classifier or varying the model parameters (Kuncheva, 2004; Breiman, 1996). However, it is still mandatory for late fusion that the individual classifiers have an acceptable accuracy (Kuncheva, 2004). A set of classifiers which are trained with the objective to be diverse are often referred to as classifier ensemble. According to Dietterich (2000), there are three main reason why ensembles of classifiers have the ability to outperform a single model, i.e. "statistical", "computational" and "representational" reason. From a statistical perspective, it is safer to combine the outputs of an ensemble rather than selecting a single model (Kuncheva, 2004). This strategy reduces the risk of picking a single model which does not generalize. From a computational perspective, the aggregated model is useful in scenarios in which the optimization procedure leads to different local optima. Combining a set of models trained with different initial parameters bears a lower risk than picking an individual model. Furthermore, combined models offer a richer representation in terms of a more accurate classifier function than a single model.

Mid-level fusion

The third fusion technique realizes a compromise of both aforementioned fusion techniques. The idea is to create a *mid-level fusion* with the help of a layered (or hierarchical) compound of classifiers. In the first layer, the feature vectors are mapped to two sets of intermediate classes. The corresponding class outputs are then concatenated to form a new feature vector which is then processed by a classifier on the second layer. The classifier in the second layer performs the final mapping to the actual target class. Figure 2.11b depicts the mid-level fusion. The blue feature vector is mapped to an intermediate orange class, while the green feature vector is independently mapped to another intermediate class which is indicated by the color magenta. Both class outputs are then concatenated and used as input to a classifier in the second layer which performs the final mapping to the red target class. The idea is to realize a smooth transition to the final target class by using intermediate classifier outputs which gradually densifies the information. The intermediate classes have to be tailored to the information in the feature vectors, be of relevance for the final target class and have to complement each other. This concept contrasts to late fusion in which all classifiers of the ensemble aim at recognizing the identical classes. A mixture of late fusion and mid-level fusion is realized by *multi-class MCS*, i.e. one-vs-one or one-vs-rest classifier systems (Bishop, 2006, page 182ff.). A one-vs-one classifier system is based on a set of $I \cdot (I-1)/2$ classifiers where I denotes the number of classes. One classifier is trained

Multi-class MCS

for every pair of the $I \cdot (I-1)/2$ possible class combinations. The final class assignment is obtained by collecting and combining all classifier decisions, e.g. by performing voting. However, this approach can lead to regions in feature space in which no class assignment can be carried out (Bishop, 2006, page 182ff.). The problem can be solved by using base classifiers with probabilistic class memberships outputs. The classifier outputs can then be transfered to the final distribution over I classes using a trained classifier or a mapping algorithm, e.g. pairwise coupling (Hastie and Tibshirani, 1998). An alternative to the one-vs-one classifier is of a one-vs-rest classifier system in which only I base classifiers are required. In this setting, every class is trained once against the joined set of the remaining classes (Theodoridis and Koutroumbas, 2006, page 127ff.).

In both, mid-level fusion and late fusion the outputs of classifiers have to be combined in a way that as much information as possible is preserved. This can be achieved by using classifiers which provide additional outputs beside crisp classifier decisions (Thiel, 2010). An obvious choice is to use a measure which describes the uncertainty associated with a classifier decision. However, there is a large number of possible interpretations of uncertainty (Bloch et al., 2001) and it is not always clear how to determine and represent uncertainty. Two of the most frequently used models for uncertainty are: (1) class membership probability, i.e to which degree of certainty is a data sample a member of a class; and (2) classifier confidence, i.e. how certain is the classifier about the current decision (Thiel, 2010, page 42ff.). Class membership probabilities can be derived using probabilistic models or in case of discriminative approaches by assessing the distance to the separating hyperplane (Platt, 2000). Classifier confidences can be derived using the certainty of the class memberships: The closer the class membership probability gets to zero or one, the more confident is the classifier decision (Bishop, 2006, page 42ff.). An alternative classifier confidence is derived by measuring the agreement in a classifier ensemble with respect to class probability memberships (Kuncheva, 2004, page 295ff.; Chow, 1970), e.g. based on standard deviation. Confidences can be used to weight or to reject classifier decisions.

Temporal fusion

Many real-world scenarios, require that not only classifier outputs from multiple feature sets (or modalities) are combined, but also classifier outputs from different time instances (Jeon and Landgrebe, 1999; Box et al., 2008). In order to perform *temporal fusion*, the most recent classifier outputs have to be collected in form of a stream. The fusion can then be performed by shifting a window over the stream and combining the contained classifier decisions to a single classification output using standard late fusion approaches, e.g. voting or averaging (Meudt et al., 2013). Temporal fusion can significantly improve the accuracy by smoothing noisy classifier decisions over time. However, it creates also many challenges such as the handling of missing classifier decisions, e.g. due to sensor failures. Furthermore, several streams of multimodal classifier outputs may need to be combined.

This raises further questions such as how to handle different sample rates or what is the optimal fusion strategy: multimodal followed by temporal fusion or temporal and followed by multimodal fusion (Schels et al., 2014; Dietrich et al., 2003; Dietrich, 2004).

2.10. Layered Hidden Markov Model

Section 1.2.3 introduced a number of classifier architectures which are able to recognize complex classes based on elementary cues. The *layered hidden Markov model* (LHMM) proposed by Oliver et al. (2002) forms the basis to the layered architectures developed later in this thesis. The functional principle of the LHMM is illustrated in Figure 2.12. The depicted example comprise two layers indexed by k. Each layer consists of a set of HMM for classification and a stream containing the crisp class decisions of the HMM for each time step. The classification is performed using a window which is temporally shifted over the stream of the preceding layer. The stream in each layer is given by a binary matrix $\mathbf{Z}^{(k)} \in \mathbb{R}^{T \times I_k}$ where T represents the last time step available and I_k the number of classes of layer k. In Figure 2.12, the layer $k = 1$ is composed of the window with the HMM for classification at the bottom and the stream of classifier decisions at the top. Since no preceding layer is available in the lowest layer, the window is shifted over the features extracted from the

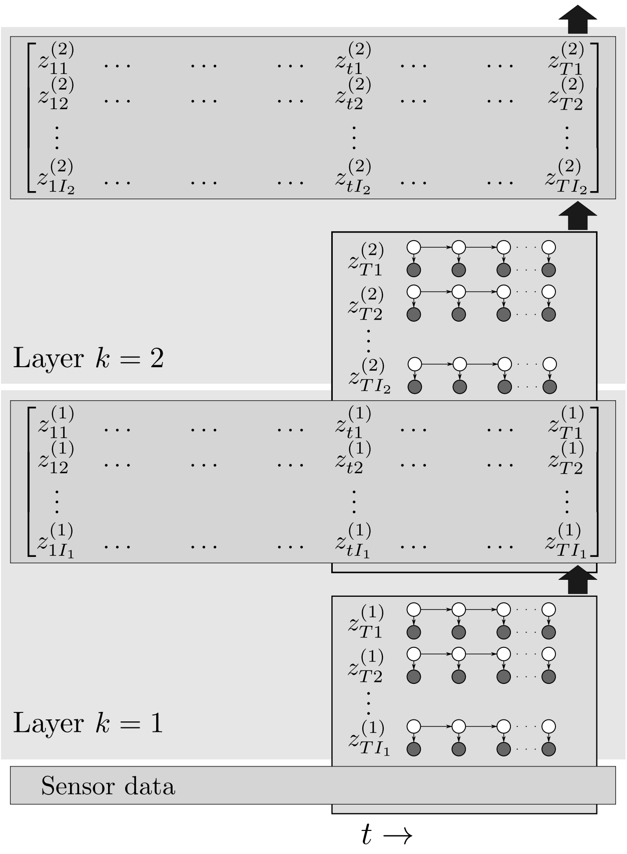

$z^{(2)}_{11}$ … … … $z^{(2)}_{t1}$ … … $z^{(2)}_{T1}$
$z^{(2)}_{12}$ … … … $z^{(2)}_{t2}$ … … $z^{(2)}_{T2}$
$z^{(2)}_{1I_2}$ … … … $z^{(2)}_{tI_2}$ … … $z^{(2)}_{TI_2}$
$z^{(2)}_{T1}$
$z^{(2)}_{T2}$
$z^{(2)}_{TI_2}$
Layer $k = 2$
$z^{(1)}_{11}$ … … … $z^{(1)}_{t1}$ … … $z^{(1)}_{T1}$
$z^{(1)}_{12}$ … … … $z^{(1)}_{t2}$ … … $z^{(1)}_{T2}$
$z^{(1)}_{1I_1}$ … … … $z^{(1)}_{tI_1}$ … … $z^{(1)}_{TI_1}$
$z^{(1)}_{T1}$
$z^{(1)}_{T2}$
$z^{(1)}_{TI_1}$
Layer $k = 1$
Sensor data
$t \rightarrow$

Figure 2.12. Processing in the LHMM as proposed by Oliver et al. (2002). Each layer consists of a set of HMM and a stream of decisions. Each layer recognizes a specific set of classes using a window that is shifted over the input stream. The crisp class decisions form a stream by continuously concatenating the latest class decision. The stream can be accessed by the subsequent layer which uses again a set of HMM to recognize a set of classes. The procedure is repeated until the last layer produces an output stream of class decisions. Schematic drawing adapted from Oliver et al. (2004).

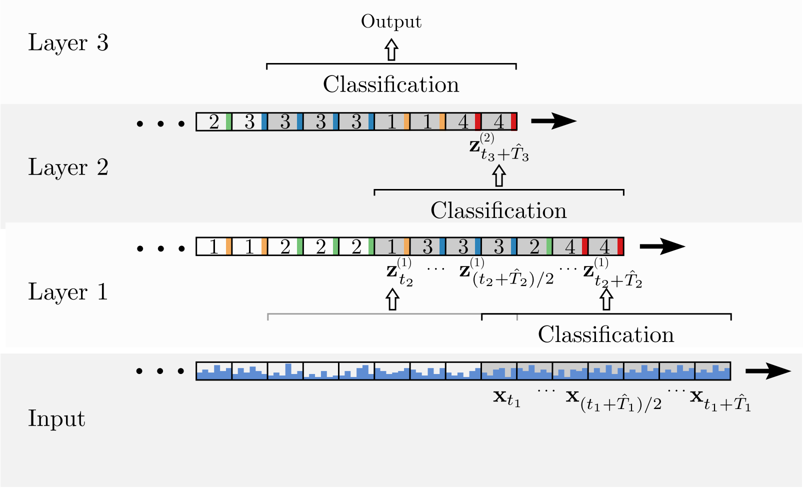

Layer 3
Output
Classification
Layer 2
2 3 3 3 3 1 1 4 4
$\mathbf{z}^{(2)}_{t_3+\hat{T}_3}$
Classification
Layer 1
1 1 2 2 2 1 3 3 3 2 4 4
$\mathbf{z}^{(1)}_{t_2}$ ⋯ $\mathbf{z}^{(1)}_{(t_2+\hat{T}_2)/2}$ ⋯ $\mathbf{z}^{(1)}_{t_2+\hat{T}_2}$
Classification
Input
$\mathbf{x}_{t_1}$ ⋯ $\mathbf{x}_{(t_1+\hat{T}_1)/2}$ ⋯ $\mathbf{x}_{t_1+\hat{T}_1}$

Figure 2.13. LHMM passing discrete class assignments to the next layer. The lowest layer operates on the stream of real-valued features. The subsequent layers operate on the streams of the former layer which is composed of crisp class assignments.

sensor data. The data contained in the window is used by the HMM to recognize multiple sequential sub-patterns. The resulting discrete class assignments are then stored into the binary matrix $\mathbf{Z}^{(1)}$. The subsequent layer $k = 2$ shifts a new window over the stream of the preceding layer's binary matrix $\mathbf{Z}^{(1)}$. The sequence in the window is passed to a new set of HMM in order to recognize the next set of patterns. Again the recognized classes are stored into a binary matrix $\mathbf{Z}^{(2)}$. In the given example, the classes recognized in the second layer are considered as the final output. Figure 2.13 provides a more detailed example of the information encoded in the streams. The set of HMM in the lowest layer operates on a stream of real-valued vectors which are depicted by a sequence of frames containing blue bars with varying heights. The resulting binary class decisions are concatenated to a new stream depicted in the figure by a sequence of frames with discrete color-coded numbers. This stream is then processed in the next layer which results in a new stream of crisp class decisions. The procedure is repeated until the final class decision is available.

The LHMM can recognize more complex patterns in higher layers since they will be composed of sub-patterns provided in preceding layers. A side-effect is that the temporal granularity, i.e. the time covered by the windows, increases in each layer with respect to the sensor data. Furthermore, the LHMM is robust against changes in the environment. Especially upper layers which process more abstract inputs are less dependent on sensor changes. The architecture can be trained in a bottom-up fashion using cross-validation, starting with the lowest layer. Once the first layer is trained, the test set output can be used to form a new complete training set for the subsequent layer. This procedure can be repeated for all layers of the architecture. However, the outputs of the test set are obtained by classifiers which were trained on the development set. Therefore, one can argue that the cross-validation is corrupted. To obtain unbiased generalized outputs, it is necessary to hold back a set of test data which has never been involved in the training process such that an unbiased evaluation of the final layer can be conducted.

Oliver et al. (2002) evaluated the LHMM in an office scenario to detect the desktop activities: “Phone conversation”, “Presentation”, “Face-to-face conversation”, “User present”, “Engaged in some other activity”, “Distant conversation” (outside the field of view) and “Nobody present”. The first layer detects classes using the audio and video modalities. From the audio channel, the system recognizes the classes: “Human speech”, “Music”, “Silence”, “Ambient noise”, “Phone ringing” and “Keyboard typing sounds”. From the video channel the system derives: “Nobody”, “One person”, “One active person present” or “Multiple people present”. The evaluation of the LHMM showed promising results. However, the study possesses some critical issues. Patterns recognized as the final outputs are distinguishable merely by the occurrences of unique events in the preceding layer. For instance, a single person speaking gives clear evidence to a “Phone conversation”, whereas the combination of a single person speaking and keyboard events encode the class “Presentation”. Hence, the temporal progress of the second layer is negligible in the given scenario. Furthermore, only a distinct set of activities can be recognized due to the limited size of the window. Oliver et al. (2004) remarked that passing discrete class assignments to the next layer is disadvantageous since useful information is discarded and, alternatively, suggested to pass the HMM log-likelihoods to the next layer. In fact, in more complex scenarios the propagation of discrete class assignments can be a serious restriction. However, real-valued log-likelihoods are not suitable for the application in LHMM since they need to be modeled by density functions which usually have problems covering the complete range of real-values.

2.11. Fuzzy-in Fuzzy-out Support Vector Machine

As already pointed out in Section 2.9, uncertainty in classification can arise from multiple sources. Recordings in real-world scenarios can be afflicted with noise, for instance due to measurements which are interfering with the target pattern. Furthermore, patterns can have a large variety of appearances such that a simple discriminative function can be insufficient for a successful separation. Uncertainty in real-world scenarios can also originate from ambiguous class definition which leads to inconsistent annotations of data samples. This ambiguity can be soften by using a fuzzy class membership annotation scheme in which class labels range in $[0, 1]$ and sum up to one. One of the most popular advances in statistical learning theory in the last decades is the support vector machine (SVM) (Vapnik, 1999; Christiani and Shawe-Taylor, 2000) which addresses the problem of structural risk minimization by maximizing the margin of the separating hyperplane to the data samples of the different classes. Platt (2000) extended the SVM approach by probabilistic outputs. Thiel et al. (2007) picked up this concept and introduced the *fuzzy-in fuzzy-out support* *vector machine* (F^2-SVM) which extends the SVM and enables it to learn from fuzzy class membership labels. The F^2-SVM will be presented in the following.

The dataset $\mathcal{T} = \{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$ describes a two-class classification problem with N data samples $\mathbf{x}^{(n)}$ where each data sample is associated with a fuzzy class membership $y^{(n)}$ in the range of $[0,1]$. For convenience, the notation is extended by the subscripts $-$ and $+$ to specify the class memberships from the two perspectives $y_{+}^{(n)} = y^{(n)}$ and $y_{-}^{(n)} = 1 - y^{(n)}$. The classification constraints of the F^2-SVM are defined by

$$\begin{aligned} 1 - \xi_{+}^{(n)} &\leq \mathbf{w}^{\mathrm{T}}\mathbf{x}^{(n)} + b && (2.118) \\ -(1 - \xi_{-}^{(n)}) &\geq \mathbf{w}^{\mathrm{T}}\mathbf{x}^{(n)} + b && (2.119) \\ 0 &\leq \xi_{+}^{(n)} && (2.120) \\ 0 &\leq \xi_{-}^{(n)} && (2.121) \end{aligned}$$

where $\mathbf{w}$ and b describe the normal vector and bias of the hyperplane. In contrast to the conventional SVM, two slack variables $\xi_{+}^{(n)}$ and $\xi_{-}^{(n)}$ are given for each of the two class membership perspectives. The objective function is defined by

$$\text{minimize} \quad \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C\sum_{n=1}^{N}(\xi_{+}^{(n)}y_{+}^{(n)} + \xi_{-}^{(n)}y_{-}^{(n)}). \tag{2.122}$$

The slack variable compensates a misclassification of a data sample at the cost of a penalty, as to be seen in Equation 2.118 and Equation 2.119. The penalty is weighted by the fuzzy class membership. The trivial solution of negative slack variables is eliminated by Equation 2.120 and Equation 2.121. The Lagrangian function combines the objective function with the constraints using the Lagrange multipliers $\beta_{+}^{(n)}$, $\beta_{-}^{(n)}$, $\alpha_{+}^{(n)}$ and $\alpha_{-}^{(n)}$:

$$\begin{aligned} L(\mathbf{w}, b, \boldsymbol{\xi}_{+}, \boldsymbol{\xi}_{-}, \boldsymbol{\alpha}_{+}, \boldsymbol{\alpha}_{-}, \boldsymbol{\beta}_{+}, \boldsymbol{\beta}_{-}) = {} & \tfrac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C\textstyle\sum_{n=1}^{N}(\xi_{+}^{(n)}y_{+}^{(n)} + \xi_{-}^{(n)}y_{-}^{(n)}) \\ & - \sum_{n=1}^{N}(\beta_{+}^{(n)}\xi_{+}^{(n)} + \beta_{-}^{(n)}\xi_{-}^{(n)}) \\ & - \sum_{n=1}^{N}\alpha_{+}^{(n)}(\mathbf{w}^{\mathrm{T}}\mathbf{x}^{(n)} + b - 1 + \xi_{+}^{(n)}) \\ & + \sum_{n=1}^{N}\alpha_{-}^{(n)}(\mathbf{w}^{\mathrm{T}}\mathbf{x}^{(n)} + b + (1 - \xi_{-}^{(n)})). \end{aligned} \tag{2.123}$$

The current problem formulation is to complex to find the optimal parameters which minimize the given function. Therefore, the dual form has to be derived which depends only on $\alpha_{+}^{(n)}$ and $\alpha_{-}^{(n)}$. In the following, we refrain from listing all variables in the head of the Lagrangian function to keep the notation uncluttered. The dual form is derived by taking the derivative of $\mathbf{w}$, b, $\xi_{+}^{(n)}$ and $\xi_{-}^{(n)}$ and setting the outcome equal to zero:

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_{n=1}^{N} \alpha_{+}^{(n)} \mathbf{x}^{(n)} + \sum_{n=1}^{N} \alpha_{-}^{(n)} \mathbf{x}^{(n)} = 0 \tag{2.124}$$

$$\Leftrightarrow \qquad \mathbf{w} = \sum_{n=1}^{N} (\alpha_{+}^{(n)} - \alpha_{-}^{(n)}) \mathbf{x}^{(n)} \tag{2.125}$$

$$\frac{\partial L}{\partial b} = -\sum_{n=1}^{N} \alpha_{+}^{(n)} + \alpha_{-}^{(n)} = 0 \tag{2.126}$$

$$\frac{\partial L}{\partial \xi_{+}^{(n)}} = -C y_{+}^{(n)} - \beta_{+}^{(n)} - \alpha_{+}^{(n)} = 0 \tag{2.127}$$

$$\frac{\partial L}{\partial \xi_{-}^{(n)}} = -C y_{-}^{(n)} - \beta_{-}^{(n)} - \alpha_{-}^{(n)} = 0. \tag{2.128}$$

The resulting equations have to be rearranged and substituted back into the original problem formulation of Equation 2.123. Further, rearranging leads to the compact dual form

$$\begin{aligned} L(\boldsymbol{\alpha}_{+}, \boldsymbol{\alpha}_{-}) = & -\frac{1}{2} \sum_{n_1=1}^{N} \sum_{n_2=1}^{N} (\alpha_{+}^{(n_1)} - \alpha_{-}^{(n_1)})(\alpha_{+}^{(n_2)} - \alpha_{-}^{(n_2)}) \mathbf{x}^{(n_1)\mathrm{T}} \mathbf{x}^{(n_2)} \\ & + \sum_{n=1}^{N} (\alpha_{+}^{(n)} + \alpha_{-}^{(n)}) \end{aligned} \tag{2.129}$$

constrained by

$$\sum_{n=1}^{N} (\alpha_{+}^{(n)} + \alpha_{-}^{(n)}) = 0 \tag{2.130}$$

$$0 \leq \alpha_{+}^{(n)} \leq C y_{+}^{(n)} \tag{2.131}$$

$$0 \leq \alpha_{-}^{(n)} \leq C y_{-}^{(n)}. \tag{2.132}$$

The optimization takes the form of a quadratic programming problem which can be solved using the sequential minimal optimization (SMO) algorithm (Platt, 1999; Thiel et al., 2007). Classification of a new data sample $\hat{\mathbf{x}}$ is performed by evaluating

$$d(\hat{\mathbf{x}}) = \sum_{n=1}^{N} (\alpha_{+}^{(n)} - \alpha_{-}^{(n)}) \mathbf{x}^{(n)\mathrm{T}} \hat{\mathbf{x}} + b \tag{2.133}$$

where the sum is usually reduced to iterate only over the support vectors. Data samples are called support vectors in case the corresponding Lagrangian variables α_{+}^{n} and α_{-}^{n} strictly fulfill the Equation 2.131 and Equation 2.132. The outcome $d(\hat{\mathbf{x}})$ represents the distance of the test sample to the hyperplane. The bias b is determined with the help of the support vectors α_{+}^{n} and α_{-}^{n} for which the hyperplane is per definition one. Hence, b can be derived by solving the equation

$$b = \sum_{n=1}^{N} (\alpha_{+}^{(n)} - \alpha_{-}^{(n)}) \mathbf{x}^{(n)\mathrm{T}} \mathbf{x}^{(n)} - 1. \tag{2.134}$$

A discrete class assignment can be performed using the sign of $d(\hat{\mathbf{x}})$. To obtain a fuzzy class membership output, an additional transformation from the distance to the target distribution has to be performed with the help of a sigmoid function (Platt, 2000).

Multi-class classification problems can be addressed by applying one-vs-rest and one-vs-one classifier systems. The corresponding MCS operate on the dataset $\mathcal{T} = \{(\mathbf{x}^{(n)}, \mathbf{y}^{(n)})\}_{n=1}^{N}$ with I classes where $\mathbf{y}^{(n)} = (y_1^{(n)}, \ldots, y_I^{(n)}), y_i^{(n)} \in [0, 1]$ and $\sum_{i=1}^{I} y_i^{(n)} = 1$. According to Thiel et al. (2007), a one-vs-rest classifier system using F^2-SVM can be realized by creating I classification problems of the form $y_{+}^{(n)} = y_i^{(n)}$ and $y_{-}^{(n)} = 1 - y_i^{(n)}$ where $i \in 1, \ldots, I$. The I classifiers are then combined using a multi-class extension of the probabilistic output algorithm (Platt, 2000). A one-vs-one classifier system can be realized by creating $I(I-1)/2$ classifier problems of the form $y_{+}^{(n)} = y_i^{(n)}$ and $y_{-}^{(n)} = y_j^{(n)}$ for each pair $i, j \in 1, \ldots, I$, $i \neq j$. In practical applications, Thiel et al. (2007) propose to reduce the complexity of the problem by including only samples with a membership being above the uniform class distribution. The final fuzzy class membership distribution is obtained by performing pairwise coupling on the classifier outputs as proposed by Hastie and Tibshirani (1998).

2.12. Multi-class F^2-Support Vector Machine

Weston and Watkin (1998) introduced the multi-class support vector machine (MC-SVM) which integrates the training of the multiple SVM usually required for the one-vs-rest classifier systems into a single optimization problem. Schwenker et al. (2014) combined the F^2-SVM and MC-SVM and proposed the *multi-class fuzzy-in and fuzzy-out support vector machine* (MC-F^2-SVM) which will be presented in this section.

The dataset $\mathcal{T} = \{(\mathbf{x}^{(n)}, \mathbf{y}^{(n)})\}_{n=1}^{N}$ describes an I-class classification problem where the fuzzy class memberships are given in form of a vector $\mathbf{y}^{(n)} = (y_1^{(n)}, \ldots, y_I^{(n)})$, $y_i^{(n)} \in [0, 1]$ and $\sum_{i=1}^{I} y_i^{(n)} = 1$. Similarly to Weston and Watkin (1998), the MC-F^2-SVM uses I one-vs-rest hyperplanes. The constraints of the MC-F^2-SVM are given by

$$1 - \xi_i^{(n)} \leq (\mathbf{w}_i^{\mathrm{T}} \mathbf{x}^{(n)} + b_i) - (\mathbf{w}_j^{\mathrm{T}} \mathbf{x}^{(n)} + b_j) \qquad n \in \mathcal{C}_i \text{ and } j \neq i \tag{2.135}$$

$$0 \leq \xi_i^{(n)} \tag{2.136}$$

where $\mathbf{w}_i$ and b_i describe the hyperplane separating class i from the rest of the dataset. The set $\mathcal{C}_i = \{n : n = 1, \ldots, N \mid y_i^{(n)} > y_j^{(n)} \text{ and } j = 1, \ldots, I\}$ contains the data samples for which the class i has the largest class membership compared to other classes. Equation 2.135

specifies the separation functionality of the hyperplanes, whereas Equation 2.136 excludes the trivial solution of negative slack variables. The objective function is defined by

$$\text{minimize} \quad \frac{1}{2}\sum_{i=1}^{I}\mathbf{w}_i^{\mathrm{T}}\mathbf{w}_i + C\sum_{i=1}^{I}\sum_{n\in\mathcal{C}_i}\sum_{j=1}^{I}\xi_i^{(n)}(y_i^{(n)} - y_j^{(n)}). \tag{2.137}$$

Analogously to the F^2-SVM, the fuzzy class memberships are utilized to weight the slack variables. The stronger the class membership compared to the remaining classes, the higher the penalty of the corresponding slack variable. Practically, the term $(y_i^{(n)} - y_j^{(n)})$ in which j is equal to i is canceled out and, hence, can be removed from the sum. However, the sum will remain unchanged in order to keep the notation uncluttered. The complete Lagrangian function is given by

$$\begin{aligned} L(\{\mathbf{w}_i, b_i, \boldsymbol{\xi}_i, \boldsymbol{\alpha}_i, \boldsymbol{\beta}_i\}_{i=1}^{I}) = {} & \frac{1}{2}\sum_{i=1}^{K}\mathbf{w}_i^{\mathrm{T}}\mathbf{w}_i + C\sum_{i=1}^{I}\sum_{n\in\mathcal{C}_i}\sum_{j=1}^{I}\xi_i^{(n)}(y_i^{(n)} - y_j^{(n)}) \\ & + \sum_{i=1}^{I}\sum_{n\in\mathcal{C}_i}\sum_{\substack{j=1\\ j\neq i}}^{I}\alpha_i^{(n)}\big(1 - \xi_i^{(n)} + (\mathbf{w}_i^{\mathrm{T}}\mathbf{x}^{(n)} + b_i) - (\mathbf{w}_j^{\mathrm{T}}\mathbf{x}^{(n)} + b_j)\big) \\ & - \sum_{n=1}^{N}\sum_{i=1}^{I}\beta_i^{(n)}\xi_i^{(n)}. \end{aligned} \tag{2.138}$$

The dual form is derived by taking the derivative with respect to the parameters and setting the result equal to zero

$$\frac{\partial L}{\partial \mathbf{w}_i} = \mathbf{w}_i + \sum_{n\in\mathcal{C}_i}\sum_{\substack{j=1\\ j\neq i}}^{I}\alpha_i^{(n)}\mathbf{x}^{(n)} - \sum_{n\in\mathcal{C}_i}\sum_{\substack{j=1\\ j\neq i}}^{I}\alpha_j^{(n)}\mathbf{x}^{(n)} = 0 \tag{2.139}$$

$$\Leftrightarrow \quad \mathbf{w}_i = \underbrace{\sum_{n\in\mathcal{C}_i}\sum_{\substack{j=1\\ j\neq i}}^{I}\alpha_j^{(n)}\mathbf{x}^{(n)}}_{=:\mathbf{u}_i} - \underbrace{\sum_{n\in\mathcal{C}_i}\sum_{\substack{j=1\\ j\neq i}}^{I}\alpha_i^{(n)}\mathbf{x}^{(n)}}_{=:\mathbf{v}_i} \tag{2.140}$$

$$\frac{\partial L}{\partial b_i} = \sum_{n\in\mathcal{C}_i}\sum_{\substack{j=1\\ j\neq i}}^{I}\alpha_i^{(n)} - \sum_{n\in\mathcal{C}_i}\sum_{\substack{j=1\\ j\neq i}}^{I}\alpha_j^{(n)} = 0 \tag{2.141}$$

$$\Leftrightarrow \quad b_i\sum_{i=1}^{I}\sum_{n\in\mathcal{C}_i}\sum_{\substack{j=1\\ j\neq i}}^{I}\alpha_i^{(n)} - b_j\sum_{j}\sum_{n\in\mathcal{C}_j}\sum_{\substack{i=1\\ i\neq j}}^{I}\alpha_i^{(n)} = 0 \tag{2.142}$$

$$\Leftrightarrow \quad \sum_{i=1}^{I}\sum_{n\in\mathcal{C}_i}\sum_{\substack{j=1\\ j\neq i}}^{I}\alpha_i^{(n)}(b_i - b_j) = 0 \tag{2.143}$$

$$\frac{\partial L}{\partial \xi_i^{(n)}} = C\sum_{\substack{j=1\\ j\neq i}}^{I}(y_i^{(n)} - y_j^{(n)}) - \sum_{\substack{j=1\\ j\neq i}}^{I}\alpha_i^{(n)} - \beta_i^{(n)} = 0. \tag{2.144}$$

Equation 2.140 provides the definition $\mathbf{w}_i = \mathbf{u}_i - \mathbf{v}_i$ which is used to substitute $\mathbf{w}_i$ in the primal form:

$$
\begin{aligned}
L(\{\mathbf{w}_i, b_i, \boldsymbol{\xi}_i, \boldsymbol{\alpha}_i, \boldsymbol{\beta}_i\}_{i=1}^{I}) = \; & \tfrac{1}{2}\textstyle\sum_{i=1}^{I} \big(\mathbf{u}_i - \mathbf{v}_i\big)\big(\mathbf{u}_i - \mathbf{v}_i\big) && (2.145)\\
& + \sum_{i=1}^{I} \sum_{n\in\mathcal{C}_i} \xi_i^{(n)} \underbrace{\Big(\sum_{\substack{j=1\\ j\neq i}}^{I} C(y_i^{(n)} - y_j^{(n)}) - \sum_{\substack{j=1\\ j\neq i}}^{I} \alpha_i^{(n)} - \beta_i^{(n)}\Big)}_{=\ 0 \text{ according to Equation 2.144}} && (2.146)\\
& + \sum_{i=1}^{I} \sum_{n\in\mathcal{C}_i} \sum_{\substack{j=1\\ j\neq i}}^{I} \alpha_i^{(n)} && (2.147)\\
& + \sum_{i=1}^{I} \sum_{n\in\mathcal{C}_i} \sum_{\substack{j=1\\ j\neq i}}^{I} \alpha_i^{(n)}(\mathbf{u}_i - \mathbf{v}_i)^{\mathrm{T}}\mathbf{x}^{(n)} - \alpha_i^{(n)}(\mathbf{u}_j - \mathbf{v}_j)^{\mathrm{T}}\mathbf{x}^{(n)} && (2.148)\\
& + \underbrace{\sum_{i=1}^{I} \sum_{n\in\mathcal{C}_i} \sum_{\substack{j=1\\ j\neq i}}^{I} \alpha_i^{(n)}(b_i - b_j).}_{=0 \text{ according to Equation 2.143}} && (2.149)
\end{aligned}
$$

Further substituting and rearranging using Equation 2.144 and Equation 2.143, results in the final dual form given by

$$
L(\{\boldsymbol{\alpha}_i\}_{i=1}^{I}) = \; -\frac{1}{2}\sum_{i=1}^{I} \big(\mathbf{u}_i - \mathbf{v}_i\big)\big(\mathbf{u}_i - \mathbf{v}_i\big) + \sum_{i=1}^{I} \sum_{n\in\mathcal{C}_i} \sum_{\substack{j=1\\ j\neq i}}^{I} \alpha_i^{(n)}. \tag{2.150}
$$

The dual form is optimized with respect to the lower and upper bounds $0 \leq \alpha_i^{(n)} \leq C(y_i^{(n)} - y_j^{(n)})$ and the equality constraints $\sum_{n\in\mathcal{C}_i} \sum_{\substack{j=1\\ j\neq i}}^{I} \big(\alpha_i^{(n)} - \alpha_j^{(n)}\big) = 0$. The distance to the i^{th} hyperplane is given by

$$
d_i(\hat{\mathbf{x}}) = \; \sum_{n\in\mathcal{C}_i} \sum_{\substack{j=1\\ j\neq i}}^{I} \alpha_j^{(n)}\mathbf{x}^{(n)\mathrm{T}}\hat{\mathbf{x}} - \sum_{n\in\mathcal{C}_i} \sum_{\substack{j=1\\ j\neq i}}^{I} \alpha_i^{(n)}\mathbf{x}^{(n)\mathrm{T}}\hat{\mathbf{x}} + b_i. \tag{2.151}
$$

The bias is determined using a set of linear equations:

$$
\begin{aligned}
1 = \; & (\mathbf{w}_i^{\mathrm{T}}\mathbf{x}^{(n)} + b_i) - (\mathbf{w}_j^{\mathrm{T}}\mathbf{x}^{(n)} + b_j) && (2.152)\\
\Leftrightarrow \qquad b_i - b_j = \; & 1 - \mathbf{w}_i^{\mathrm{T}}\mathbf{x}^{(n)} - \mathbf{w}_j^{\mathrm{T}}\mathbf{x}^{(n)} && (2.153)
\end{aligned}
$$

where n is an index taken from the set of support vectors which fulfill the constraints $0 < \alpha_i^{(n)} < C(y_i^{(n)} - y_j^{(n)})$. The additional constraint $\sum_{i=1}^{I} b_i = 0$ can be added to obtain a problem having a complete rank.

Chapter 3

Methodological Advances in Cognitive Technical Systems

The current chapter presents the scientific contributions based on the basics introduced in the preceding chapter. It begins by presenting the ensemble GMM (EGMM) for enhanced probability density estimation. Subsequently, two extensions to the HMM are proposed, namely the conditioned hidden Markov model (CHMM) and the fuzzy conditioned hidden Markov model (FCHMM). The CHMM extends the HMM by an additional sequence of causal states which influence the selection of hidden states. The FCHMM replaces the sequence of causal states by a sequence of causal fuzzy memberships. The next section addresses the integration of the GPD into the HMM. The integration results in two approaches, namely HMM using GPD (HMM-GPD) and CHMM using GPD (CHMM-GPD). In contrast to the HMM-GPD, the hidden states of the CHMM-GPD can be shared and, therefore, be supported by different classes. Following this, two fusion approaches for multimodal temporal MCS are presented, i.e. the Markov fusion networks (MFN) and the Kalman filter for classifier fusion. Both approaches are able to handle loss of decisions in a unified framework. Subsequently, two concepts for layered architectures are introduced which allow the integration of sub-symbolic and symbolic knowledge and the recognition of complex classes, i.e. the unidirectional layered architecture (ULA) and the bidirectional layered architecture (BLA). The ULA propagates information from the lower layer operating on data derived from sensors to the upper layers in which more complex classes are recognized. The BLA additionally propagates information downwards such that lower layers can benefit from the context known in the upper layers. In the last section, an alternative approach to implement the MC-F^2-SVM is proposed, namely the IC-MC-F^2-SVM which integrates the fuzzy labels into the problem formulation using the inequality constraints.

The approaches in this chapter are designed to be applied in CTS. They follow various paradigms which are of benefit in real-world scenarios being afflicted with many sources of uncertainty. One aspect which is frequently underestimated in such scenarios is the importance of time in pattern recognition. Therefore, most of the proposed approaches process temporal data sequences, i.e. CHMM, FCHMM, HMM-GPD, CHMM-GPD, MFN, Kalman filter for classifier fusion, ULA and BLA. Sequences of incomplete features or decisions which vary in size over time can be handled by the HMM-GPD, CHMM-GPD, MFN and Kalman filter for classifier fusion. The handling of uncertainty is addressed by EGMM, FCHMM, MFN, Kalman filter for classifier fusion, ULA, BLA and IC-MC-F^2-SVM. The integration of sub-symbolic and symbolic information can be realized using the CHMM, FCHMM, ULA and BLA. Multi-class problems are addressed by the CHMM, CHMM-GPD, MFN, Kalman filter for classifier fusion, ULA, BLA and IC-MC-F^2-SVM. Multimodal decisions derived from different channels can be combined using the CHMM, MFN and Kalman filter for classifier fusion. In the following, the proposed approaches will be explained in greater detail along with their properties.

3.1. Ensemble GMM

As outlined in Section 2.2, the EM algorithm for the GMM has several drawbacks (Bishop, 2006, p. 434). First of all, it is unlikely that a density was generated by a mixture of Gaussians. Therefore, it is generally difficult to find an optimal GMM parameter configuration, i.e. number of mixture components or constraints on the covariance matrix (spherical or diagonal). The EM algorithm depends on the initialization of parameters and, hence, may end in different local optima. Furthermore, the EM algorithm suffers from the appearance of singularities, i.e. one mixture component can become responsible for a single observation.

The *ensemble GMM* (EGMM) addresses these issues by making use of the MCS concept of ensembles (Glodek et al., 2013c, 2010). The ensemble technique increases the robustness and performance of models by combining a set of diverse models (Kuncheva, 2004, p. 237ff.). The EGMM follows the same principles and combines a set of diverse GMM.

Ormoneit and Tresp (1998) first suggested the combination of a set of GMM to obtain an improved density. The proposed approaches make use of subset averaging (without replacement) and bagging (with replacement). Furthermore, two extensions to the GMM EM algorithm were presented, namely the maximum penalized likelihood and the so-called Bayesian approach. The two extensions regularize the estimation of the GMM by the means of a conjugated prior to the model parameters. However, no variations in the initial GMM configurations were suggested. An alternative approach named aggregated EM (Ag-EM) was proposed by (Shinozaki and Kawahara, 2008). The Ag-EM algorithm divides the dataset into L partitions in order to obtain L different estimates in the E-step. The estimations are then used in the M-step to create M different models. As a result, the following E-step creates $L \times M$ estimates which are then reduced down to L estimates using averaging. The iteration starts again by creating M models in the following M-step. A major limitation of this approach is that the GMM configurations must be identical for all models to be combined (Shinozaki and Kawahara, 2008).

The EGMM combines a set of M individually trained GMM. The diversity of these models is realized by using subsets of the training data, random initialization of parameters, i.e. mixture components, means and covariances, and differently shaped covariance matrix. The EGMM algorithm is given by following listing:

Input: A dataset $\mathcal{T} = \{\mathbf{x}^{(n)} \; : \; \mathbf{x}^{(n)} \in \mathbb{R}^D, n = 1, \ldots, N\}$, the number of potential ensemble members L, the number of members in the final ensemble $M \leq L$.

1. Generate a set of initial GMM parameters $\{\theta_l\}_{l=1}^{L}$ and datasets $\{\mathcal{T}_l\}_{l=1}^{L}$.
   a) Create the GMM configuration θ_l, e.g. by varying the number of mixture components or the shape of the covariance matrices
   b) Create the training set $\mathcal{T}_l$ from $\mathcal{T}$, e.g. using bootstrap samples (Breiman, 1996).
2. Make use of the EM algorithm to maximize the likelihoods of the GMM parameterized by $\theta_1, \ldots, \theta_L$.
3. Select M out of the L GMM which have the highest log-likelihoods. Without loss of generality these M GMM are parameterized by $\theta_1, ..., \theta_M$. The EGMM is called pruned EGMM if $M < L$.
4. Compute the weights $w_1, \ldots, w_M$ with $w_m \in [0, 1]$ and $\sum_{m=1}^{M} w_m = 1$, e.g. $w_m = 1/M$.

Output: The EGMM density of a new data sample $\hat{\mathbf{x}}$ is obtained by evaluating $\mathcal{E}(\hat{\mathbf{x}}, \{\theta_m\}_{m=1}^{M}) = \sum_{m=1}^{M} w_m \mathcal{G}(\hat{\mathbf{x}}, \theta_m)$.

The averaging preserves the properties of the probability density, e.g. smoothness or $\int_{-\infty}^{\infty} \mathcal{E}(\hat{\mathbf{x}}, \theta_m) \mathrm{d}\mathbf{x} = 1$. The parameter M should be chosen with the objective to limit the complexity of the final probability density function or to discard GMM with singularities. Alternatively, singularities can be compensated by using a suitable combination rule such as the point-wise median which is robust against ensemble outliers. However, this combination rule provides an estimate will not integrate to one.

3.2. Conditioned Hidden Markov Model

The HMM is still one of the most frequently used approaches to model sequences in pattern recognition. Classification using HMM is performed by creating specific models for each class. The class assignment is then determined according to the model with the highest likelihood given the test sequence. The *conditioned hidden Markov model* (CHMM) was developed in order to create a unified model such that more information can be drawn from limited training data. The hidden states of the CHMM represent the entire dataset such that a mapping from classes to hidden states is performed in order to obtain a class assignment. Furthermore, the CHMM can be applied to perform a unified training of datasets containing symbolic and sub-symbolic sequences. The CHMM extends the HMM by an additional sequence of outer causes $\mathbf{y}$ which influence the hidden states. The likelihood of the CHMM is given by

$$\begin{aligned} p(\mathbf{X}|\mathbf{y} \shortmid \lambda) = & \sum_{\mathbf{w}\in\Omega_{\mathsf{w}}^{T}} p(\mathsf{w}_1 = w_1|y_1 \shortmid \boldsymbol{\pi}) \prod_{t=2}^{T} p(\mathsf{w}_t = w_t|\mathsf{w}_{t-1} = w_{t-1}, y_t \shortmid \mathbf{A}) \cdot \\ & \prod_{t=1}^{T} p(\mathbf{x}_t|\mathsf{w}_t = w_t \shortmid \boldsymbol{\theta}) \end{aligned} \tag{3.1}$$

where the set $\lambda = \{\boldsymbol{\pi}, \mathbf{A}, \boldsymbol{\theta}\}$ contains the individual model parameters. In contrast to the HMM, the initial hidden state probability and the transition probability are additionally influenced by the random variable y_t. The probability of the initial hidden state is parameterized by $\boldsymbol{\pi}$ with a total number of $|\Omega_{\mathsf{w}}| \times |\Omega_{\mathsf{y}}|$ parameters. The transition probability is given by the matrix $\mathbf{A}$ and has $|\Omega_{\mathsf{w}}| \times |\Omega_{\mathsf{w}}| \times |\Omega_{\mathsf{y}}|$ parameters. Figure 3.1 shows the graphical model of the CHMM where nodes colored in dark grayed are observed variables and nodes colored in lighter gray are only observable during training, but unknown when testing a new sequence. The sequence of hidden random variables $\boldsymbol{\mathsf{w}} = (\mathsf{w}_1, \ldots, \mathsf{w}_T)$ effects the sequence of observations. The hidden variables themselves are influenced by the sequence of outer causes (or labels) $\boldsymbol{\mathsf{y}}$. The joint probability given the test sequences $\hat{\mathbf{X}}$ and $\hat{\mathbf{y}}$ is determined by $p(\hat{\mathbf{X}}, \hat{\mathbf{y}}) = p(\hat{\mathbf{X}}|\hat{\mathbf{y}})p(\hat{\mathbf{y}})$ where $p(\hat{\mathbf{y}})$ can be regarded as a prior over the sequence of labels. The distribution over the outer causes is given by

$$p(\hat{\mathbf{y}}|\hat{\mathbf{X}}) = \frac{p(\hat{\mathbf{X}}, \hat{\mathbf{y}})}{\sum_{\mathbf{y}\in\Omega_{\mathsf{y}}^{T}} p(\hat{\mathbf{X}}, \mathbf{y})} \quad \forall \hat{\mathbf{y}} \in \Omega_{\mathsf{y}}^{T}. \tag{3.2}$$

There are two types of inference problems concerning the CHMM. The first one addresses the inference of the probability distribution over hidden states $\boldsymbol{\mathsf{w}}$ given a sequence of observations $\mathbf{X}$ and a sequence of causal states $mathbfY$. The second inference problem aims at inferring the probability distribution of the sequence of hidden states $\boldsymbol{\mathsf{w}}$ and the sequence of causal states $\boldsymbol{\mathsf{y}}$ given a sequence of observations $\mathbf{X}$. The first problem is required for training, but can also be used for testing a limited set of causal state sequences. For instance, in case of testing causal sequences which contain only one kind of state in pre-segmented datasets. It might also be promising to identify the temporal location of a label change. To do so,

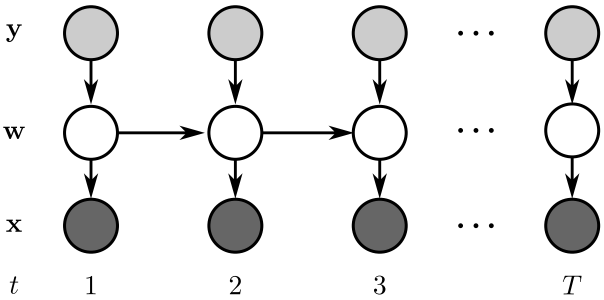


Figure 3.1. Graphical representation of the CHMM. The Markov chain **w** is composed of a sequence of hidden random variables w_t. Each hidden random variable w_t, except the one of the first time step, depends on its predecessor w_{t-1}. Furthermore, each hidden random variable emits an observation $\mathbf{x}_t$. The selection of the hidden state w_t is influenced by the hidden state w_{t-1} and by the outer cause y_t.

the training has to be performed using sequences composed of at least two labels. Testing can then be done on a fixed set of causal sequences in which the first half of the sequence is composed of one label and the second half of another label. Performing the exact inference for both random variables **w** and **y** allows direct access to the probability distribution of the causal sequence. This topic is addressed in detail in Section 3.3.

The sequence of outer causes can also be used to provide additional symbolic knowledge about which hidden state is more likely. To do so, the sequence of outer causes is augmented by evidential information. The outer cause random variable takes states from the Cartesian product of the label and evidence states. Since this procedure eventually significantly extends the parameter space, it may be advisable to fall back to the HMM classification scheme. In other words, to use only one label state. Making use of external knowledge relates the CHMM to the coupled HMM proposed by Brand et al. (1997). In coupled HMM, hidden states at time step t of one HMM chain are additionally influenced by the hidden states in time step $t-1$ of another HMM chain. In contrast to the coupled HMM, the CHMM does not use multiple chains of HMM. The hidden states of the CHMM are influenced by outer causes provided by an independent classifier or another external sources.

The LDCRF described in Section 2.6 extends the HMM in a similar way to the CHMM, but uses a Markov network. However, the structure of the LDCRF allows exact inference such that there is no need to resign from Bayesian networks. Especially, because of the comprehensive work already achieved by modeling the HMM as a Bayesian network. Furthermore, Morency et al. (2007) restricted the model to have a fixed association between labels and hidden states. While this is advantageous in order to save parameters and, therefore, to achieve a good convergence property, the drawback is of course the restricted model. Another similar structure is given by the input-output hidden Markov model (IOHMM) which was proposed by Bengio and Frasconi (1996). In contrast to the IOHMM, the CHMM output does not directly dependent on the input variable. Furthermore, the objective of the CHMM is not to transform an input sequence to an output sequence, but to infer from the possible underlying input sequence the output sequence.

In the following, inference and parameter learning for CHMM is explained in detail. Furthermore, the HMM-MCS and CHMM-MCS for multimodal classifier fusion is introduced.

3.2.1. Inference

Forward algorithm

Inference in CHMM can be performed by modifying the forward-backward algorithm of the HMM presented in Section 2.5.1. The new forward variable $\alpha_{t,\mathbf{y}}(j)$ is defined as the probability to be in the hidden state j at time step t and to have observed the sequence $\mathbf{X}_{1..t}$ from the first time step to the time step t given the corresponding sequence of outer causes $\mathbf{y}_{1..t}$. Analogously to the HMM, the forward algorithm uses dynamic programming such that the recursive formula of the forward variable is given by

$$\begin{aligned}
\alpha_{t,\mathbf{y}}(j) &= p(\mathbf{X}_{1..t}, \mathsf{w}_t = j|\mathbf{y}_{1..t}) && (3.3)\\
&= p(\mathbf{x}_t|\mathsf{w}_t = j)p(\mathbf{X}_{1..t-1}, \mathsf{w}_t = j|\mathbf{y}_{1..t}) && (3.4)\\
&= p(\mathbf{x}_t|\mathsf{w}_t = j)\sum_{i\in\Omega_\mathsf{w}} p(\mathbf{X}_{1..t-1}, \mathsf{w}_{t-1} = i, \mathsf{w}_t = j|\mathbf{y}_{1..t}) && (3.5)\\
&= p(\mathbf{x}_t|\mathsf{w}_t = j)\cdot\\
&\qquad \sum_{i\in\Omega_\mathsf{w}} p(\mathbf{X}_{1..t-1}, \mathsf{w}_{t-1} = i|\mathbf{y}_{1..t-1})p(\mathsf{w}_t = j|\mathsf{w}_{t-1} = i, y_t) && (3.6)\\
&= p(\mathbf{x}_t|\mathsf{w}_t = j)\sum_{i\in\Omega_\mathsf{w}} \alpha_{t-1,\mathbf{y}}(i)\; p(\mathsf{w}_t = j|\mathsf{w}_{t-1} = i, y_t). && (3.7)
\end{aligned}$$

Backward algorithm

The backward algorithm starts from the end of the sequences $\mathbf{X}$ and $\mathbf{y}$ such that the backward variable is given by

$$\begin{aligned}
\beta_{t-1,\mathbf{y}}(j) &= p(\mathbf{X}_{t..T}|\mathsf{w}_{t-1} = j, \mathbf{y}_{t..T}) && (3.8)\\
&= \sum_{i\in\Omega_\mathsf{w}} p(\mathbf{X}_{t..T}, \mathsf{w}_t = i|\mathsf{w}_{t-1} = j, \mathbf{y}_{t..T}) && (3.9)\\
&= \sum_{i\in\Omega_\mathsf{w}} p(\mathbf{X}_{(t+1)..T}|\mathsf{w}_t = i, \mathbf{y}_{(t+1)..T})\cdot\\
&\qquad p(\mathbf{x}_t|\mathsf{w}_t = i)p(\mathsf{w}_t = i|\mathsf{w}_{t-1} = j, y_t) && (3.10)\\
&= \sum_{i\in\Omega_\mathsf{w}} \beta_{t,\mathbf{y}}(i)p(\mathbf{x}_t|\mathsf{w}_t = i)p(\mathsf{w}_t = i|\mathsf{w}_{t-1} = j, y_t). && (3.11)
\end{aligned}$$

The initial and terminal values of the forward and backward variables are given by

$$\begin{aligned}
\alpha_{1,\mathbf{y}}(j) &= p(\mathbf{x}_1|\mathsf{w}_1 = j)p(\mathsf{w}_1 = j|y_1) && (3.12)\\
\beta_{T,\mathbf{y}}(j) &= 1 && (3.13)\\
\beta_{0,\mathbf{y}}(j) &= \sum_{i\in\Omega_\mathsf{w}} \beta_{1,\mathbf{y}}(j)p(\mathbf{x}_1|\mathsf{w}_1 = i)p(\mathsf{w}_1 = i|y_1). && (3.14)
\end{aligned}$$

Once the forward and backward variables of the sequences $\mathbf{y}$ and $\mathbf{X}$ are determined, the responsibility and transition probability can be computed. These quantities are later required for parameter learning. The responsibility and transition probability are given by

$$\gamma_{t,\mathbf{y}}(j) = \frac{\alpha_{t,\mathbf{y}}(j)\beta_{t,\mathbf{y}}(j)}{p(\mathbf{X})} \tag{3.15}$$

$$\xi_{t-1,t,\mathbf{y}}(i,j) = \frac{\alpha_{t-1,\mathbf{y}}(i)p(\mathbf{x}_t|\mathsf{w}_t = j)p(\mathsf{w}_t = j|\mathsf{w}_{t-1} = i, y_t)\beta_{t,\mathbf{y}}(j)}{p(\mathbf{X})}. \tag{3.16}$$

The probability of $\mathbf{X}$ given $\mathbf{y}$ is derived by the summing up $\alpha_{T,\mathbf{y}}(i)$ over all hidden states i:

$$p(\mathbf{X}|\mathbf{y}) = \sum_{i\in\Omega_{\mathsf{w}}} \alpha_{T,\mathbf{y}}(i). \tag{3.17}$$

Analogously to the HMM, the probabilities of the forward and backward variables can take values close to the machine precision. Therefore, it is again advisable to use scaling factors.

3.2.2. Parameter Learning

EM-algorithm

The M-step of the EM algorithm maximizes the expectation of the posterior distribution over the hidden states with the help of the complete data log-likelihood given by

$$Q(\lambda, \lambda^{old}) = \sum_{n=1}^{N} \sum_{\mathbf{w}\in\Omega_{\mathsf{w}}^{T}} p(\mathbf{w}|\mathbf{X}^{(n)}, \mathbf{y}^{(n)} \mid \lambda^{old}) \ln p(\mathbf{X}^{(n)}, \mathbf{w}|\mathbf{y}^{(n)} \mid \lambda). \tag{3.18}$$

where the sequences $\mathbf{X}^{(n)}$ and $\mathbf{y}^{(n)}$ are observation and label sequences of the dataset $\mathcal{T} = \{(\mathbf{X}^{(n)}, \mathbf{y}^{(n)})\}_{n=1}^{N}$. Substituting the responsibility and transition probability of the E-step into the complete data log-likelihood and expanding the equation results in

$$\begin{aligned} Q(\lambda, \lambda^{old}) = \sum_{n=1}^{N} \sum_{\mathbf{w}\in\Omega_{\mathsf{w}}^{T}} & p(\mathbf{w}|\mathbf{X}^{(n)}, \mathbf{y}^{(n)} \mid \lambda^{old})\cdot \\ & \ln\Bigg(p(\mathsf{w}_1 = w_1|\mathsf{y}_1 = y_1^{(n)} \mid \boldsymbol{\pi})\cdot \\ & \prod_{t=2}^{T} p(\mathsf{w}_t = w_t|\mathsf{w}_{t-1} = w_{t-1}, \mathbf{y}_1 = y_1^{(n)} \mid \mathbf{A})\cdot \\ & \prod_{t=1}^{T} p(\mathbf{x}_t = \mathbf{x}_t^{(n)}|\mathsf{w}_t = w_t \mid \boldsymbol{\theta})\Bigg) \end{aligned} \tag{3.19}$$

$$= \sum_{n=1}^{N} \sum_{w_1 \in \Omega_{\mathrm{w}}} \gamma_{1,\mathbf{y}^{(n)}}^{(n)}(w_1) \ln \pi_{w_1, y_1^{(n)}} +$$

$$\sum_{t=2}^{T} \sum_{w_{t-1} \in \Omega_{\mathrm{w}}} \sum_{w_t \in \Omega_{\mathrm{w}}} \xi_{t-1,t,\mathbf{y}^{(n)}}^{(n)}(w_{t-1}, w_t) \ln a_{w_{t-1}, w_t, y_t^{(n)}} +$$

$$\sum_{t=1}^{T} \sum_{w_t \in \Omega_{\mathrm{w}}} \gamma_{t,\mathbf{y}^{(n)}}^{(n)}(w_t) \ln p(\mathbf{x}_t^{(n)} | \mathsf{w}_t = w_t \shortmid \boldsymbol{\theta}) \tag{3.20}$$

$$= \sum_{n=1}^{N} Q^{(n)}(\lambda, \lambda^{\mathrm{old}}). \tag{3.21}$$

The target parameter are maximized by taking the corresponding derivatives of $Q(\lambda, \lambda^{old})$. The outcome is then set equal to zero such that the equation can be rearranged to obtain the update formula

First, the update formula with respect to the initial state probability $\boldsymbol{\pi}$ is derived. A constraint which is weighted by the Lagrange multiplier κ is added in order to assert that the corresponding probability distribution will sum up to one. Furthermore, it is presumed without loss of generalization that the first element of the outer causal sequence $y_1^{(n)}$ is in state y_1. The following equation is obtained:

$$\frac{\partial Q(\lambda, \lambda^{\mathrm{old}})^{(n)}}{\partial \pi_{w_1, y_1}} = \frac{\partial}{\partial \pi_{w_1, y_1}} \sum_{j \in \Omega_{\mathrm{w}}} \gamma_{1,\mathbf{y}^{(n)}}^{(n)}(j) \ln \pi_{w_t, y_1} - \kappa \left(\sum_{j \in \Omega_{\mathrm{w}}} \pi_{j, y_1} - 1 \right) = 0 \tag{3.22}$$

$$\Leftrightarrow \quad \frac{\gamma_{1,\mathbf{y}^{(n)}}^{(n)}(w_1)}{\pi_{w_1, y_1}} - \kappa \quad = 0 \tag{3.23}$$

$$\Leftrightarrow \quad \gamma_{1,\mathbf{y}^{(n)}}^{(n)}(w_1) - \pi_{w_1, y_1} \kappa = 0. \tag{3.24}$$

The Lagrangian multiplier κ is determined by adding a sum over the hidden states and solving the equation:

$$\Rightarrow \sum_{j \in \Omega_{\mathrm{w}}} \gamma_{1,\mathbf{y}^{(n)}}^{(n)}(j) - \pi_{j, y_1} \kappa = 0 \tag{3.25}$$

$$\Leftrightarrow \quad \sum_{j \in \Omega_{\mathrm{w}}} \kappa \pi_{j, y_1} = \sum_{j \in \Omega_{\mathrm{w}}} \gamma_{1,\mathbf{y}^{(n)}}^{(n)}(j) \tag{3.26}$$

$$\Leftrightarrow \quad \kappa = \sum_{j \in \Omega_{\mathrm{w}}} \gamma_{1,\mathbf{y}^{(n)}}^{(n)}(j). \tag{3.27}$$

The outcome is substituted back into Equation 3.24 which results in

$$\Rightarrow \quad \frac{\gamma_{1,\mathbf{y}^{(n)}}^{(n)}(w_1)}{\pi_{w_1, y_1}} - \kappa = 0 \tag{3.28}$$

$$\Leftrightarrow \frac{\gamma_{1,\mathbf{y}^{(n)}}^{(n)}(w_1)}{\pi_{w_1, y_1}} - \sum_{j \in \Omega_{\mathrm{w}}} \gamma_{1,\mathbf{y}^{(n)}}^{(n)}(j) = 0 \tag{3.29}$$

$$\Leftrightarrow \qquad \pi_{w_1,y_1} = \frac{\gamma^{(n)}_{1,\mathbf{y}^{(n)}}(w_1)}{\sum_{j\in\Omega_{\mathsf{w}}} \gamma^{(n)}_{1,\mathbf{y}^{(n)}}(j)}. \tag{3.30}$$

The next step is to extend the equation to the complete dataset. An additional modification is required since the first outer cause state might not be the one of interest. Therefore, the function $\delta_{y_1^{(n)}=y_1}$ is used to select the target state. The final update formula is given by

$$\pi_{w_1,y_1} = \frac{\sum_{n=1}^{N} \delta_{y_1^{(n)}=y_1} \gamma^{(n)}_{1,\mathbf{y}^{(n)}}(w_1)}{\sum_{n=1}^{N} \sum_{j\in\Omega_{\mathsf{w}}} \delta_{y_1^{(n)}=y_1} \gamma^{(n)}_{1,\mathbf{y}^{(n)}}(j)}. \tag{3.31}$$

In the next step, the update formula for the element $a_{v,w,y}$ of the state transition matrix $\mathbf{A}$ is derived. To begin with, it is assumed that the sequence $\mathbf{y}^{(n)}$ is composed exclusively of the target state y. In order to derive the update formula, the relevant terms of Equation 3.20 are kept and a regularization term to ensure that the parameters are correctly normalized with a Lagrange multiplier κ is added. Hence, the derivate is given by

$$\begin{aligned}
\frac{\partial Q(\lambda,\lambda^{\text{old}})}{\partial a_{v,w,y}} =& \frac{\partial}{\partial a_{v,w,y}} \sum_{t=2}^{T} \sum_{i\in\Omega_{\mathsf{w}}} \sum_{j\in\Omega_{\mathsf{w}}} \xi^{(n)}_{t-1,t,\mathbf{y}^{(n)}}(i,j) \ln a_{i,j,y} - \\
& \sum_{i\in\Omega_{\mathsf{w}}} \kappa \Big(\sum_{j\in\Omega_{\mathsf{w}}} a_{i,j,y} - 1 \Big) = 0 && (3.32)\\
\Leftrightarrow \qquad & \sum_{t=2}^{T} \frac{\xi^{(n)}_{t-1,t,\mathbf{y}^{(n)}}(v,w)}{a_{v,w,y}} - \kappa = 0 && (3.33)\\
\Leftrightarrow \qquad & \sum_{t=2}^{T} \xi^{(n)}_{t-1,t,\mathbf{y}^{(n)}}(v,w) - a_{v,w,y}\kappa = 0. && (3.34)
\end{aligned}$$

The Lagrange multiplier κ is determined by introducing a sum over the hidden states with respect to the random variable w_t. Rearranging the expression to obtain the Lagrangian multiplier results in

$$\begin{aligned}
& \sum_{t=2}^{T} \sum_{j\in\Omega_{\mathsf{w}}} \xi^{(n)}_{t-1,t,\mathbf{y}^{(n)}}(v,j) - a_{v,j,y}\kappa = 0 && (3.35)\\
\Leftrightarrow \qquad & \sum_{t=2}^{T} \sum_{j\in\Omega_{\mathsf{w}}} a_{v,j,y}\kappa = \sum_{t=2}^{T} \sum_{j\in\Omega_{\mathsf{w}}} \xi^{(n)}_{t-1,t,\mathbf{y}^{(n)}}(v,j) && (3.36)\\
\Leftrightarrow \qquad & \kappa = \sum_{t=2}^{T} \sum_{j\in\Omega_{\mathsf{w}}} \xi^{(n)}_{t-1,t,\mathbf{y}^{(n)}}(v,j). && (3.37)
\end{aligned}$$

Substituting the expression back into Equation 3.34 yields in

$$\Rightarrow \sum_{t=2}^{T} \frac{\xi^{(n)}_{t,\mathbf{y}^{(n)}}(v,w)}{a_{v,w,y}} - \kappa = \sum_{t=2}^{T} \frac{\xi^{(n)}_{t,\mathbf{y}^{(n)}}(v,w)}{a_{v,w,y}} - \sum_{j\in\Omega_{\mathsf{w}}} \xi^{(n)}_{t,\mathbf{y}^{(n)}}(v,j) = 0 \tag{3.38}$$

$$\Leftrightarrow \qquad a_{v,w,y} = \frac{\sum_{t=2}^{T} \xi_{t,\mathbf{y}^{(n)}}(v,w)}{\sum_{t=2}^{T} \sum_{j \in \Omega_{\mathsf{w}}} \xi_{t,\mathbf{y}^{(n)}}^{(n)}(v,j)}. \tag{3.39}$$

The final update formula is then obtained by extending the equation to the complete dataset and taking into account that y_t might have different target states than y:

$$a_{v,w,y} = \frac{\sum_{n=1}^{N} \sum_{t=2}^{T} \delta_{y_t^{(n)}=y} \xi_{t,\mathbf{y}^{(n)}}^{(n)}(v,w)}{\sum_{n=1}^{N} \sum_{t=2}^{T} \sum_{j \in \Omega_{\mathsf{w}}} \delta_{y_t^{(n)}=y} \xi_{t,\mathbf{y}^{(n)}}^{(n)}(v,j)}. \tag{3.40}$$

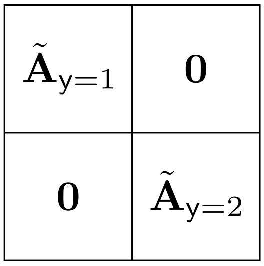
The update formula with respect to the parameters of the observation probability are identical to the ones of the classic HMM and, hence, can be applied without any changes:

$$\frac{\partial Q(\lambda, \lambda^{\mathrm{old}})}{\partial \theta_i} = \frac{\partial}{\partial \theta_i} \sum_{t=1}^{T} \sum_{j \in \mathsf{w}_t} \gamma_{t,\mathbf{y}}^{(n)}(j) \ln p(\mathbf{x}_t | \mathsf{w}_t = j \mid \boldsymbol{\theta}) = 0. \tag{3.41}$$

Therefore, the reader is referred to the Section 2.5.3 and 2.5.4.

Generally, the CHMM needs to be trained with an increased numbers of hidden states, because, conversely to the classic HMM, the CHMM needs to cover data of all classes. A qualified initialization of the observation parameters can be performed by first partitioning the data using the labels of the causal sequence and then carrying out a clustering according to the number of required hidden states. If the causal sequences are composed exclusively of one target label, a pre-training with HMM for each label can be performed. The resulting maximized parameters $\tilde{\mathbf{a}}_{\mathsf{y}=y}$, $\tilde{\mathbf{A}}_{\mathsf{y}=y}$ and $\tilde{\theta}_{\mathsf{y}=y}$ can then be inserted into the initial CHMM. There are two ways to handle the resulting unassigned elements in the CHMM transition matrix which yield in two approaches: the restricted CHMM and the unrestricted CHMM. The transition matrix $\mathbf{A}$ of the restricted CHMM is shown in Figure 3.2. The undetermined values of the matrix are set to zero such that the expressiveness of the CHMM is reduced to the setting of a classic HMM. However, in this setting the classification is performed using the probability distribution $p(\mathsf{y} = \mathbf{y} | \mathsf{X} = \mathbf{X})$ and not the model likelihood. In case of the unrestricted CHMM, a subset or all unassigned parameters of the transition matrix are set

Figure 3.2. Initialization of the restricted CHMM transition matrix using pre-trained HMM. The restricted CHMM has a sparsely initialized transition matrix. Provided, that the training label sequences contain only one sort of label, an HMM is trained for each set of label sequences. The resulting transition matrices $\tilde{\mathbf{A}}_{\mathsf{y}=y}$ are inserted into the transition matrix of the CHMM. Free values are set to zero.



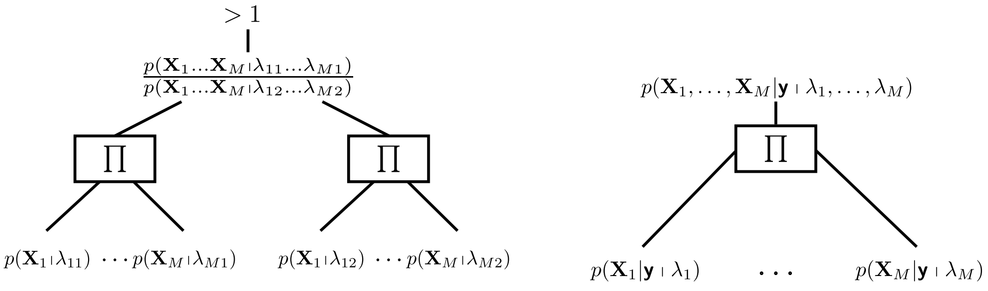
$\tilde{\mathbf{A}}_{\mathsf{y}=1}$	$\mathbf{0}$
$\mathbf{0}$	$\tilde{\mathbf{A}}_{\mathsf{y}=2}$

to random values. This approach increases the model's complexity but can also increase the model expressiveness. Ideally, the training of the unrestricted CHMM is performed using causal sequences containing multiple labels such that the transitions between labels are learned.

3.2.3. Late Classifier Fusion Using CHMM

The classification using the HMM is typically performed by individually training one model for each class and assigning the class by selecting the model which has the highest likelihood for a given test sequence. However, in some cases it is favorable to train multiple HMM per class and to perform late fusion. This may be in order to benefit from the advantages of late fusion, to perform a multimodal classification or to compensate different sample rates. In the following, late fusion using HMM (HMM-MCS) and late fusion using CHMM (CHMM-MCS) are presented and compared to each other.

The classification problem to be solved has I classes and sequences obtained from M modalities. In case of HMM-MCS, one HMM is trained for each class and modality resulting in a number of $I \cdot M$ HMM. A class assignment is performed by evaluating the likelihoods of all models. The likelihoods of each class are multiplied to obtain I combined likelihoods. The class is assigned to the highest combined likelihood. The CHMM-MCS requires only one model to be trained for each modality such that M CHMM are obtained. Each CHMM provides a probability distribution which are regarded as being independent to each other. Hence, the distributions can be combined by multiplication and a subsequent normalization. The respective operations should be performed in logarithmic space to avoid numerical imprecisions.


> 1
$\frac{p(\mathbf{X}_1 ... \mathbf{X}_M | \lambda_{11} ... \lambda_{M1})}{p(\mathbf{X}_1 ... \mathbf{X}_M | \lambda_{12} ... \lambda_{M2})}$
Π
Π
$p(\mathbf{X}_1 | \lambda_{11}) \cdots p(\mathbf{X}_M | \lambda_{M1})$
$p(\mathbf{X}_1 | \lambda_{12}) \cdots p(\mathbf{X}_M | \lambda_{M2})$
$p(\mathbf{X}_1, \ldots, \mathbf{X}_M | \mathbf{y} | \lambda_1, \ldots, \lambda_M)$
Π
$p(\mathbf{X}_1 | \mathbf{y} | \lambda_1)$
$\cdots$
$p(\mathbf{X}_M | \mathbf{y} | \lambda_M)$

(a) HMM-MCS

(b) CHMM-MCS

Figure 3.3. HMM-MCS and CHMM-MCS late fusion techniques to combine the output of the HMM and the CHMM exemplified using a two-class classification problem and *M* feature sets. (a) Fusion architecture using the classic HMM. For each class, all M HMM outputs are combined independently by multiplying the probabilities derived for each feature. The final class assignment is obtained by the ratio of the resulting two probabilities. (b) Fusion architecture using the CHMM. The CHMM provides a class distribution for each of the M features. These distributions can be combined using decision averaging or, as illustrated, the naïve Bayes approach denoted by the product.

Figure 3.3 depicts the differences between the HMM-MCS and CHMM-MCS using the example of a two-class classification problem with M modalities. Figure 3.3a shows the HMM-MCS in which a model λ_{ym} is trained for each class and modality. Testing a new sequence is performed by inferring the likelihoods of all HMM and combining the results of each modality. The class decision is derived by selecting the class with the highest combined likelihood or, as shown in the figure, by determining the ratio of the probabilities. The CHMM-MCS is shown in Figure 3.3b. Since the CHMM provides a probability for a labeled sequence, only one model has to be trained for each modality. The testing is performed by determining the probability distributions of all possible label sequences. These probability distributions are then combined by multiplication. The final assignment to a class is carried out by choosing the label sequence with the highest combined probability.

3.3. Fuzzy Conditioned Hidden Markov Model

The *fuzzy conditioned hidden Markov model* (FCHMM) extends the CHMM by replacing the sequence of discrete outer causes by a sequence of fuzzy outer causes (Glodek et al., 2014a). For this purpose, the FCHMM is equipped with an additional sequence $\mathsf{\mathbf{z}}$ of random variables. The new sequence influences the selection of the discrete causes $\mathsf{\mathbf{y}}$. Figure 3.4 depicts the graphical model of the FCHMM. Each state of the expert sequence $\mathsf{\mathbf{z}}$ encodes an output which can be provided by an external classifier system. However, since the external classifier system can theoretically have an infinite number of possible outputs, we choose $z \in \mathbb{N}$ such that a definite state can be assigned at each time step. The distribution of the observations $\mathbf{X}$ given the model λ depending on the sequence of outer causes $\mathbf{z}$ can be obtained by marginalizing over all possible causes $\mathbf{y}$:

$$\begin{aligned}p(\mathbf{X}|\mathbf{z} \shortmid \lambda) = \sum_{\mathbf{y}\in\Omega_{\mathsf{y}}^{T}} \sum_{\mathbf{w}\in\Omega_{\mathsf{w}}^{T}} & p(\mathsf{w}_1 = w_1|y_1 \shortmid \boldsymbol{\pi})\cdot \\ & \prod_{t=2}^{T} p(\mathsf{w}_t = w_t|\mathsf{w}_{t-1} = w_{t-1}, \mathsf{y}_t = y_t \shortmid \mathbf{A})\cdot \\ & \prod_{t=1}^{T} p(\mathsf{y}_t = y_t|z_t \shortmid \mathbf{B}) p(\mathbf{x}_t|\mathsf{w}_t = w_t \shortmid \boldsymbol{\theta}).\end{aligned} \tag{3.42}$$

The distribution $p(\mathsf{y}_t = y_t|z_t \shortmid \mathbf{B})$ maps an expert state z_t to the outer cause y_t. The sequence of expert states $\mathbf{z}$ can be represented as a vector of states. However, since each state z_t encodes a probability distribution over the random variable y_t, the probability distribution is directly stored in the matrix $\mathbf{B}$ of the size $|\Omega_{\mathsf{y}}| \times T$. The sequence of expert $\mathbf{z}$ was introduced in order to improve the selection of hidden states. However, the degree of fuzziness which can for instance be measured by entropy has a considerable impact to the performance of the FCHMM. An expert sequence in which the fuzzy values are almost

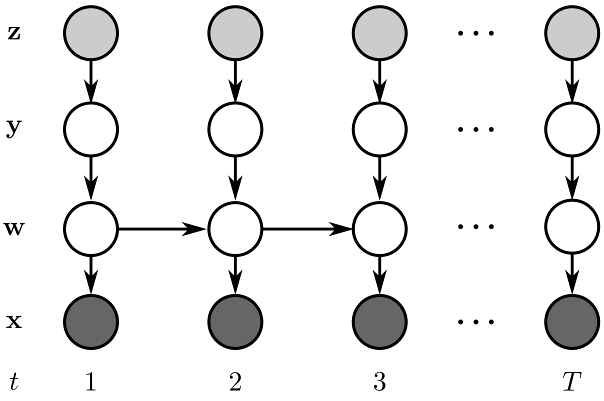


Figure 3.4. Graphical representation of the FCHMM. The Markov chain $\mathbf{w}$ is composed of a sequence of hidden random variables w_t. Each hidden random variable w_t, except the one of the first time step, depends on its predecessor w_{t-1}. Furthermore, each hidden random variable emits an observation $\mathbf{x}_t$. The selection of the hidden state w_t is influenced by the predecessors hidden state w_{t-1} and by the outer cause y_t. An expert state z_t influences the outer cause y_t where each expert state encodes a unique probability distribution of y_t.

uniformly distributed will most probably lower the performance of the FCHMM. An almost crisp expert sequence which is labeled inaccurately will let the model converge to wrong parameters. One reason for this sensitive behavior is that the fuzziness is intensified by accumulating the values over the sequence. Hence, it is advisable to additionally control or learn the level of fuzziness, for instance by raising all fuzzy values by the power of p and a subsequent normalization.

3.3.1. Inference

Forward algorithm

The forward-backward algorithm of the FCHMM extends the forward-backward algorithm of the CHMM by the new expert sequence $\mathbf{z}$ and, thus, additionally marginalizes over the sequence of outer causes $\mathbf{y}$. The forward variable $\alpha_{t,\mathbf{z}}(j)$ for the hidden state j and time step t is given by

$$\alpha_{t,\mathbf{z}}(j) = p(\mathbf{X}_{1..t}, \mathsf{w}_t = j|\mathbf{z}_{1..t}) \tag{3.43}$$

$$= \sum_{\mathbf{y_{1..t}} \in \Omega_{\mathsf{y}}^t} p(\mathbf{x}_t|\mathsf{w}_t = j) p(\mathbf{X}_{1..t-1}, \mathsf{w}_t = j|\mathbf{y}_{1..t}) p(\mathbf{y}_{1..t}|\mathbf{z}_{1..t}) \tag{3.44}$$

$$= \sum_{\mathbf{y_{1..t}} \in \Omega_{\mathsf{y}}^t} p(\mathbf{x}_t|\mathsf{w}_t = j) \cdot \sum_{i \in \Omega_{\mathsf{w}}} p(\mathbf{X}_{1..t-1}, \mathsf{w}_{t-1} = i, \mathsf{w}_t = j|\mathbf{y}_{1..t}) p(\mathbf{y}_{1..t}|\mathbf{z}_{1..t}) \tag{3.45}$$

$$= \sum_{\mathbf{y_{1..t-1}} \in \Omega_{\mathsf{y}}^{t-1}} p(\mathbf{x}_t|\mathsf{w}_t = j) \cdot \sum_{i \in \Omega_{\mathsf{w}}} p(\mathbf{X}_{1..t-1}, \mathsf{w}_{t-1} = i|\mathbf{y}_{1..t-1}) p(\mathbf{y}_{1..t-1}|\mathbf{z}_{1..t-1}) \cdot \sum_{k \in \Omega_{\mathsf{y}}} p(\mathsf{w}_t = j|\mathsf{w}_{t-1} = i, \mathsf{y}_t = k) p(\mathsf{y}_t = k|z_t) \tag{3.46}$$

$$= p(\mathbf{x}_t|\mathsf{w}_t = j) \sum_{i \in \Omega_{\mathsf{w}}} \alpha_{t-1}(i) \cdot \sum_{k \in \Omega_{\mathsf{y}}} p(\mathsf{w}_t = j|\mathsf{w}_{t-1} = i, \mathsf{y}_t = k) p(\mathsf{y}_t = k|z_t) \tag{3.47}$$

$$
\begin{aligned}
&= \sum_{k\in\Omega_{\mathsf{y}}} p(\mathbf{x}_t|\mathsf{w}_t=j)\cdot \\
&\qquad \sum_{i\in\Omega_{\mathsf{w}}} \alpha_{t-1,\mathbf{z}}(i)p(\mathsf{w}_t=j|\mathsf{w}_{t-1}=i,\mathsf{y}_t=k)p(\mathsf{y}_t=k|z_t) && (3.48)\\
&= \sum_{k\in\Omega_{\mathsf{y}}} \widehat{\alpha}_{t,\mathbf{z}}(j,k). && (3.49)
\end{aligned}
$$

A new variable $\widehat{\alpha}_{t,\mathbf{z}}(j,k)$ is introduced to keep the notation uncluttered. It auxiliary forward variable additionally depends on the state k of the causal random variable y_t:

$$
\begin{aligned}
\widehat{\alpha}_{t,\mathbf{z}}(j,k) &= p(\mathbf{X}_{1..t},\mathsf{w}_t=j,\mathsf{y}_t=k|\mathbf{z}_{1..t}) && (3.50)\\
&= p(\mathbf{x}_t|\mathsf{w}_t=j)\sum_{i\in\Omega_{\mathsf{w}}} \alpha_{t-1,\mathbf{z}}(i)p(\mathsf{w}_t=j|\mathsf{w}_{t-1}=i,\mathsf{y}_t=k)\cdot \\
&\qquad p(\mathsf{y}_t=k|z_t). && (3.51)
\end{aligned}
$$

Backward algorithm

The backward variable $\beta_{t-1,\mathbf{z}}(j)$ of the hidden state j and time step $t-1$ is analogously extended:

$$
\begin{aligned}
\beta_{t-1,\mathbf{z}}(j) &= p(\mathbf{X}_{t..T}|\mathsf{w}_{t-1}=j,\mathbf{z}_{t..T}) && (3.52)\\
&= \sum_{\mathbf{y_{t..T}}\in\Omega_{\mathsf{y}}^{T-t+1}} p(\mathbf{X}_{t..T}|\mathsf{w}_{t-1}=j,\mathbf{y}_{t..T})p(\mathbf{y}_{t..T}|\mathbf{z}_{t..T}) && (3.53)\\
&= \sum_{\mathbf{y_{t..T}}\in\Omega_{\mathsf{y}}^{T-t+1}} \sum_{i\in\Omega_{\mathsf{w}}} p(\mathbf{X}_{t..T},\mathsf{w}_t=i|\mathsf{w}_{t-1}=j,\mathbf{y}_{t..T})p(\mathbf{y}_{t..T}|\mathbf{z}_{t..T}) && (3.54)\\
&= \sum_{\mathbf{y_{t..T}}\in\Omega_{\mathsf{y}}^{T-t}} \sum_{i\in\Omega_{\mathsf{w}}} p(\mathbf{X}_{(t+1)..T}|\mathsf{w}_t=i,\mathbf{y}_{(t+1)..T})\cdot \\
&\qquad p(\mathbf{y}_{(t+1)..T}|\mathbf{z}_{(t+1)..T})p(\mathbf{x}_t|\mathsf{w}_t=i)\cdot \\
&\qquad \sum_{k\in\Omega_{\mathsf{y}}} p(\mathsf{w}_t=i|\mathsf{w}_{t-1}=j,\mathsf{y}_t=k)p(\mathsf{y}_t=k|z_t) && (3.55)\\
&= \sum_{i\in\Omega_{\mathsf{w}}} \beta_{t,\mathbf{z}}(i)p(\mathbf{x}_t|\mathsf{w}_t=i)\cdot \\
&\qquad \sum_{k\in\Omega_{\mathsf{y}}} p(\mathsf{w}_t=i|\mathsf{w}_{t-1}=j,\mathsf{y}_t=k)p(\mathsf{y}_t=k|z_t) && (3.56)\\
&= \sum_{k\in\Omega_{\mathsf{y}}} \widehat{\beta}_{t-1,\mathbf{z}}(j,k)p(\mathsf{y}_t=k|z_t). && (3.57)
\end{aligned}
$$

Again an auxiliary variable is introduced to keep the notation uncluttered. The new variable additionally depends on the state k of the causal random variable y_t:

$$
\begin{aligned}
\widehat{\beta}_{t-1,\mathbf{z}}(j,k) &= p(\mathbf{X}_{t..T}|\mathsf{w}_{t-1}=j,\mathsf{y}_t=k) && (3.58)\\
&= \sum_{i\in\Omega_{\mathsf{w}}} \beta_{t,\mathbf{z}}(i)p(\mathbf{x}_t|\mathsf{w}_t=i)p(\mathsf{w}_t=i|\mathsf{w}_{t-1}=j,\mathsf{y}_t=k). && (3.59)
\end{aligned}
$$

The initial and terminal values of the forward and backward recursions are given by

$$\begin{aligned}
\widehat{\alpha}_{1,\mathbf{z}}(j,k) &= \sum_{l\in\Omega_{\mathsf{z}}} p(\mathbf{x}_1|\mathsf{w}_1=j)p(\mathsf{w}_1=j|\mathsf{y}_1=k)p(\mathsf{y}_1=k|\mathsf{z}_1=z_1) && (3.60)\\
\widehat{\beta}_{T,\mathbf{z}}(j,k) &= 1 && (3.61)\\
\beta_{T,\mathbf{z}}(j) &= 1 && (3.62)\\
\widehat{\beta}_{0,\mathbf{z}}(j,k) &= \sum_{i\in\Omega_{\mathsf{w}}} \beta_{1,\mathbf{z}}(j)p(\mathbf{x}_1|\mathsf{w}_1=i)p(\mathsf{w}_1=i|\mathsf{y}_1=k). && (3.63)
\end{aligned}$$

The responsibilities, i.e. the state probabilities of being in the hidden state j at time step t given the complete sequences $\mathbf{X}$ and $\mathbf{z}$, can be determined using the forward and backward variables:

$$\begin{aligned}
\gamma_{t,\mathbf{z}}(j) &= p(\mathsf{w}_t=j|\mathbf{X},\mathbf{z}) && (3.64)\\
&= \frac{\alpha_{t,\mathbf{z}}(j)\beta_{t,\mathbf{z}}(j)}{p(\mathbf{X})}. && (3.65)
\end{aligned}$$

The transition probability of passing over from state i to state j at the time steps $t-1$ to t given the complete sequences $\mathbf{X}$ and $\mathbf{z}$ can be evaluated by

$$\begin{aligned}
\xi_{t-1,t,\mathbf{z}}(i,j) &= p(\mathsf{w}_{t-1}=i,\mathsf{w}_t=j|\mathbf{X},\mathbf{z}) && (3.66)\\
&= \frac{p(\mathbf{X}|\mathsf{w}_{t-1}=i,\mathsf{w}_t=j,\mathbf{z})p(\mathsf{w}_{t-1}=i,\mathsf{w}_t=j)}{p(\mathbf{X})} && (3.67)\\
&= \frac{1}{p(\mathbf{X})}p(\mathbf{X}_{1..t-1}|\mathsf{w}_{t-1}=i,\mathbf{z}_{1..t-1})p(\mathbf{x}_t|\mathsf{w}_t=j)\cdot\\
&\quad p(\mathbf{X}_{t+1..T}|\mathsf{w}_t=j,\mathbf{z}_{t+1..T})p(\mathsf{w}_t=j|\mathsf{w}_{t-1}=i)p(\mathsf{w}_{t-1}=i) && (3.68)\\
&= \frac{1}{p(\mathbf{X})}\alpha_{t-1,\mathbf{z}}(i)p(\mathbf{x}_t|\mathsf{w}_t=j)\cdot\\
&\quad \left(\sum_{k\in\Omega_{\mathsf{y}}} p(\mathsf{w}_t=j|\mathsf{w}_{t-1}=i,\mathsf{y}_t=k)p(\mathsf{y}_t=k|z_t)\right)\beta_{t,\mathbf{z}}(j). && (3.69)
\end{aligned}$$

The responsibility $\gamma_{t,\mathbf{z}}(j)$ can be used to derive the parameters θ_j of the observations emitted by hidden state j. However, the transition probability $\xi_{t-1,t,\mathbf{z}}(i,j)$ cannot be utilized in this form, because the transition matrix additionally depends on the causal output $\mathsf{y}_t=k$. Therefore, two additional probabilities are needed. The extended state probability given by

$$\begin{aligned}
\widehat{\gamma}_{t,\mathbf{z}}(j,k) &= p(\mathsf{w}_t=j,\mathsf{y}_t=k|\mathbf{X},\mathbf{z}) && (3.70)\\
&= \frac{p(\mathbf{X}|\mathsf{w}_t=j,\mathsf{y}_t=k,\mathbf{z})p(\mathsf{w}_t=j,\mathsf{y}_t=k)}{p(\mathbf{X})} && (3.71)\\
&= \frac{p(\mathbf{X}_{1..t},\mathsf{w}_t=j,\mathsf{y}_t=k|\mathbf{z}_{1..t})p(\mathbf{X}_{t+1..T}|\mathsf{w}_t=j,\mathbf{z}_{t+1..T})}{p(\mathbf{X})} && (3.72)\\
&= \frac{\widehat{\alpha}_{t,\mathbf{z}}(j,k)\beta_{t,\mathbf{z}}(j)}{p(\mathbf{X})} && (3.73)
\end{aligned}$$

and the extended transition probability given by

$$\begin{aligned}
\widehat{\xi}_{t-1,t,\mathbf{z}}(i,j,k) &= p(\mathsf{w}_{t-1}=i,\mathsf{w}_t=j,\mathsf{y}_t=k|\mathbf{X},\mathbf{z}) && (3.74)\\
&= \frac{p(\mathbf{X}|\mathsf{w}_{t-1}=i,\mathsf{w}_t=j,\mathsf{y}_t=k,\mathbf{z})p(\mathsf{w}_{t-1}=i,\mathsf{w}_t=j,\mathsf{y}_t=k)}{p(\mathbf{X})} && (3.75)\\
&= \frac{1}{p(\mathbf{X})}p(\mathbf{X}_{1..t-1}|\mathsf{w}_{t-1}=i,\mathbf{z}_{1..t-1})p(\mathbf{x}_t|\mathsf{w}_t=j)\cdot\\
&\quad p(\mathsf{w}_t=j|\mathsf{w}_{t-1}=i,\mathsf{y}_t=k)p(\mathsf{w}_{t-1}=i,\mathsf{y}_t=k)\cdot\\
&\quad p(\mathbf{X}_{t+1..T}|\mathsf{w}_t=j,\mathbf{z}_{t+1..T}) && (3.76)\\
&= \frac{1}{p(\mathbf{X})}\alpha_{t-1,\mathbf{z}}(i)p(\mathbf{x}_t|\mathsf{w}_t=j)\cdot\\
&\quad p(\mathsf{w}_t=j|\mathsf{w}_{t-1}=i,\mathsf{y}_t=k)\Big(\sum_{l=1}^{K}p(\mathsf{y}_t=k|z_t)\Big)\beta_{t,\mathbf{z}}(j). && (3.77)
\end{aligned}$$

The extended responsibility $\widehat{\gamma}_{t,\mathbf{z}}(j,k)$ together with the former transition probability $\xi_{t-1,t,\mathbf{z}}(i,j)$ can be utilized to maximize the parameters in case a independence assumption is made on the transition matrix, i.e. $p(\mathsf{w}_t\!=\!i|\mathsf{w}_{t-1}\!=\!j,\mathsf{y}_t\!=\!k,\mathbf{A}) = p(\mathsf{w}_t\!=\!i|\mathsf{w}_{t-1}\!=\!j,\mathbf{A}_1)\cdot p(\mathsf{w}_t\!=\!i|\mathsf{y}_t\!=\!k,\mathbf{A}_2)$. In the original setting, the three-dimensional transition matrix can be derived using the extended transition probability $\widehat{\xi}_{t-1,t,\mathbf{z}}(i,j,k)$.

The likelihood of an observation sequence $\hat{\mathbf{X}}$ given a sequence of experts $\mathsf{\mathbf{z}}$ is given by

$$p(\hat{\mathbf{X}}|\mathsf{\mathbf{z}}) = \sum_{i\in\Omega_{\mathsf{w}}}\alpha_{T,\mathbf{z}}(i). \tag{3.78}$$

The exact inference to obtain the distribution over the random variable $\mathsf{\mathbf{y}}$ and $\mathsf{\mathbf{w}}$ can be computed using $\widehat{\gamma}_{t,\mathbf{z}}(j,k)$ by providing a prior of uniform distributed causal states. The probability distribution of the random variable $\mathsf{\mathbf{y}}$ is then obtained by marginalizing over the hidden states.

3.3.2. Parameter Learning

EM-algorithm

Parameter learning is realized using again the EM-algorithm. The expected complete data log-likelihood of the FCHMM is defined by

$$Q(\lambda,\lambda^{old}) = \sum_{n=1}^{N}\sum_{\mathbf{w}\in\Omega_{\mathsf{w}}^{T}}\sum_{\mathbf{y}\in\Omega_{\mathsf{y}}^{T}} p(\mathbf{w},\mathbf{y}|\mathbf{X},\mathbf{z}\shortmid\lambda^{old})\ln p(\mathbf{X},\mathbf{w},\mathbf{y}|\mathbf{z}\shortmid\lambda). \tag{3.79}$$

In contrast to the parameter learning of the CHMM, an additional marginalization over the causal states is required. Substituting the model parameters and the responsibilities and transition probabilities results in:

$$
\begin{aligned}
Q(\lambda, \lambda^{old}) = & \sum_{n=1}^{N} \sum_{\mathbf{w} \in \Omega_{\mathsf{w}}^{T}} \sum_{\mathbf{y} \in \Omega_{\mathsf{y}}^{T}} p(\mathbf{w}, \mathbf{y} | \mathbf{X}, \mathbf{z} \shortmid \lambda^{old}) \\
& \ln \Bigg(p(\mathsf{w}_1 = w_1 | \mathsf{y}_1 = y_1 \shortmid \boldsymbol{\pi}) p(\mathsf{y}_1 = y_1 | z_1 \shortmid \mathbf{B}) \cdot \\
& \prod_{t=2}^{T} p(\mathsf{w}_t = w_t | \mathsf{w}_{t-1} = w_{t-1}, \mathsf{y}_1 = y_1, \mathbf{A}) p(\mathsf{y}_t = y_t | z_t \shortmid \mathbf{B}) \cdot \\
& \prod_{t=1}^{T} p(\mathbf{x}_t = \mathbf{x}_t | \mathsf{w}_t = w_t \shortmid \boldsymbol{\theta}) \Bigg) \qquad (3.80) \\
= & \sum_{n=1}^{N} \sum_{w_1 \in \Omega_{\mathsf{w}}} \sum_{y_1 \in \Omega_{\mathsf{y}}} \widehat{\gamma}_{1,\mathbf{z}}(w_1, y_1) \ln \pi_{w_1, y_1} p(\mathsf{y}_1 = y_1 | z_1 \shortmid \mathbf{B}) + \\
& \sum_{t=2}^{T} \sum_{w_{t-1} \in \Omega_{\mathsf{w}}} \sum_{w_t \in \Omega_{\mathsf{w}}} \sum_{y_t \in \Omega_{\mathsf{y}}} \widehat{\xi}_{t-1,t,\mathbf{z}}(w_{t-1}, w_t, y_t) \cdot \\
& \ln a_{w_{t-1}, w_t, y_t} p(\mathsf{y}_t = y_t | z_t \shortmid \mathbf{B}) + \\
& \sum_{t=1}^{T} \sum_{w_t \in \Omega_{\mathsf{w}}} \gamma_{t,\mathbf{z}}(w_t) \ln p(\mathbf{x}_t | \mathsf{w}_t = w_t \shortmid \boldsymbol{\theta}). \qquad (3.81)
\end{aligned}
$$

First, the update formula of the parameter π_{i,y_1} which models the initial hidden state probability will be derived. Adding the constraint with the Lagrange multiplier κ and taking the derivative with respect to the target parameter, as well as setting the outcome equal to zero leads to

$$
\begin{aligned}
\frac{\partial Q(\lambda, \lambda^{\text{old}})}{\partial \pi_{w_1, y_1}} = & \frac{\partial}{\partial \pi_{w_1, y_1}} \sum_{j \in \Omega_{\mathsf{w}}} \sum_{k \in \Omega_{\mathsf{y}}} \widehat{\gamma}_{1,\mathbf{z}^{(n)}}(j, k) \ln \pi_{j,k} b_{k, z_1^{(n)}} - \\
& \sum_{k \in \Omega_{\mathsf{y}}} \kappa \Bigg(\sum_{j \in \Omega_{\mathsf{w}}} \pi_{j,k} - 1 \Bigg) = 0 \qquad (3.82) \\
= & \frac{\widehat{\gamma}_{1,\mathbf{z}}(w_1, y_1)}{\pi_{w_1, y_1} b_{y_1, z_1^{(n)}}} - \kappa = 0 \qquad (3.83) \\
= & \widehat{\gamma}_{1,\mathbf{z}^{(n)}}(w_1, y_1) - b_{y_1, z_1^{(n)}} \pi_{w_1, y_1} \kappa = 0. \qquad (3.84)
\end{aligned}
$$

Expanding the expression and rearranging the equation to find the solution for the Lagrange multiplier κ results in:

$$
\begin{aligned}
& \Rightarrow \sum_{j \in \Omega_{\mathsf{w}}} \widehat{\gamma}_{1,\mathbf{z}^{(n)}}(j, y_1) - b_{y_1, z_1^{(n)}} \pi_{j, y_1} \kappa = 0 \qquad (3.85) \\
& \Leftrightarrow \qquad \kappa \sum_{j \in \Omega_{\mathsf{w}}} b_{y_1, z_1^{(n)}} \pi_{j, y_1} = \sum_{j \in \Omega_{\mathsf{w}}} \widehat{\gamma}_{1,\mathbf{z}^{(n)}}(j, y_1) \qquad (3.86) \\
& \Leftrightarrow \qquad \kappa = \frac{\sum_{j \in \Omega_{\mathsf{w}}} \widehat{\gamma}_{1,\mathbf{z}^{(n)}}(j, y_1)}{b_{y_1, z_1^{(n)}}}. \qquad (3.87)
\end{aligned}
$$

The outcome is substituted back into Equation 3.83 which is then rearranged to obtain the target parameter:

$$\frac{\widehat{\gamma}_{1,\mathbf{z}^{(n)}}(w_1, y_1)}{\pi_{w_1,y_1} b_{y_1,z_1^{(n)}}} - \kappa = 0 \tag{3.88}$$

$$\Leftrightarrow \frac{\widehat{\gamma}_{1,\mathbf{z}^{(n)}}(w_1, y_1)}{\pi_{w_1,y_1} b_{y_1,z_1^{(n)}}} - \frac{\sum_{j\in\Omega_\mathrm{w}} \widehat{\gamma}_{1,\mathbf{z}^{(n)}}(j, y_1)}{b_{y_1,z_1^{(n)}}} = 0 \tag{3.89}$$

$$\Leftrightarrow \qquad \frac{\pi_{w_1,y_1} b_{y_1,z_1^{(n)}}}{\widehat{\gamma}_{1,\mathbf{z}^{(n)}}(w_1, y_1)} = \frac{b_{y_1,z_1^{(n)}}}{\sum_{j\in\Omega_\mathrm{w}} \widehat{\gamma}_{1,\mathbf{z}^{(n)}}(j, y_1)} \tag{3.90}$$

$$\Leftrightarrow \qquad \pi_{w_1,y_1} = \frac{\widehat{\gamma}_{1,\mathbf{z}^{(n)}}(w_1, y_1)}{\sum_{j\in\Omega_\mathrm{w}} \widehat{\gamma}_{1,\mathbf{z}^{(n)}}(j, y_1)}. \tag{3.91}$$

Considering the complete dataset, the update formula of the initial parameter is given by

$$\pi_{w_1,y_1} = \frac{\sum_{n=1}^{N} \delta_{y_1^{(n)}=y_1} \widehat{\gamma}_{1,\mathbf{z}^{(n)}}(w_1, y_1)}{\sum_{n=1}^{N} \sum_{j\in\Omega_\mathrm{w}} \delta_{y_1^{(n)}=y_1} \widehat{\gamma}_{1,\mathbf{z}^{(n)}}(j, y_1)}. \tag{3.92}$$

Next, the update formula for the state transition probability a_{ijk} will be derived. In the first step, a regularization constraint with a corresponding Lagrange multiplier κ is added. Subsequently, the derivative of the formula given the target parameter is taken. The outcome is set equal zero and rearranged:

$$\begin{aligned} \frac{\partial Q(\lambda, \lambda^{\mathrm{old}})}{\partial a_{v,w,y}} = \frac{\partial}{\partial a_{v,w,y}} \sum_{t=2}^{T} \sum_{i\in\Omega_\mathrm{w}} \sum_{j\in\Omega_\mathrm{w}} \sum_{k\in\Omega_\mathrm{y}} \widehat{\xi}_{t-1,t,\mathbf{z}^{(n)}}(i, j, k) \ln a_{i,j,k} b_{k,z_t^{(n)}} - \\ \sum_{k\in\Omega_\mathrm{y}} \sum_{i\in\Omega_\mathrm{w}} \kappa \Big(\sum_{j\in\Omega_\mathrm{w}} a_{ijk} - 1 \Big) = 0 \end{aligned} \tag{3.93}$$

$$\Leftrightarrow \qquad \sum_{t=2}^{T} \frac{\widehat{\xi}_{t-1,t,\mathbf{z}}(v, w, y)}{a_{v,w,y} b_{y,z_t^{(n)}}} - \kappa = 0 \tag{3.94}$$

$$\Leftrightarrow \qquad \sum_{t=2}^{T} \widehat{\xi}_{t-1,t,\mathbf{z}^{(n)}}(v, w, y) - a_{v,w,y} b_{y,z_t^{(n)}} \kappa = 0. \tag{3.95}$$

The equation is then extended by a sum over all hidden states in order to determine the Lagrange multiplier κ to obtain:

$$\sum_{t=2}^{T} \sum_{j\in\Omega_\mathrm{w}} \widehat{\xi}_{t-1,t,\mathbf{z}^{(n)}}(v, j, y) - a_{v,j,y} b_{y,z_t^{(n)}} \kappa =0 \tag{3.96}$$

$$\Leftrightarrow \qquad \sum_{t=2}^{T} b_{y,z_t^{(n)}} \sum_{j\in\Omega_\mathrm{w}} a_{v,j,y} \kappa = \sum_{t=2}^{T} \sum_{j\in\Omega_\mathrm{w}} \widehat{\xi}_{t-1,t,\mathbf{z}^{(n)}}(v, j, y) \tag{3.97}$$

$$\Leftrightarrow \qquad \kappa = \sum_{t=2}^{T} \frac{\sum_{j\in\Omega_\mathrm{w}} \widehat{\xi}_{t-1,t,\mathbf{z}^{(n)}}(v, j, y)}{b_{y,z_t^{(n)}}}. \tag{3.98}$$

Substituting the outcome back into Equation 3.94 and further rearranging yields in the update formula:

$$\Rightarrow \sum_{t=2}^{T} \frac{\widehat{\xi}_{t-1,t,\mathbf{z}^{(n)}}(v,w,y)}{a_{v,w,y} b_{y_t, z_t^{(n)}}} - \kappa = 0 \tag{3.99}$$

$$\Leftrightarrow \qquad 0 = \sum_{t=2}^{T} \frac{\widehat{\xi}_{t-1,t,\mathbf{z}^{(n)}}(v,w,y)}{a_{v,w,y} b_{y, z_t^{(n)}}} - \frac{\sum_{j \in \Omega_{\mathsf{w}}} \widehat{\xi}_{t-1,t,\mathbf{z}^{(n)}}(v,j,y)}{b_{y, z_t^{(n)}}} \tag{3.100}$$

$$\Leftrightarrow \qquad a_{v,w,y} = \frac{\sum_{t=2}^{T} \widehat{\xi}_{t-1,t,\mathbf{z}^{(n)}}(v,w,y)}{\sum_{t=2}^{T} \sum_{j \in \Omega_{\mathsf{w}}} \widehat{\xi}_{t-1,t,\mathbf{z}^{(n)}}(v,j,y)}. \tag{3.101}$$

The final update formula on the complete dataset is then given by

$$a_{v,w,y} = \frac{\sum_{n=1}^{N} \sum_{t=2}^{T} \delta_{y_t^{(n)}=y} \widehat{\xi}_{t-1,t,\mathbf{z}^{(n)}}(v,w,y)}{\sum_{n=1}^{N} \sum_{t=2}^{T} \sum_{j \in \Omega \mathsf{w}} \delta_{y_t^{(n)}=y} \widehat{\xi}_{t-1,t,\mathbf{z}^{(n)}}(v,j,y)}. \tag{3.102}$$

Analogously to the HMM and CHMM, an observation $\mathbf{x}_t$ depends only on the corresponding hidden stat w_t. Therefore, all observation models presented for the HMM can also be applied to the FCHMM without any modifications.

3.4. HMM Using Graph Probability Densities

The GPD introduced in the previous chapter can be used to fit densities to small variable-sized labeled and weighted graphs. This section describes how GPD can be integrated into an HMM (Glodek et al., 2013d). The resulting *HMM using GPD* (HMM-GPD) cannot only process sequential patterns but also observations of variable length.

We begin with defining a graph observation $\mathbf{x}_t = \{(\mathring{x}_{mt}, \tilde{\mathbf{x}}_{mt})\}_{m=1}^{M}$ for a time step t. The discrete element $\mathring{x}_{mt}$ encodes an edge which, by definition, can occur only once in a time step. The continuous vector $\tilde{\mathbf{x}}_{mt}$ contains the weights assigned to the edge and its vertices. Each hidden state i is associated with a GPD model θ_i:

$$p(\mathbf{x}_t | \mathsf{w} = i \shortmid \boldsymbol{\theta}) = \prod_{m=1}^{M} p(\mathring{x}_{mt}, \tilde{\mathbf{x}}_{mt} | \mathsf{w} = i \shortmid \boldsymbol{\theta}) \tag{3.103}$$

$$= \prod_{m=1}^{M} \sum_{k=1}^{K} \nu_{i\mathring{x}_{mt}k} \mathcal{N}(\tilde{\mathbf{x}}_{mt} \shortmid \boldsymbol{\mu}_{i\mathring{x}_{m}k}, \boldsymbol{\Sigma}_{i\mathring{x}_{mt}k}). \tag{3.104}$$

To update the parameters of the GPD in the M-step, the GPD has to be placed into the complete-data log-likelihood of the HMM given by Equation 2.81. The parameters are maximized by taking the derivative and the setting the expression equal to zero:

$$\frac{\partial Q^{(n)}(\lambda, \lambda^{\text{old}})}{\partial \theta_i} = \frac{\partial}{\partial \theta_i} \sum_{n=1}^{N} \sum_{t=1}^{T} \sum_{j \in \mathsf{w}_t} \gamma_t^{(n)}(j) \ln \prod_{m=1}^{M} p(\mathring{x}_{mt}^{(n)}, \tilde{\mathbf{x}}_{mt}^{(n)} | \mathsf{w} = j \shortmid \boldsymbol{\theta}) \tag{3.105}$$

$$= \sum_{n=1}^{N}\sum_{t=1}^{T}\gamma_t^{(n)}(i)\sum_{m=1}^{M}\frac{\partial}{\partial\theta_i}\ln p(\mathring{x}_{mt}^{(n)}, \tilde{\mathbf{x}}_{mt}^{(n)}|\mathsf{w}=i \shortmid \boldsymbol{\theta}) = 0. \tag{3.106}$$

The final update formulas are derived analogously as in Section 2.4. Hence, the update formula for the mixing variables is given by

$$\nu_{i\mathring{x}l} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{T}\sum_{m=1}^{M}\gamma_t^{(n)}(i)\delta_{\mathring{x}_{mt}^{(n)}=\mathring{x}}\widetilde{\gamma}_t^{(n)}(i,\mathring{x},l)\nu_{i\mathring{x}l}}{\sum_{n=1}^{N}\sum_{t=1}^{T}\sum_{m=1}^{M}\sum_{k=1}^{K}\gamma_t^{(n)}(i)\delta_{\mathring{x}_{mt}^{(n)}=\mathring{x}}\widetilde{\gamma}_t^{(n)}(i,\mathring{x},k)\nu_{i\mathring{x}k}} \tag{3.107}$$

and the updates formula for the means and the covariance matrices are given by

$$\boldsymbol{\mu}_{i\mathring{x}l} = \frac{\sum\limits_{n=1}^{N}\sum\limits_{t=1}^{T}\sum\limits_{m=1}^{M}\gamma_t^{(n)}(i)\delta_{\mathring{x}_{tm}^{(n)}=\mathring{x}}\widetilde{\gamma}_t^{(n)}(i,\mathring{x},l)\tilde{\mathbf{x}}_{tm}^{(n)}}{\sum\limits_{t=1}^{T}\sum\limits_{n=1}^{N}\sum\limits_{m=1}^{M}\gamma_t^{(n)}(i)\delta_{\mathring{x}_{tm}^{(n)}=\mathring{x}}\widetilde{\gamma}_t^{(n)}(i,l,\mathring{x})} \tag{3.108}$$

and

$$\boldsymbol{\Sigma}_{i\mathring{x}l} = \frac{\sum\limits_{n=1}^{N}\sum\limits_{t=1}^{T}\sum\limits_{m=1}^{M}\gamma_t^{(n)}(i)\delta_{\mathring{x}_{tm}^{(n)}=\mathring{x}}\widetilde{\gamma}_t^{(n)}(i,\mathring{x},l)(\tilde{\mathbf{x}}_{tm}^{(n)}-\boldsymbol{\mu}_{i\mathring{x}l})(\tilde{\mathbf{x}}_{tm}^{(n)}-\boldsymbol{\mu}_{i\mathring{x}l})^{\mathrm{T}}}{\sum\limits_{t=1}^{T}\sum\limits_{n=1}^{N}\sum\limits_{m=1}^{M}\gamma_t^{(n)}(i)\delta_{\mathring{x}_{tm}^{(n)}=\mathring{x}}\widetilde{\gamma}_t^{(n)}(i,l,\mathring{x})}. \tag{3.109}$$

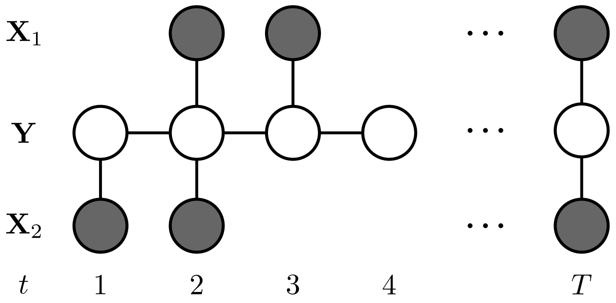
Similar to the GPD a sufficient number of samples for each edge and hidden state is required for training in order to set up sufficient statistics. This requirement can be reduced with the help of CHMM which allows the sharing of hidden states.

3.5. Markov Fusion Network

Real-world applications using pattern recognition have demanding requirements such as real-time processing, robustness against noise and failure safety. These requirements are often addressed by making use of late fusion MCS. However, there are only a few late fusion approaches which explicitly address these requirements in a unified way.

Pattern recognition algorithms must provide their outputs in time to allow succeeding algorithms to process their predictions. Therefore, all parts of the pattern recognition architecture need to be capable of real-time processing. Furthermore, the classifier architecture has to be as robust as possible to noisy inputs. Mostly because the sensor inputs are derived under less restricted conditions. For instance, there might be interfering signals in the audio channel or changing illumination settings in the video channel. In addition, the system must be able to handle sensor failures which may not only be caused by a sensor malfunction, but by the absence of a sensor signal, e.g. silence instead of an utterance or a person who has turned away instead of facing the camera. The issue of robustness is usually addressed by combining multiple modalities using late fusion. However, real-world scenarios can also

Figure 3.5. Graphical representation of the MFN. The sequence of combined estimates $\mathbf{y}_t$ is influenced by the available decisions $\mathbf{X}_{mt}$ of the source m and $t \in \mathcal{L}_m$ and adjacent combined estimates $\mathbf{y}_{t-1}$ and $\mathbf{y}_{t+1}$.



bring the advantage to perform an additional temporal fusion. In multimodal and temporal fusion, the main challenge is to deal with contradicting cues from different sources (Schels et al., 2012b; Palm and Glodek, 2013).

The *Markov fusion network* (MFN) is a novel late fusion algorithm which addresses these requirements (Glodek et al., 2014c; Siegert et al., 2013; Schels et al., 2013b; Krell et al., 2012; Glodek et al., 2012c,b). The MFN combines uncertain decisions over time provided by multiple classifiers. The classifier decisions are temporally connected by a Markov chain in order to smooth out outliers and to reconstruct missing decisions. The output is given by a combined decision stream which is calculated based on multiple input decision streams, e.g. from different modalities. The MFN is controlled by parameters which weight the different decision streams and the similarity of decisions over time. Furthermore, the approach allows the use of multinomial input distributions. The combined output decision stream of the MFN is conform with the axioms of probability theory.

The input to the MFN is given by a number of $M \times T$ probability distributions over I classes where M denotes the number of classifiers providing decisions and T is the number of time steps. In other words, for each time step a classifier provides a class distribution in every modality. The probability distribution of a classifier $m = 1, \ldots, M$ at time step $t \in \mathcal{L}_m$ is a vector $\mathbf{x}_{mt} \in [0,1]^I$ of which the elements sum up to one. The set $\mathcal{L}_m$ contains the time steps in which probability distributions of the classifier m are available. A probability distribution may not be available in case of a sensor failure. Assuming without loss of generality that the probability distributions are available for all time steps. Then the MFN will integrate the classifier predictions $\mathbf{X}_m \in [0,1]^{I \times T}$ to a combined estimate $\mathbf{Y} \in [0,1]^{I \times T}$ by making use of two main objectives. The first objective requires that in each time step the estimated probability distribution is similar to the provided classifier decisions. The objective is realized by setting up a dependency between the estimates and the classifier decisions. The second objective is based on the assumption that the estimated probability distributions have similarities in the temporal proximity. The temporal fusion is implemented by connecting the estimates of each time step using a Markov chain. The MFN reconstructs regions which have no supporting classifier decisions by propagating information along the Markov chain.

Figure 3.5 depicts the graphical model of an exemplary MFN which integrates two sequences of classifier decisions $\mathbf{X}_1$ and $\mathbf{X}_2$ to a combined estimate $\mathbf{Y}$. For each time step, the classifier decisions are connected to the corresponding estimate. Whenever a class distribution is not available, the input node and the connecting link is omitted. The estimates themselves are temporally connected by the Markov chain.

Data potential

The MFN is defined by three potential functions, namely the data potential Ψ, the smoothness potential Φ and the distribution potential Ξ. The *data potential* Ψ is a sum over Ψ_m enforcing the estimates $\mathbf{y}_t$ to be similar to the classifier distributions $\mathbf{x}_{mt}$. The data potential is given by

$$\Psi = \sum_{m}^{M} \Psi_m = \sum_{m}^{M} \sum_{i=1}^{I} \sum_{t \in \mathcal{L}_m} k_{mt}(x_{mit} - y_{it})^2 \tag{3.110}$$

Smoothness potential

where $\mathbf{K} \in \mathbb{R}_{\geq 0}^{M \times T}$ weights the reliability of the classifier m at time step t. The second potential Φ models the Markov chain which enforces lateral smoothness. The *smoothness potential* is given by

$$\Phi = \sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{s \in N(t)} w_{\min(t,s)}(y_{it} - y_{is})^2 \tag{3.111}$$

Distributional potential

where $\mathbf{w} \in \mathbb{R}_{\geq 0}^{T-1}$ weights the differences between two adjacent nodes in the chain, or in other words, the parameter controls the strength of smoothing over time. The set $N(t)$ contains indices of the adjacent nodes for a given time step t, e.g. $N(t) := \{t-1, t+1\}$. The third potential Ξ asserts that the resulting estimate satisfies the probability axioms. The *distributional potential* is given by

$$\Xi = u \cdot \sum_{t=1}^{T} \Big(\big(1 - \sum_{i=1}^{I} y_{it}\big)^2 + \sum_{i=1}^{I} \delta_{[0 > y_{it}]} \cdot y_{it}^2 \Big) \tag{3.112}$$

where u weights the relevance of the potential and $\delta_{0 > y_{it}}$ equals one in case the condition $0 > y_{it}$ holds true and zero otherwise. For each time step, the potential enforces the estimate to sum up to one and penalizes negative values. It is possible to omit the distributional potential in case a crisp two-class classification problem is estimated using a scalar stream.

The final probability density function of the estimate $\mathbf{Y}$ and the classifier decisions $\mathbf{X}_m$ is given by

$$p(\mathbf{Y}, \mathbf{X}_1, \ldots, \mathbf{X}_M) = \frac{1}{Z} \exp\Big(-\frac{1}{2}(\Psi + \Phi + \Xi) \Big). \tag{3.113}$$

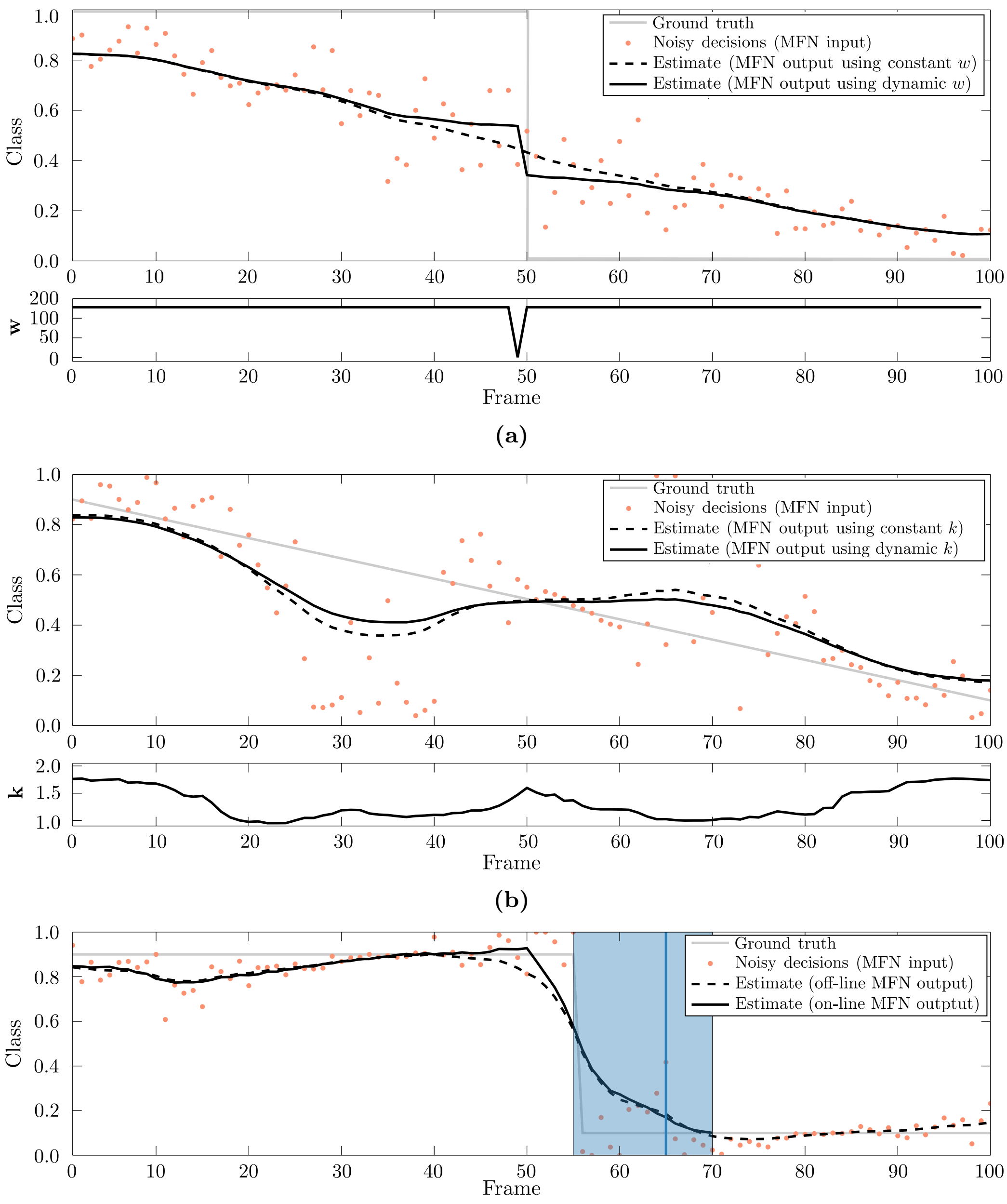
However, it is computational infeasible to determine the actual probability density because the partition function has to be computed by sampling over the complete state space. Fortunately, it is sufficient to find the mode of the density function such that the most likely estimate $\mathbf{Y}$ can be derived with only marginal computational effort.

To find the most likely sequence of $\mathbf{Y}$ for a given set of parameters, we make use of an iterative gradient descent algorithm. Each iteration of the gradient descent algorithm requires only $O(T \cdot I \cdot M)$, see Algorithm A.2. However, in large regions of missing class distributions the lateral similarity might only be propagated slowly. In these cases, it is beneficial to choose the initial values of $\mathbf{Y}$ carefully. A suitable initialization can be performed by taking the mean of the available probability masses for each time step and a linear interpolation in regions in which no distributions are available. A threshold which is based on the change of the estimate between two consecutive iteration steps can be used as a stopping criterion.

The MFN can be utilized in an offline processing mode in which a complete recording of a fixed length is computed at once. Alternatively, the MFN can be applied only to the most recent data utilizing a sliding window. In the following, we refer to this approach as the online processing mode.

Three artificially generated show cases are presented in order to demonstrate the effect of the parameters to the algorithm. Figure 3.6a depicts the influence of the parameter $\mathbf{w}$ which enforces the similarity over time. The orange dots represent a two-class distribution provided by a classifier (only one class of the probability distribution is shown), whereas the gray line is assumed to be the ground truth which shows a sudden change at the 50^{th} frame and is only marginally covered by the input data. The dashed line shows the MFN estimate using constant values for the parameter $\mathbf{w}$ which offers a close fit to the orange dots ($k_{t1} = 2$ for $t = 1 \ldots T$ and $w_t = 128$ for $t = 1 \ldots T-1$). In case additional knowledge about the class change is available, it is possible to weaken the similarity by a sudden decrease of the parameter $\mathbf{w}$. The solid black line is based on the same parameter configuration with the exception of time step $t = 49$ in which we set $w_{49} = 0.5$. The time series of the parameter $\mathbf{w}$ is shown in the lower part of the figure. The estimate based on the alternative setting is making a steep step downwards in order to minimize the energy function. Obviously, this estimate is closer to the assumed ground truth function.

The second example is shown in Figure 3.6b and demonstrates the control of the estimate using the data parameter $\mathbf{k}$. Again, the gray line represents the ground truth which is in this example given by a linear decay. The input distributions (orange dots) are locally distorted and, again, only one class of the probability distribution is depicted. The level of distortion increases depending on the distance to the ground truth function. The dashed black line shows the estimate using the configuration $k_{t1} = 2$ for $t = 1 \ldots T$ and $w_t = 128$ for $t = 1 \ldots T-1$. Hence, all classifier decisions effect the outcome with the same strength. In the given artificial setting, it is self-evident to derive classifier confidence based on the standard deviation. For this purpose, a sliding window of ten frames is utilized to create a dynamic measure which weights the input decisions using the data parameter $\mathbf{k}$. The solid



(a)

(b)

(c)

Figure 3.6. MFN show cases demonstrating the influence of different potential parameters and explaining the online and offline processing modes. (a) Influence of the smoothness parameter **w** to the estimated decisions utilizing noisy classifier decisions as input to the MFN. (b) Influence of the dynamic weighting of the parameter k_{tm} to the estimated decisions utilizing noisy classifier decisions as input to the MFN. (c) Online and offline proceesing modes using the MFN. In online processing mode, a window is shifted over the stream of classifier decisions. The MFN is applied in each window. In offline processing mode, the MFN is applied to the complete sequence at once. For a detailed description please refer to the text.

black line shows the estimate using the dynamic weighting of **k** as shown in the lower part of the figure. In regions of low variances the parameter k_{t1} takes values close to 2, whereas in regions of heavily scattered input decisions the parameter k_{t1} gets close to 1. The plot shows that the output of the MFN using a dynamical weighting is clearly less influenced by outliers. The differences between the online and offline processing modes are explained in Figure 3.6c. The dashed curve shows the estimate of the MFN applied to the complete sequence at once, i.e. offline processing mode. The solid curve shows the online processing mode which is performed by determining the estimate using a sliding window of 15 frames, as indicated by the filled light blue square. For each frame, the MFN is fitted to the decisions covered by the respective window using the initialization of the preceding window. If a slight delay is in accordance with the specification of the application, it is advisable to refrain from the estimate of the last time step as the final result since an estimated value being placed in the middle of the window is less sensitive to outliers. In the figure we utilized the estimate at frame number ten within the window which is indicated by the vertical blue line.

The MFN offers a large set of advantageous properties: (1) It models the temporal relationship between probabilistic classifier decisions; (2) it combines multiple classifier decisions from different modalities or views and is able to weight them dynamically; (3) it handles missing classifier decisions, e.g. due to sensor failures; (4) the estimate follows the probability axioms; and (5) it can easily be deployed in real-time scenarios.

3.5.1. Parameter Learning

This section presents an MFN learning procedure which can be used to derive the optimal assignments of the data and smoothness potential parameters based on a training set $\mathcal{T} = \{\{\mathbf{X}_m^{(n)}\}_{m=1}^M, \mathbf{Y}^{(n)}\}_{n=1}^N$. The learning of the parameters depends on many factors. One important factor is the share of missing class decisions in each individual input decision stream. Furthermore, the task is complicated by the fact that there are no restrictions for the parameters such that not a unique but an infinite number of solutions are possible. In addition that, the data and smoothness potential parameters have a strong mutual dependency. Therefore, the proposed parameter learning approach is restricted to a data parameter $\mathbf{k} \in \mathbb{R}^M$ with no temporal resolution and a single scalar w for the lateral smoothness. The algorithm is composed of two steps in which the data and smoothness potential parameters are optimized. First, a regularization term is introduced to enforce $\mathbf{k}$ to converge to a unique solution. This is achieved by constraining the Euclidean length of data potential parameter vector to one. The resulting objective function to be minimized is thus given by

$$\Psi = \sum_{n=1}^{N} \sum_{m=1}^{M} \frac{1}{|\mathcal{L}_m^{(n)}|} \sum_{i=1}^{I} \sum_{t \in \mathcal{L}_m^{(n)}} k_m (x_{mit}^{(n)} - y_{it}^{(n)})^2 + \frac{\beta}{2} (\sum_{m=1}^{M} k_m^2 - 1) \tag{3.114}$$

where β is the Lagrange multiplier of the regularization term. The update formula is obtained by taking the derivative with respect to the parameter k_m:

$$\frac{\partial \Psi}{\partial k_m} = \frac{\partial}{\partial k_m} \sum_{n=1}^{N} \sum_{o=1}^{M} \frac{1}{|\mathcal{L}_o^{(n)}|} \sum_{i=1}^{I} \sum_{t \in \mathcal{L}_o^{(n)}} k_o (x_{oit}^{(n)} - y_{it}^{(n)})^2 + \frac{\beta}{2} (\sum_{o=1}^{M} k_o^2 - 1) \tag{3.115}$$

$$= \sum_{n=1}^{N} \frac{1}{|\mathcal{L}_m^{(n)}|} \sum_{i=1}^{I} \sum_{t \in \mathcal{L}_m^{(n)}} (x_{mit}^{(n)} - y_{it}^{(n)})^2 + \beta k_m = 0. \tag{3.116}$$

By normalizing k_m with $1/|\mathcal{L}_m|$ the focus is set only on the reliability, i.e. how accurate are the decisions of one classifier with disregard to the number of misses. Rearranging the formula and expanding it by k_m and isolating the Lagrange multiplier β results in

$$\Rightarrow \quad -\beta k_m = \sum_{n=1}^{N} \frac{1}{|\mathcal{L}_m^{(n)}|} \sum_{i=1}^{I} \sum_{t \in \mathcal{L}_m^{(n)}} (x_{mit}^{(n)} - y_{it}^{(n)})^2 \tag{3.117}$$

$$\Leftrightarrow \quad \sum_{m=1}^{M} \beta k_m^2 = - \sum_{n=1}^{N} \sum_{m=1}^{M} k_m \frac{1}{|\mathcal{L}_m^{(n)}|} \sum_{i=1}^{I} \sum_{t \in \mathcal{L}_m^{(n)}} (x_{mit}^{(n)} - y_{it}^{(n)})^2 \tag{3.118}$$

$$\Leftrightarrow \quad \beta = - \sum_{n=1}^{N} \sum_{m=1}^{M} k_m \frac{1}{|\mathcal{L}_m^{(n)}|} \sum_{i=1}^{I} \sum_{t \in \mathcal{L}_m^{(n)}} (x_{mit}^{(n)} - y_{it}^{(n)})^2. \tag{3.119}$$

Re-substituting the solution back into Equation 3.116, we obtain the gradient:

$$\begin{aligned} \Delta k_m^{\text{new}} = & \sum_{n=1}^{N} \frac{1}{|\mathcal{L}_m^{(n)}|} \sum_{i=1}^{I} \sum_{t \in \mathcal{L}_m^{(n)}} (x_{mit}^{(n)} - y_{it}^{(n)})^2 - \\ & k_m \sum_{m=1}^{M} k_m \frac{1}{|\mathcal{L}_m^{(n)}|} \sum_{i=1}^{I} \sum_{t \in \mathcal{L}_m^{(n)}} (x_{mit}^{(n)} - y_{it}^{(n)})^2. \end{aligned} \tag{3.120}$$

The optimal parameter $\mathbf{k}$ can be determined by iteratively computing the gradient and evaluating:

$$k_m^{\text{new}} = k_m + \epsilon \Delta k_m^{\text{new}} \tag{3.121}$$

where $\epsilon \in (0, 1]$ is the learning rate which approaches zero. The optimal scalar parameter w of the smoothness potential is determined by minimizing the root mean square (RMS) error between the ground truth annotation provided by the dataset $\mathcal{T}$ and the MFN estimate. The MFN estimate can be computed by choosing a value for w and using the optimal data parameter $\mathbf{k}$ and the input decisions streams of the dataset. Since there is only one unique solution for the smoothness potential parameter in this setting, the optimization algorithm starts with two initial values $\hat{w}$ and $\check{w}$ for which the corresponding RMS errors are determined. The errors are then used to derive the change in direction of the parameter in which the RMS error decreases. This minimization can be performed using a standard optimization algorithm.

In summary, the complete algorithm starts by finding the optimal data potential parameter $\mathbf{k}$ and then finds the best fitting smoothness potential parameter w. If multi-class problems are addressed, the optimization algorithm has to additionally keep track of finding a suitable parameter u to ensure that the probability axioms are not violated.

3.5.2. Early, Mid-level and Late Fusion Architectures

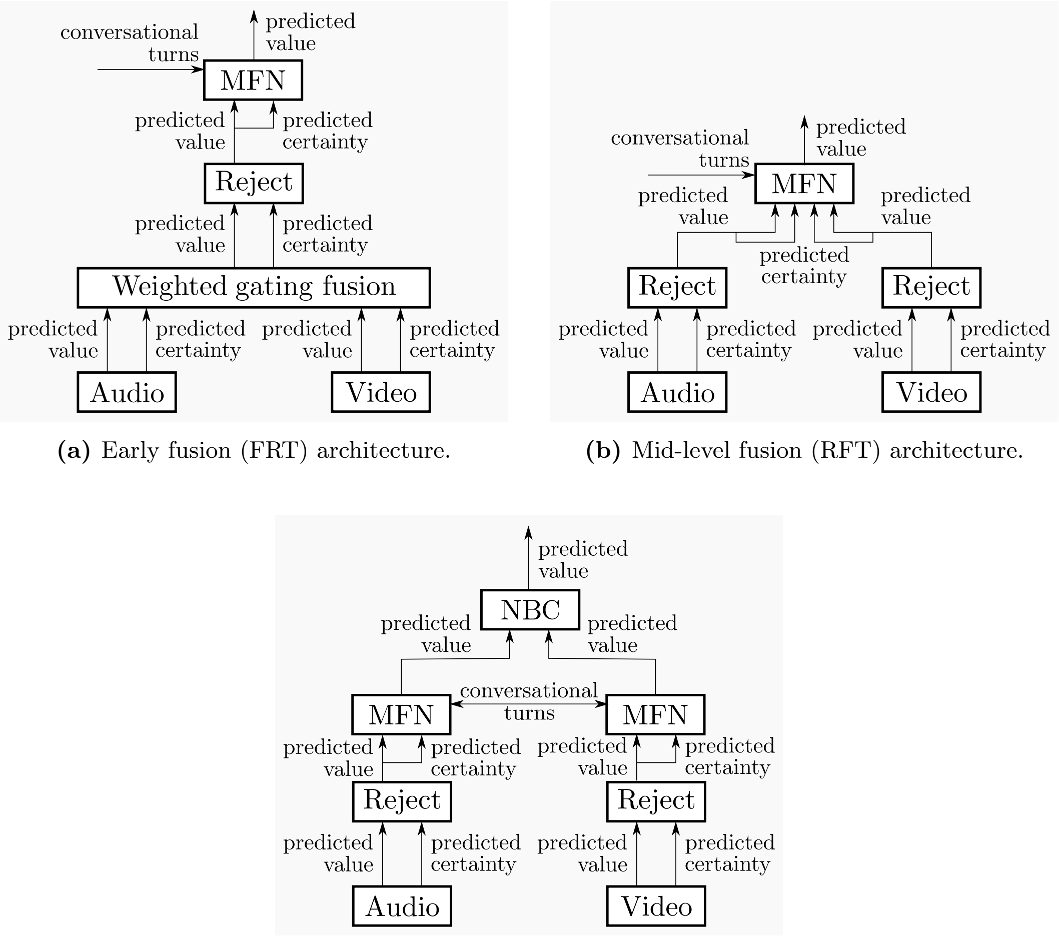
The MFN offers a large variety of possible MCS to combine multimodal sequential data. Three tasks of a temporal multimodal MCS can be identified: multimodal fusion, temporal fusion and handling of missing classifier decisions. These tasks can be transfered to a categorization of MCS similar to the one proposed by Dietrich (2004). We decompose the MCS into three components, namely the F-step (fusion of decisions), the R-step (rejection of uncertain decisions) and the T-step (temporal integration of decisions). By varying the F-step, we obtain three architectures FRT, RFT and RTF which can be regarded as early, mid-level and late-fusion architectures from the perspective of classifier outputs.

Figure 3.7 illustrates these architectures based using the example of affective state recognition in a dialog using audio and video channels. The audio recognition processes human utterances such that decisions are only fragmentarily available, while the video recognition operates on facial features requiring that the subject is facing the camera. Furthermore, it is presumed that the classifiers provide not only probabilistic classifier decisions, but also their decision confidences.

In case of the early fusion (FRT) architecture, the decisions and the confidences are combined before the rejection and temporal fusion takes place, as shown in Figure 3.7a. For each frame, the available decisions and confidences are averaged. In case no decisions are provided by any classifier, the final frame will be marked as having no decision available. Due to the early and rather rudimentary fusion, no outstanding performance can be expected by the FRT architecture. Figure 3.7b illustrates the mid-level (RFT) architecture which first rejects audio and video decisions separately based on confidences and then performs simultaneously a multimodal and temporal fusion using the MFN. The late fusion (RTF) architecture which is shown in Figure 3.7c implements the multimodal fusion after the temporal integration using an NBC. The temporal fusion in each modality is performed using the MFN.

3.6. Kalman Filter for Classifier Fusion

The Kalman filter constitutes an alternative approach for temporal multimodal classifier fusion (Glodek et al., 2013b; Kächele et al., 2014a). The input to the *Kalman filter for classifier fusion* is again a temporal sequence $\mathbf{X} \in [0, 1]^{M \times T}$ of M classifier decisions having the length T where the classifier decisions of each time step sum up to one. However, in



(a) Early fusion (FRT) architecture.

(b) Mid-level fusion (RFT) architecture.

(c) Late fusion (RTF) architecture.

Figure 3.7. Three MFN architectures FRT, RFT and RTF. The architectures are composed of three steps: the F-step (fusion of decisions), the R-step (rejection of uncertain decisions) and the T-step (temporal integration of the decisions). (a) The FRT architecture first combines the modalities, rejects uncertain classifier decisions and then performs the temporal fusion. (b) The RFT architecture first rejects uncertain classifier decisions and then performs the multimodal and temporal fusion together. (c) The RTF architecture first rejects uncertain classifier decisions, then performs the temporal fusion and then performs the late multimodal fusion using an NBC. For a detailed description please refer to the text.

contrast to the MFN the current design of the Kalman filter for classifier fusion does not allow a multi-class fusion. The estimate of the proposed Kalman filter models only a one-dimensional sequence. Therefore, only two-class classification problems can be considered. However, the Kalman filter takes natively care of missing classifier decisions.

The Kalman filter for classifier fusion has a reduced complexity compared to the traditional formalization in Section 2.7. The predicted state estimate is given by

$$\widehat{\mu}_{t+1} = a \cdot \mu_t + b \cdot u \tag{3.122}$$

$$\widehat{\sigma}_{t+1} = a \cdot \sigma_t \cdot a + q \tag{3.123}$$

where the matrices are replaced by scalars. The update and control terms are weighted by a

and b, respectively. The control u can be used to bias the prediction to a certain value, e.g. the least informative classifier prediction 0.5 in case of a two-class problem and with predictions ranging between $[0, 1]$. Due to the restriction of the state space to the values $[0, 1]$, the use of popular process models like dead reckoning which propagates the state using the last state and its first derivation with respect to the time is not used. Consequently, a non-linear version of the dead reckoning model would be necessary to keep the state restrictions. The covariance of the prediction is given by $\widehat{\sigma}_t$ and obtained by combining the a posteriori covariance with an additional covariance q which models the process noise. The successive update step is performed for every classifier m and requires three auxiliary variables, namely the residuum γ, the innovation variance s and the Kalman gain k_{t+1}:

$$\gamma = x_{mt+1} - h \cdot \widehat{\mu}_{t+1} \tag{3.124}$$

$$s = h \cdot \widehat{\sigma}_{t+1} \cdot h + r_m \tag{3.125}$$

$$k_{t+1} = h \cdot \widehat{\sigma}_{t+1} \cdot s^{-1} \tag{3.126}$$

where h is the observation model which maps the predicted quantity to the new estimate and r_m is the corresponding observation noise. These outcomes are then used to obtain the updated mean and variance:

$$\mu_{t+1} = \widehat{\mu}_{t+1} + k_{t+1} \cdot \gamma \tag{3.127}$$

$$\sigma_{t+1} = \widehat{\sigma}_t - k \cdot s \cdot k. \tag{3.128}$$

Missing classifier decisions are replaced by a measurement prior $\tilde{x}_{mt}$ equals to 0.5 and a corresponding observation noise $\tilde{r}_m$ which is set to a large value compared to the observation noise.

The final result is given by the vector of updated mean values $\boldsymbol{\mu}$. However, other quantities of the Kalman filter can provide additional information about the fusion process, such as updated variance which can serve as a confidence measure for the classification. Parameter learning can be performed by an additional backward pass comparable to the backward algorithm of the HMM and maximizing the parameters (Bishop, 2006, page 642ff.).

3.7. Layered Architectures for Complex Pattern Recognition

Basic and complex entities

There is a significant difference between pattern recognition operating close to the sensors and the abstract knowledge processing in symbolic AI (Dinsmore et al., 1992). Both paradigms use opposed types of entities with particular properties. This section starts by working out the details of these entities. In the following, we will call the classes detected

Table 3.1. Properties of basic and complex entities. Basic and complex entities describe patterns spanning from reification to abstraction. For a detailed description please refer to the text.

Properties	**Basic entity**	**Complex entity**
Observability	Direct	Indirect
Appearance	Similar	Variable
Context	Isolated	Relational
Temporal granularity	Small	Large
Divisibility	Elementary	Compiled
Processing	Sub-symbolic	Symbolic
Model origin	Data-driven	Human-generated

by pattern recognition the *basic entities*, and the relations modeled in symbolic AI the *complex entities*. A detailed list of properties describing these entities is provided in Table 3.1. The two entities can be distinguished by the properties: observability, appearance, context, temporal granularity, divisibility, processing and model origin.

The detection of basic entities is usually based on features which are extracted directly from sensor data. Hence, basic entities are regarded as being directly observable, whereas complex entities are only indirectly accessible with the help of the symbolic AI rules.

Basic entities are learned from observations given by a limited-sized dataset which covers all possible variations of the entity's appearance. Patterns having to many variations and which are further insufficiently covered by observations cannot be appropriately represented by statistical learning approaches. This reflects the fact that patterns of a basic entity are characterized by similar appearances. Opposed to that complex entities are defined by relationships on an abstract level. The reifications of a complex entity generally have a large variety of appearances (Glodek et al., 2014a).

As already mentioned, complex entities are defined by their relational rules. Usually these rules have a temporal or spatial nature which embeds the complex entities into a context, e.g. a temporal order of activities or a spatial linkage to other entities in the proximity. Basic entities on the other hand focus on patterns which are rather isolated, e.g. limited to a region of interest or a closed temporal window comprising only the target pattern.

Since basic entities usually disregard the context, they are typically characterized by a short-term, virtually static, perception span. Complex entities on the other hand usually have a temporal expansion, because state changes which are essential in symbolic AI require time to be executed. Hence, complex entities have a larger temporal granularity than basic entities (Oliver et al., 2004).

Complex entities can be compiled of basic entities by making use of the temporal and spatial relationships. Basic entity can be regarded as being elementary. An isolated object without any context provides only a minimal gain of information. However, linking basic entities to each other using relational rules can increase the value of information significantly. Therefore, basic entities (sub-patterns) can be used to compile complex entities (Glodek et al., 2011b, 2013a).

The next property distinguishes complex entities from basic entities by their way of processing. Basic entities are statistically learned from data and, therefore, described sub-symbolically. As a result, it is difficult for a human to put in words what constitutes a basic entity. In contrast to that, complex entities are described by a set of symbolic rules which enables humans to reproduce the decision making leading to a complex entity. It is important to emphasize that the terms symbolic and sub-symbolic do not relate to the representation but to the way of processing.

The last property is the model origin which is closely linked to the processing of the entities. Basic entities are derived using data-driven approaches. That means that they are statistically learned by fitting model parameters to a given dataset. Complex entities on the other side describe complex relationships in the real world such that the corresponding representations usually need to be generated by human.

The goal of this work is to bridge the gap between these two entities. This can be achieved by bringing both paradigms closer together such that they meet each other halfway. The classifiers have to detect basic entities which are useful for being processed by the symbolic AI, while the symbolic AI has to incorporate the uncertainty resulting from statistical pattern recognition.

3.7.1. Unidirectional Layered Architecture

The *unidirectional layered architecture* (ULA) extends the LHMM presented in Section 2.10 in multiple ways (Glodek et al., 2014a, 2013a, 2012a,d, 2011b,a).

In the ULA, the stream of discrete class assignments in a layer is replaced by stream of distributions over classes in order to increase the amount of information passed to the next layer. Figure 3.8 depicts an exemplary ULA in which streams of class distributions are propagated to the subsequent layers. Obviously, more information is transfered than in the LHMM architecture depicted in Figure 2.13.

Furthermore, the requirements on machine learning algorithms which are allowed to be deployed in the layers are relaxed. Different classifiers approaches may be more suitable depending on the characteristics of the patterns to be recognized within a layer. In case a pattern has no temporal expansion, it is plausible to resign from HMM and to use a standard classifier, e.g. NBC or SVM. However, generally the use of dynamic Markov models is still advisable since, as already pointed out, patterns in real-world scenarios are often characterized by their temporal nature. Patterns to be recognized in higher layers will start to show the properties of complex entities. The increasing variability of these patterns makes it difficult to collect a sufficient amount of data in order to train conventional pattern recognition approaches. However, the property that complex entities can be compiled from

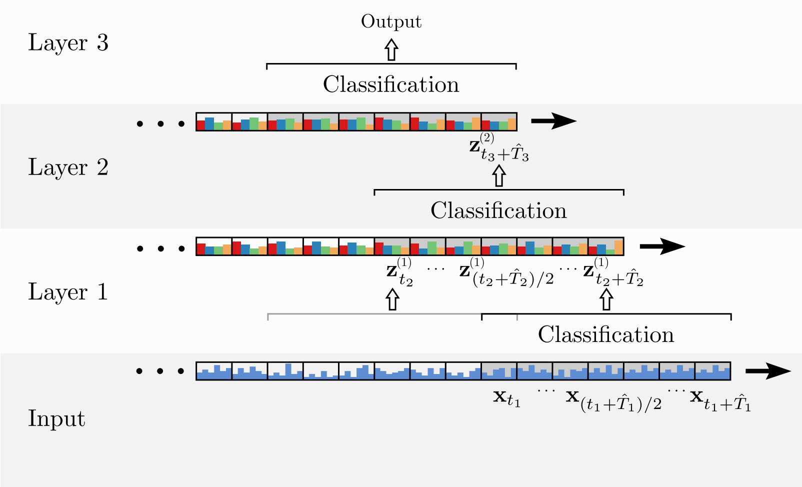

Layer 3
Output
Classification
Layer 2
$\mathbf{z}^{(2)}_{t_3+\hat{T}_3}$
Classification
Layer 1
$\mathbf{z}^{(1)}_{t_2}$ ··· $\mathbf{z}^{(1)}_{(t_2+\hat{T}_2)/2}$ ··· $\mathbf{z}^{(1)}_{t_2+\hat{T}_2}$
Classification
Input
$\mathbf{x}_{t_1}$ ··· $\mathbf{x}_{(t_1+\hat{T}_1)/2}$ ··· $\mathbf{x}_{t_1+\hat{T}_1}$

Figure 3.8. ULA passing probability distributions over classes to the next layer. The lowest layer operates on the stream of real-valued features. The subsequent layers operate on the stream of the former layer which is composed of probability class distributions.

basic entities using relational rules allows the application of symbolic AI. But the symbolic AI methods must make commitments: It is important that the seamless handling of uncertainty is preserved. Therefore, the transition to the next layer cannot be realized by strict symbolic AI methods, but requires probabilistic logic (Glodek et al., 2014a). A good candidate to do so is the DMLN which was introduced in Section 2.8. The distributions over classes from the previous layer are provided to the DMLN as observed inputs. The DMLN itself requires an additional random variable which represents the same class distribution, but can change during inference.

3.7.2. Bidirectional Layered Architecture

The ULA propagates information from the lower layers to the upper layers. That means, complex entities are compiled of basic entities. However, complex entities can also be used to validate basic entities. The recognition using sensor data may be based on misleading observations such that wrong information is propagated upwards. Ideally, this leads to an inconsistent situation in an upper layer which has the effect that the fault is identified and corrected. Alternatively, the upper layer can also propagate inferred information back to lower layers. The lower layer can then reconsider the proposed class distribution and help to eliminate the inconsistency.

The proposed *bidirectional layered architecture* (BLA) requires many design decisions to be made regarding the training and inference of the complete system. Questions are among others: Given that ground truth is available for each layer, should the training be performed separately in each layer? How can the feedback of the upper layers be integrated into the lower layer? How and at which time of the processing can the feedback be propagated forward and backward in time?

The BLA was developed in close collaboration with Thomas Geier from the Institute of Artificial Intelligence of the University of Ulm. In order to implement the BLA, the layered architecture is again restricted to the CHMM in the lower layer and the DMLN in the upper layer. The key idea is to improve the exchange of state information between the CHMM and the DMLN during inference. The Figure 3.9 shows a graphical representation of the BLA in which a schematic drawing of the DMLN is shown in the upper layer and the lower layer depicts a CHMM. The depicted fictional structure of DMLN must realize temporal connections between the template time slices and provide random variables which can be shared with the CHMM during inference. The DMLN models the distribution $p(\mathbf{y}, \mathbf{Z})$ where $\mathbf{Z}$ are the states of the first-order logic rules and $\mathbf{y}$ are the states which the DMLN has in common with the CHMM. The CHMM models the distribution $p(\mathbf{X}|\mathbf{y})$ where $\mathbf{X}$ are the observations extracted from sensor data which are conditioned to the shared states. The actual structure of the DMLN is human-generated, but the parameters of the DMLN can be trained using the labeled ground truth. The unobserved random variables of the causal

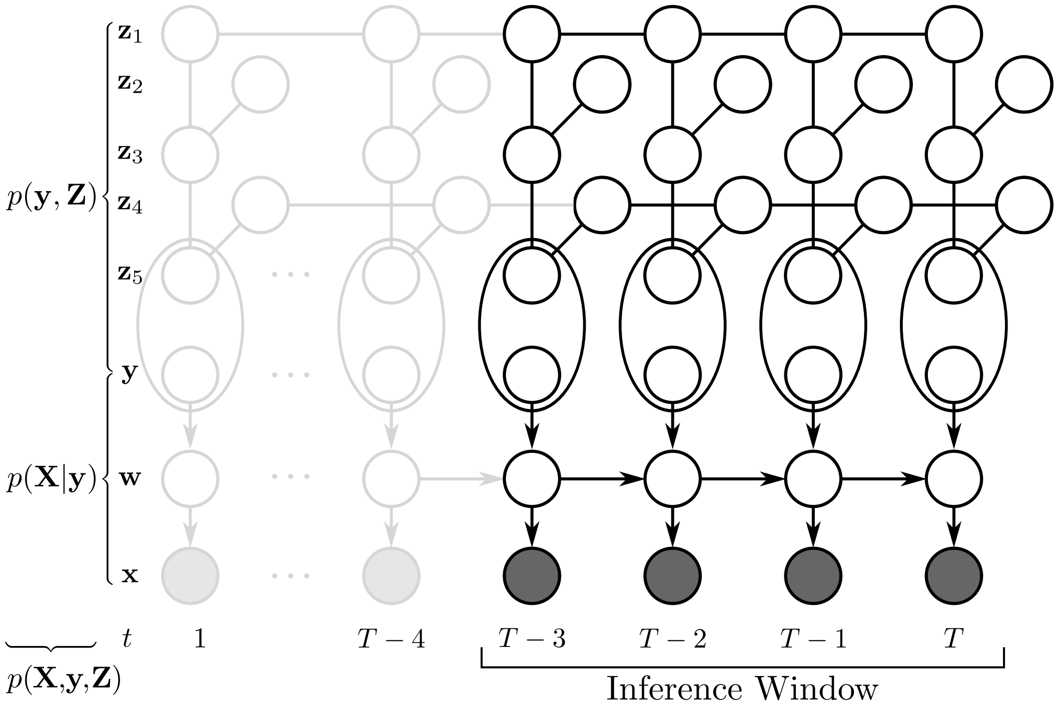


Figure 3.9. Schematic drawing of the BLA illustrating the connection between the last two layers. The upper part of the figure displays a DMLN modeling the distribution $p(\mathbf{y}, \mathbf{Z})$, whereas the lower part represents the CHMM modeling $p(\mathbf{X}|\mathbf{y})$. By connecting the shared random variables of the CHMM and DMLN layers, a complete model with the probability distribution $p(\mathbf{X}, \mathbf{y}, \mathbf{X})$ is realized. Computationally efficient inference is performed by shifting a window over the stream of the former layer which should ideally comprise multiple entities. For the purpose of illustration, the depicted inference window spans only over a small set of time slices. For a detailed description please refer to the text.

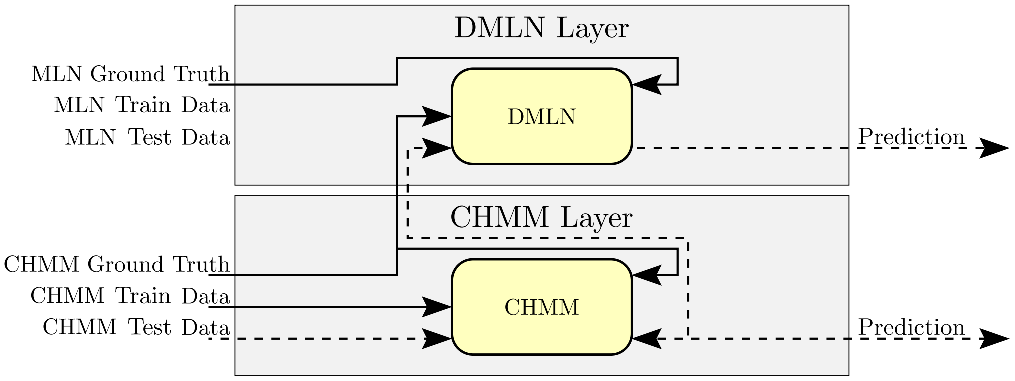


Figure 3.10. Training and testing of the BLA. The training of the CHMM is performed using the CHMM data and the CHMM labels. The DMLN is trained using the CHMM labels as inputs and the DMLN labels as targets. In order to perform the final testing, the CHMM and DMLN are joined and inference is conducted on the complete model. The shared random variables ensure that information is propagated between the layers. For a detailed description please refer to the text.

sequence of the CHMM are shared with the DMLN during inference which is required for testing. Figure 3.10 depicts the procedure of training and testing. Initially, each layer is trained individually. The CHMM layer is trained using observed data and the corresponding crisp ground truth. Usually the training data will comprise long sequences with multiple labels on the level of complex entities. Hence, the performance rating of trained models requires the incrementally testing as described in Section 3.2. The parameters for both model are fixed after the training is conducted. However, the final testing will be performed differently using both models.

In order to perform inference, the CHMM and DMLN get connected using the shared random variables $\mathbf{y}$. As a result, the predicted causal sequence of the CHMM can directly influence the DMLN and vice versa. The probability of the complete model is given by $p(\mathbf{Z}, \mathbf{y}, \mathbf{X}) = p(\mathbf{X}|\mathbf{y})p(\mathbf{y}, \mathbf{Z})$. The sharing of states between both models is illustrated in Figure 3.10 by an ellipse encircling the causal random variable of the CHMM and the random variable of the DMLN having the same semantics. The CHMM and the DMLN are both dynamic models such that the connections will unavoidably produce loops. Hence, inference has to be performed by a suitable algorithm such as loopy belief propagation (Koller and Friedman, 2009).

Another important issue which need to be addressed is the size of the model. With each time step, the model grows and becomes increasingly computational expensive. Therefore, we propose the use of an inference window as indicated in Figure 3.9. The reason to do so is that the influence of past events vanishes and will have only a minor effect on the most recent state. However, the inference window has to span over sufficient basic entities such that inconsistencies can still be deduced.

3.8. Inequality Constraint Multi-class F^2-Support Vector Machine

The MC-F^2-SVM which was presented in Section 2.12 uses the objective function to influence the hyperplane with the given multi-class fuzzy labels. This section will present a new approach to realize an MC-F^2-SVM called the *inequality constraint multi-class fuzzy-in and fuzzy-out support vector machine* (IC-MC-F^2-SVM) (Glodek et al., 2014b). The new approach incorporates the fuzzy values using the inequality constraints.

The dataset $\mathcal{T} = \{(\mathbf{x}^{(n)}, \mathbf{y}^{(n)})\}_{n=1}^{N}$ is again given by N tuples. A data sample is denoted by $\mathbf{x}^{(n)}$ and the corresponding fuzzy class membership over I classes is denoted by $\mathbf{y}^{(n)} = (y_1^{(n)}, \dots, y_I^{(n)})$ where $y_i^{(n)} \in [0,1]$ and $\sum_{i=1}^{I} y_i^{(n)} = 1$. The objective function is then given by

$$\text{minimize} \quad \frac{1}{2}\sum_{i=1}^{I} \mathbf{w}_i^{\mathrm{T}} \mathbf{w}_i + C \sum_{n=1}^{N}\sum_{i=1}^{I} \xi_i^{(n)} y_i^{(n)} \tag{3.129}$$

where the slack variable $\xi_i^{(n)}$ is additionally weighted by the quantity of the corresponding fuzzy membership $y_i^{(n)}$. This weighting amplifies the importance of the labeled memberships. The larger the membership value, the higher the penalty of the slack variable which penalizes data samples being assigned to the wrong side of the hyperplane. The inequality constraints are defined by

$$1 - \xi_i^{(n)} \leq (y_i^{(n)} - y_j^{(n)})(\mathbf{w}_i^{\mathrm{T}} \mathbf{x}^{(n)} + b_i) - (y_i^{(n)} - y_j^{(n)})(\mathbf{w}_j^{\mathrm{T}} \mathbf{x}^{(n)} + b_j) \tag{3.130}$$

$$0 \leq \xi_i^{(n)}. \tag{3.131}$$

The first inequation uses the difference between two fuzzy memberships to ensure that the hyperplane separates the data samples accurately. The second inequation restricts the slack variable to be non-negative in order to avoid a trivial solution of the problem. The objective function and the inequality constraints together assemble the constraint optimization problem which is given by

$$\begin{aligned} L(\{\mathbf{w}_i, b_i, \boldsymbol{\xi}_i, \boldsymbol{\alpha}_i, \boldsymbol{\beta}_i\}_{i=1}^{I}) = {} & \frac{1}{2}\sum_{i=1}^{I} \mathbf{w}_i^{\mathrm{T}} \mathbf{w}_i + C \sum_{n=1}^{N}\sum_{i=1}^{I} \xi_i^{(n)} y_i^{(n)} \\ & + \sum_{n=1}^{N}\sum_{j=1}^{I}\sum_{i=1}^{I} \alpha_i^{(n)} \Big(1 - \xi_i^{(n)} + (y_i^{(n)} - y_j^{(n)})(\mathbf{w}_j^{\mathrm{T}} \mathbf{x}^{(n)} + b_j) \\ & - (y_i^{(n)} - y_j^{(n)})(\mathbf{w}_i^{\mathrm{T}} \mathbf{x}^{(n)} + b_i)\Big) - \sum_{n=1}^{N}\sum_{i=1}^{K} \beta_i^{(n)} \xi_i^{(n)}. \end{aligned} \tag{3.132}$$

The problem formulation introduces the additional Lagrange multipliers $\alpha_i^{(n)}$ and $\beta_i^{(n)}$ which both have to be non-negative in order to satisfy the Karush-Kuhn-Tucker conditions (Bishop, 2006, p. 710). In the following, the Lagrange multipliers will no longer be listed as arguments of the Lagrangian function to keep the notation uncluttered. The dual form is derived by taking the derivatives of the model parameters and setting the outcomes equal to zero:

$$\frac{\partial L}{\partial \mathbf{w}_i} = \mathbf{w}_i + \sum_{n=1}^{N}\sum_{j=1}^{I}(y_j^{(n)} - y_i^{(n)})\big(\alpha_i^{(n)} + \alpha_j^{(n)}\big)\mathbf{x}^{(n)} = 0 \tag{3.133}$$

$$\Leftrightarrow \qquad \mathbf{w}_i = \sum_{n=1}^{N}\mathbf{x}^{(n)}\Bigg(\underbrace{\sum_{j=1}^{I}(y_i^{(n)} - y_j^{(n)})\alpha_i^{(n)}}_{=:u_i^{(n)}} + \underbrace{\sum_{j=1}^{I}(y_i^{(n)} - y_j^{(n)})\alpha_j^{(n)}}_{=:v_i^{(n)}}\Bigg) \tag{3.134}$$

$$\frac{\partial L}{\partial b_i} = \sum_{n=1}^{N}\sum_{j=1}^{I}\alpha_j^{(n)}(y_j^{(n)} - y_i^{(n)}) - \alpha_i^{(n)}(y_i^{(n)} - y_j^{(n)}) = 0 \tag{3.135}$$

$$\Leftrightarrow \qquad \sum_{i=1}^{I}\sum_{n=1}^{N}\sum_{j=1}^{I}(y_i^{(n)} - y_j^{(n)})\alpha_i^{(n)} b_i = \sum_{j=1}^{I}\sum_{n=1}^{N}\sum_{i=1}^{I}(y_i^{(n)} - y_j^{(n)})\alpha_i^{(n)} b_j \tag{3.136}$$

$$\frac{\partial L}{\partial \xi_i^{(n)}} = Cy_i^{(n)} - \sum_{j=1}^{I}\alpha_i^{(n)} - \beta_i^{(n)} = 0. \tag{3.137}$$

Equation 3.134 introduces two new variables $u_i^{(n)}$ and $v_i^{(n)}$ which will used in the following to keep the notation uncluttered. The obtained results are substituted back into the Equation 3.132. In the first step, $\mathbf{w}_k$ is substituted which yields

$$L(\mathbf{w}_i, b_i, \xi_i^{(n)}) = \frac{1}{2}\sum_{i=1}^{I}\sum_{n_1}\sum_{n_2}\big(u_i^{(n_1)} + v_i^{(n_1)}\big)\big(u_i^{(n_2)} + v_i^{(n_2)}\big)\mathbf{x}^{(n_1)\mathrm{T}}\mathbf{x}^{(n_2)} \tag{3.138}$$

$$+\sum_{n=1}^{N}\sum_{i=1}^{I}\xi_i^{(n)}\underbrace{\Big(Cy_i^{(n)} - \sum_{j=1}^{I}\alpha_i^{(n)} - \beta_i^{(n)}\Big)}_{=\ 0 \text{ according to Equation 3.137}} \tag{3.139}$$

$$+\sum_{n=1}^{N}\sum_{j=1}^{I}\sum_{i=1}^{I}\alpha_i^{(n)} \tag{3.140}$$

$$+\sum_{n=1}^{N}\sum_{j=1}^{I}\sum_{i=1}^{I}\alpha_i^{(n)}(y_i^{(n)} - y_j^{(n)})(\mathbf{w}_j^{\mathrm{T}}\mathbf{x}^{(n)} - \mathbf{w}_i^{\mathrm{T}}\mathbf{x}^{(n)}) \tag{3.141}$$

$$\underbrace{+\sum_{n=1}^{N}\sum_{i=1}^{I}\sum_{i=1}^{I}\alpha_i^{(n)}(y_i^{(n)} - y_j^{(n)})(b_j - b_i).}_{=\ 0 \text{ according to Equation 3.136}} \tag{3.142}$$

Further rearranging and substituting lead to the final dual form:

$$\begin{aligned} L(\mathbf{w}_i, b_i, \xi_i^{(n)}) = {} & \sum_{n=1}^{N}\sum_{j=1}^{I}\sum_{i=1}^{I}\alpha_i^{(n)} - \frac{1}{2}\sum_{i=1}^{I}\sum_{n_1}\sum_{n_2}\big(u_i^{(n_1)}u_i^{(n_2)} + u_i^{(n_1)}v_i^{(n_2)} \\ & + v_i^{(n_1)}u_i^{(n_2)} + v_i^{(n_1)}v_i^{(n_2)}\big)\mathbf{x}^{(n_1)\mathrm{T}}\mathbf{x}^{(n_2)}. \end{aligned} \tag{3.143}$$

The dual form needs to be optimized with the additional constraints $0 \leq \alpha_i^{(n)} \leq Cy_i^{(n)}$ and $\sum_{n=1}^{N}\sum_{j=1}^{I}(y_i^{(n)} - y_j^{(n)})\big(\alpha_i^{(n)} + \alpha_j^{(n)}\big) = 0$. The testing of a new data sample $\hat{\mathbf{x}}$ is done by determining the distance to the I hyperplanes and to choose the largest distance:

$$d_i(\hat{\mathbf{x}}) = \sum_{n=1}^{N} \big(\sum_{j=1}^{I} (y_i^{(n)} - y_j^{(n)}) \cdot (\alpha_i^{(n)} + \alpha_j^{(n)}) \big) \mathbf{x}^{(n)} \hat{\mathbf{x}} + b_i. \tag{3.144}$$

The bias is identified by collecting a set of support vectors for which the inequality constraints $0 < \alpha_i^{(n)} < C y_i^{(n)}$ are strictly fulfilled. These support vectors give rise to a set of linear equations

$$\begin{aligned} 1 =&\ (y_i^{(n)} - y_j^{(n)})(\mathbf{w}_i^{\mathrm{T}} \mathbf{x}^{(n)} + b_i) - (y_i^{(n)} - y_j^{(n)})(\mathbf{w}_j^{\mathrm{T}} \mathbf{x}^{(n)} + b_j) && (3.145) \\ \Leftrightarrow \quad 0 =&\ \mathbf{w}_i^{\mathrm{T}} \mathbf{x}^{(n)} + b_i - \mathbf{w}_j^{\mathrm{T}} \mathbf{x}^{(n)} - b_j - \frac{1}{(y_i^{(n)} - y_j^{(n)})} && (3.146) \\ \Leftrightarrow \quad b_i - b_j =&\ \mathbf{w}_j^{\mathrm{T}} \mathbf{x}^{(n)} - \mathbf{w}_i^{\mathrm{T}} \mathbf{x}^{(n)} + \frac{1}{(y_i^{(n)} - y_j^{(n)})}. && (3.147) \end{aligned}$$

The linear system can be solved using the pseudo inverse or a linear solver. A complete rank of the linear system can be derived by adding an additional constraint $\sum_{i=1}^{I} b_i = 0$.

Chapter 4

Datasets and Evaluation Criteria

This chapter presents the datasets and evaluation criteria which will be used to assess the performance of the methods proposed in the previous chapter. The results of the empirical evaluations are provided in Chapter 5 and are going to be organized in the same way as the current chapter. The results of the empirical evaluations will be discussed in Chapter 6.

The first part of this chapter presents the datasets used for the empirical evaluations. Section 4.1.1 presents two datasets for assessing the density estimation performance of the EGMM and related approaches. The following Section 4.1.2 describes two datasets which will be utilized to study the HMM-GPD and CHMM-GPD in the area of action recognition. The evaluation of the temporal multimodal fusion approaches, i.e. MFN and Kalman filter for classifier fusion, is mainly done using three datasets for affective state recognition which will be presented in Section 4.1.3. The last two presented datasets were recorded in order to evaluate the ULA and BLA and are described in Section 4.1.4.

The second part of this chapter introduces the measures utilized to rate the performance of the algorithms in the empirical evaluations. Two types of measures are presented: (1) performance measures for crisp classifier predictions and (2) performance measures for density estimation.

4.1. Datasets

The current section gives a detailed overview of the datasets being empirically evaluated in the following chapter. Within this work, the feature extraction is generally seen as part of a dataset, since the focus is set on the evaluation of the developed machine learning algorithms.

4.1.1. Density Estimation Datasets

Two artificial datasets are utilized to assess the performance of the EGMM and related state-of-the-art approaches. The first dataset is based on two classification problems which were proposed by Ormoneit and Tresp (1996) to measure density estimation performances. The second dataset evaluates the errors between an estimated density and a ground-truth density using the bias-variance decomposition. This evaluation criterion will be presented later in Section 4.2.

4.1.1.1. Ormoneit Dataset I and Ormoneit Dataset II

The first dataset will be used to rate density estimation algorithms by solving a non-separable two-class classification problem which is defined by two overlapping ring-shaped distributions. The performance is measured by estimating the densities for each class. The class densities are then used in an NBC. The dataset is created by drawing samples from the respective class distribution, see Algorithm A.1 in the Appendix for details. Ormoneit and Tresp (1996) who originally proposed the dataset suggested two parameter configurations. These parameter configuration are referred to as *Ormoneit Dataset I* (ODI) and *Ormoneit Dataset II* (ODII). The proposed parameter configurations are listed in Table 4.1 where σ determines the dispersion around the ring, Δ denotes the offset from the center of each ring from the origin, and N corresponds to the overall sample size of the dataset. The ODI has 400 two-dimensional samples, whereas the ODII has 800 ten-dimensional samples. Especially, ODII can be considered as being a challenging dataset because of the low number of samples

Table 4.1. Configurations of the Ormoneit Dataset I and Ormoneit Dataset II. Each dataset is defined by a configuration of the Algorithm A.1 provided in the Appendix.

Name	σ	Δ	D	N
Ormoneit Dataset I	0.2	±0.5	2	400
Ormoneit Dataset II	0.2	±0.3	10	800

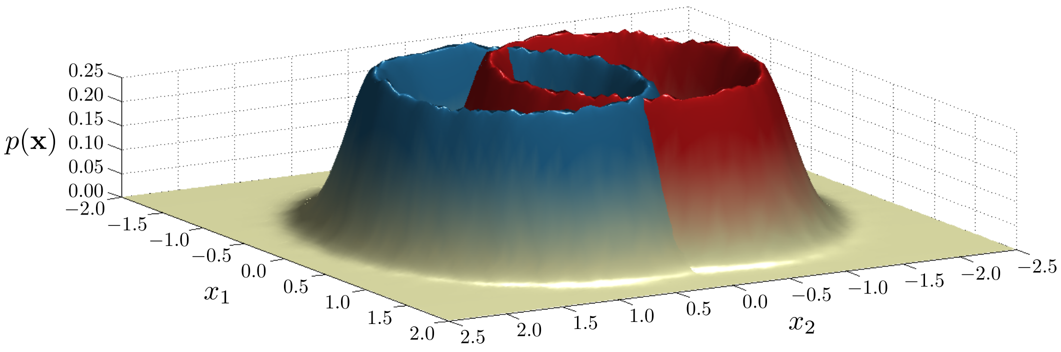


Figure 4.1. Surface plot of the probability densities defined by the Ormoneit Dataset I. The density representation is obtained using a Gaussian kernel density estimation applied to $N = 1000000$ drawn samples per class.

and the high dimensionality. Figure 4.1 visualizes the two independent ring-shaped class densities colored in red and blue that are learned from the ODI using a large number of $N = 1000000$ samples per class.

4.1.1.2. Beta Distribution Dataset

The second dataset which is named *Beta Distribution Dataset* (BDD) was designed to measure the ability of density estimation algorithms to model non-Gaussian distributions. To do so, three beta distributions with different parameters a and b are used which range from sub-Gaussian to almost Gaussian shapes (Joanes and Gill, 1998). The higher the parameters of the beta distribution, the more the distribution resembles the shape of a Gaussian distribution. The three parameters used for the dataset are (1) $a = b = 1$, (2) $a = b = 2$ and (3) $a = b = 10$. Figure 4.2 shows the shape of the selected distributions. A number of $N = 350$ samples were drawn from each of the distributions. The similarity to the beta distribution will be studied using the bias-variance decomposition which will be presented in Section 4.2.

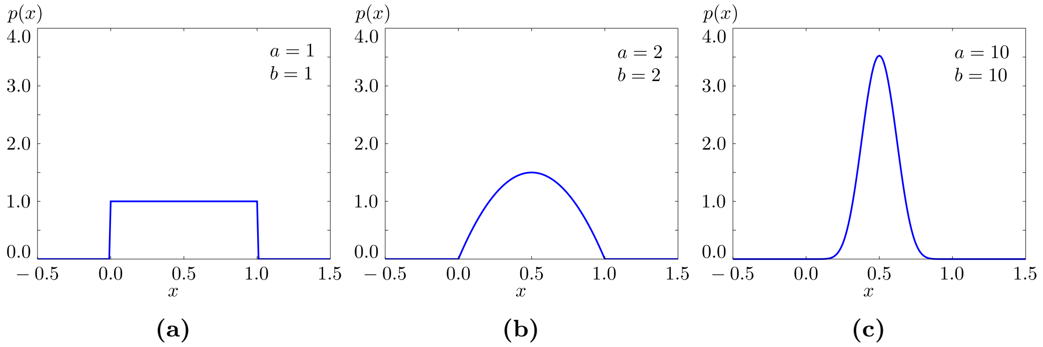


Figure 4.2. Three configurations of the beta distribution being part of the Beta Distribution Dataset. The configurations are (a) $a = b = 1$, (b) $a = b = 2$ and (c) $a = b = 10$.

4.1.2. Action Datasets

Two different datasets were recorded to study the HMM-GPD and CHMM-GPD which were presented in Section 3.4. Both datasets address the application of action recognition. The first dataset was recorded to conduct a feasibility study and It attests that the HMM-GPD can reliably recognize actions even under heavy occlusions. The second dataset is situated in a more complex setting and features a larger set of samples, classes and self-occlusions from three camera perspectives.

4.1.2.1. Occlusion Action Dataset

The first action dataset named *Occlusion Action Dataset* (OAD) was introduced to demonstrate the application of the HMM-GPD on data provided by the Kinect™ camera[1]. The camera captures an RGB image, a matching depth map, bit masks of tracked users and fitted skeleton models for up to two persons. The camera has the advantageous property not of being obtrusive, i.e. the camera does not actively emit any signal in the visible spectrum and the human does not have to wear a special gear. However, the skeleton model of the user may suffer from self-occlusions due to the single camera viewpoint. The HMM-GPD will be utilized to demonstrate how to make use of the technical benefits of the camera while circumventing the problem of self-occlusions.

The dataset contains three actions which were performed using the left or the right hand: (1) "Drink" (DC), (2) "Eat" (EA) and (3) "Read" (RN). Figure 4.3 shows the action EA in a shortend sequence of four frames. The camera was placed directly in front of the subject such that the recordings are free from any occlusions. The displayed frames show the depth map overlayed with the bit mask of the person and the skeleton model provided by the camera. The frames are resampled at 10 Hz in order to compensate the varying frame rates. The dataset is partitioned into 10 × 10 cross-validation folds, see Algorithm A.3.1 in the

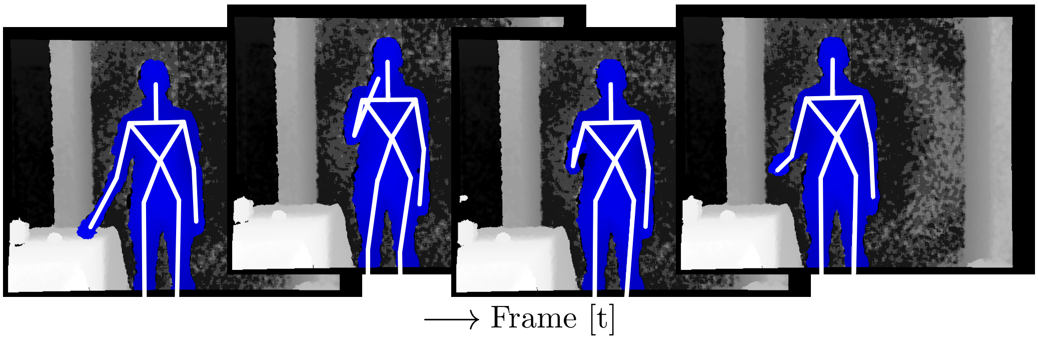
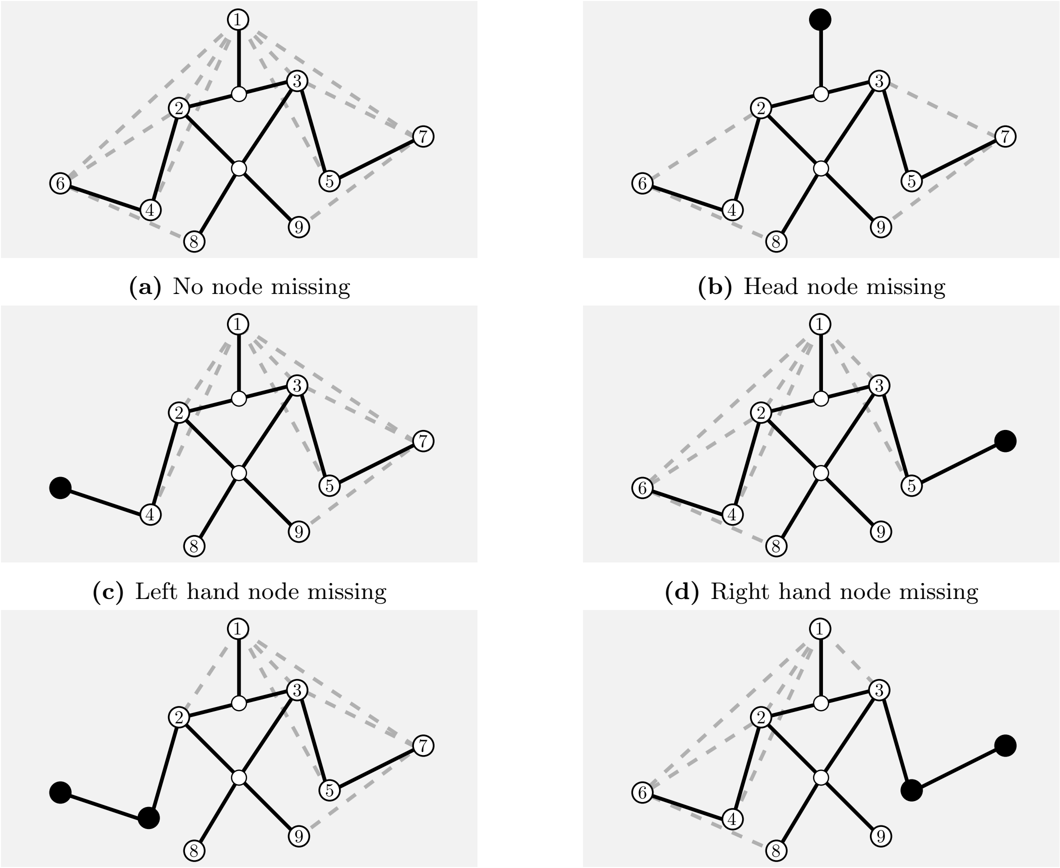


Figure 4.3. Sequence of four selected frames showing the action class "Eat" in the Occlusion Action Dataset. Each frame displays the depth map overlayed by the user's bit mask and skeleton model. The subject picks up an apple from the desk, takes a bite and then returns it back to the desk.

Table 4.2. Statistics on the Occlusion Action Dataset. N_{initial} and N_{mirror} denote the sample size of the original dataset and the dataset after mirroring the skeleton model, respectively. $N_{\text{mirror+jitter}}$ is the number of samples in the dataset after performing the mirroring and jittering.

Action	N_{initial}	N_{mirror}	$N_{\text{mirror+jitter}}$
Drink (DC)	32	64	640
Eat (EA)	30	60	600
Read (RN)	25	50	500

[1] The Kinect™ camera is an input device developed by Microsoft®. A detailed description of the camera specifications can be found at `http://www.xbox.com/en-US/xbox-360/accessories/kinect` (last visited 10/06/2015).



(a) No node missing **(b)** Head node missing

(c) Left hand node missing **(d)** Right hand node missing

(e) Two nodes on the left side missing **(f)** Two nodes on the right side missing

Figure 4.4. The six occlusion configurations of the Occlusion Action Dataset. Different parts of the skeleton model were removed in a post-processing step. The numbered nodes represent the joints of the skeleton model. Blackened nodes are masked to simulate occlusion. The black solid lines connect the nodes to form the actual skeleton model. The dashed lines indicate the edges of the graph derived by connecting the non-occluded joint nodes. The complete graph is available in the occlusion configuration (a). Graphs with varying missing joint nodes are provided in the occlusion configurations (b,c,d,e,f).

Appendix for details. Once the folds are determined, the samples are duplicated and one instance is horizontally flipped. Furthermore, the data are multiplied by creating ten copies with a small jitter which is applied to the head, hands and elbows of the skeleton model. Thus, the number of samples per fold has increased by a factor of 20 which is summarized in Table 4.2.

In the next step, a graph representation is extracted from the skeleton model using the Euclidean distances between selected joint nodes. The occlusion is artificially modeled by systematically removing sets of joints nodes. Due to the removed joint nodes, distances cannot be computed such that an incomplete graph is obtained. Figure 4.4 shows the six different occlusion configurations to extract graphs from the skeleton model. The numbered nodes mark the joints available for the feature extraction, whereas the blackened nodes are assumed to be occluded. The actual skeleton model is depicted by solid black lines. The dashed lines represent the edges of the extracted graph. If there is no occlusion, a number of ten edges is extracted as shown in Figure 4.4a. The lowest number of extracted edges in a graph is six in which the hand and elbow joints of one side are removed, i.e. Figure 4.4e and Figure 4.4f. Altogether six different occlusion configurations are considered: (a) "None", (b) "Head", (c) "Left hand ", (d) "Right hand", (e) "Left hand and left elbow" and (f) "Right hand and right elbow". In the following, the different occlusions configurations are referred to by the labels a, b, c, d, e and f, as shown in Figure 4.4. For further processing, the data sequences are transformed into the representation as described in Section 2.4. A graph in the dataset is represented by $\mathbf{x}_t = \{(\mathring{x}_{mt}, \tilde{x}_{mt})\}_{m=1}^{M}$ where t is the time step and M denotes the number of edges. Each edge is indexed using a discrete label $\mathring{x}_{mt}$ and is associated with the Euclidean distance $\tilde{x}_{mt}$ between the involved joint nodes.

4.1.2.2. UUlm Multi-perspective Action Dataset

The *UUlm Multi-perspective Action Dataset* (UUlmMAD) was created in the course of this thesis in collaboration with Georg Layher from the Vision and Perception Science Lab, University of Ulm (Glodek et al., 2015b). The goal of the dataset was to provide a basis for the development of novel action recognition approaches with many-faceted challenges.

The unique characteristics of the dataset result from the experimental setup which is schematically drawn in Figure 4.5. Three cameras[1] were placed in the equidistant distance of 7 m and 2 m height from the center of the action workspace with a 45° angular offset between the cameras. The subjects performed a number of predefined actions in front of

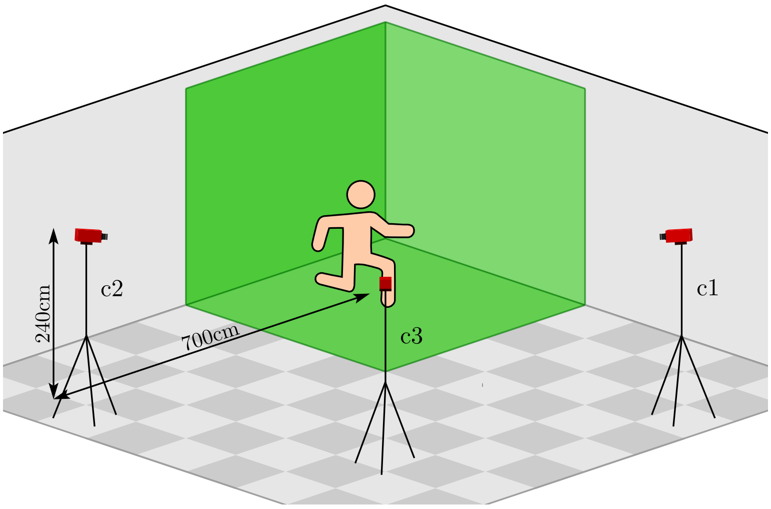
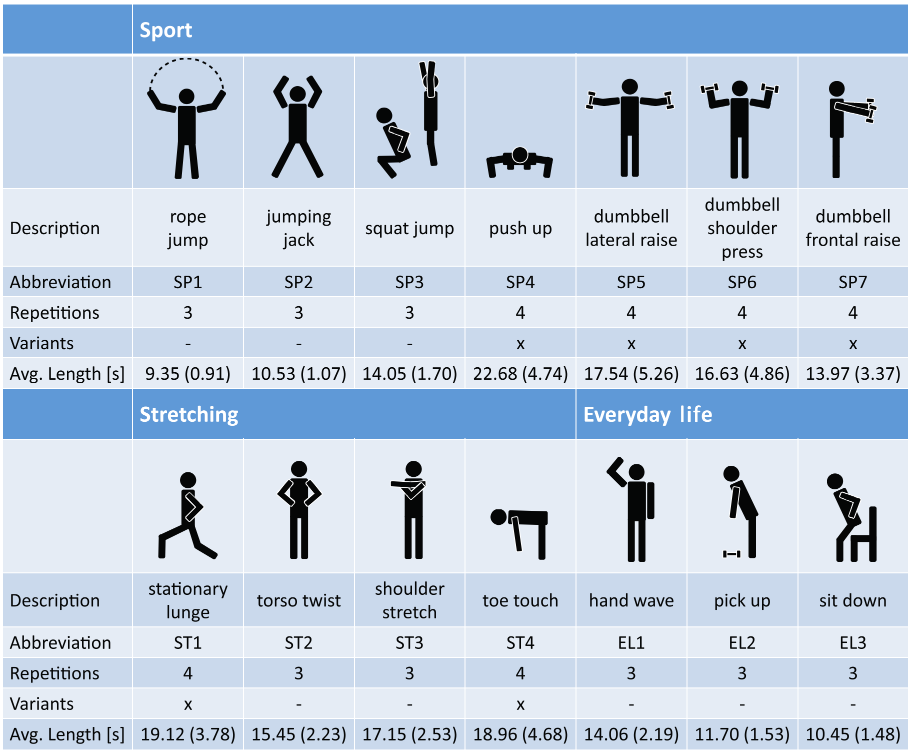


Figure 4.5. Experimental setup of the UUlm Multi-perspective Action Dataset. Three cameras c1, c2 and c3 are placed in equidistant distances of 7 m to the subject and 2 m height with 45° angular offset to capture center of the action workspace. The action workspace is revetted with a green screen.

[1] Pike F-145 from Allied Vision with a Tevidon 1,8/16 lens.



	Sport						
Description	rope jump	jumping jack	squat jump	push up	dumbbell lateral raise	dumbbell shoulder press	dumbbell frontal raise
Abbreviation	SP1	SP2	SP3	SP4	SP5	SP6	SP7
Repetitions	3	3	3	4	4	4	4
Variants	-	-	-	x	x	x	x
Avg. Length [s]	9.35 (0.91)	10.53 (1.07)	14.05 (1.70)	22.68 (4.74)	17.54 (5.26)	16.63 (4.86)	13.97 (3.37)
	Stretching				Everyday life		
Description	stationary lunge	torso twist	shoulder stretch	toe touch	hand wave	pick up	sit down
Abbreviation	ST1	ST2	ST3	ST4	EL1	EL2	EL3
Repetitions	4	3	3	4	3	3	3
Variants	x	-	-	x	-	-	-
Avg. Length [s]	19.12 (3.78)	15.45 (2.23)	17.15 (2.53)	18.96 (4.68)	14.06 (2.19)	11.70 (1.53)	10.45 (1.48)

Figure 4.6. The fourteen action classes of the UUlm Multi-perspective Action Dataset. The action classes are taken from the areas of fitness sports (SP), stretching (ST) and everyday life (EL). For each action, the table provides a pictogram, a short description, the abbreviation, the number of repetitions or variants and the average duration with standard deviation over all subjects.

a green screen. In the following, the frontal camera is named c1, the side-view or lateral camera c2 and the diagonally-placed camera c3. The cameras have a focal length of 16.3 mm and a sensor size of 2/3"and recorded with a synchronized and constant frame rate of 30 Hz using a resolution of 960 × 1280 px. All three cameras were calibrated intrinsically and extrinsically. Furthermore, the subjects wore an inertial motion capturing system[1] which recorded the 3D joint positions based on 3D gyroscopes, accelerometers and magnetometers (Roetenberg et al., 2009) to capture the body movements. The black motion capturing suit possesses orange inertial sensors which have a prominent appearance. Hence, it was covered by additional clothing whenever possible.

The performed actions are taken from the field of fitness sports (SP), stretching (ST) and everyday life (EL). The 14 actions are shown in Figure 4.6 with a pictogram, a short description, an abbreviation for future references, the conducted repetitions of an exercise, variants and the average duration with standard deviation in seconds. The actions are intentionally chosen in a way such that, depending on the viewpoint, actions may have a similar visual appearance, e.g. “Squat jump” (SP3) and “Jumping Jack” (SP2) or “Pick up” (EL2) and “Toe touch” (ST4). Each action was recorded three times. However, the subjects per-

[1] Xsens MVN Biomech^{TM} `https://www.xsens.com/` (last visited 10/06/2015)

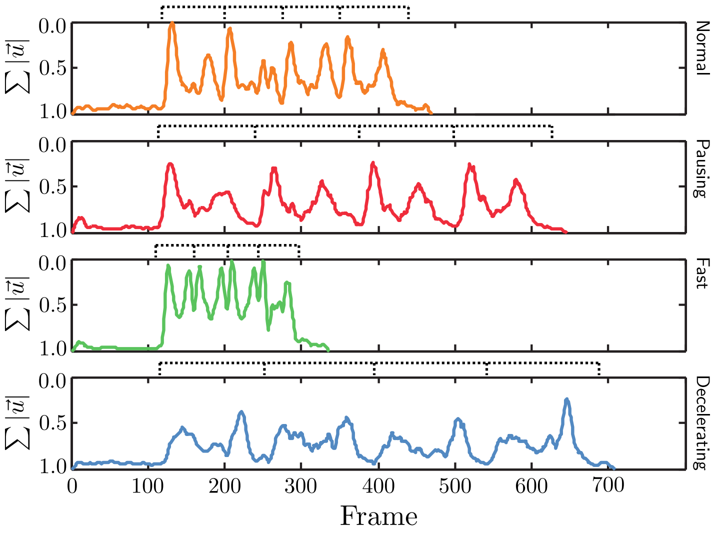


Figure 4.7. Variants of actions recorded in the UUlm Multi-perspective Action Dataset. A subset of sport action was recorded with four variants “Normal”, “Pausing”, “Fast” and “Deceleration”. The figure illustrates the concept of these variants. Each plot displays the normalized summed absolute optical flow of the frontal camera of action “Dumbbell lateral raise” (SP5) as performed by the subject p24. They are characterized by typical temporal patterns corresponding to the action variant. For a detailed description please refer to the text.

formed a subset of actions four times in different variants, i.e. “Normal”, “Pausing”, “Fast” and “Deceleration”. An action is composed of segments of movements which are repeated four times. All recordings start in a natural pose, i.e. the subject stands upright and faces camera c1 with arms placed beside the body.

Figure 4.7 exemplifies the concept of the four action variants using the action “Dumbbell lateral raise” (SP5) performed by the subject p24 from the perspective of the frontal camera c1. Each plot shows the normalized summed absolute optical flow over time for the variants “Normal“, “Pausing“, “Fast“ and “Deceleration“. On the top of each plot, the action is sectioned in its elementary movements by dotted lines, i.e. each time the dumbbell is raised and lowered a segment ends and the next begins. As seen from the plots, all recordings start with an initial state in which the subjects were asked to take up the neutral pose for three seconds. Hence, the summed absolute optical flow values in the beginning of all four plots are low. The uppermost plot shows the “Normal” variant in which the subject performed a continuous movement with average speed. The second plot shows the “Pausing” variant in which the subject was asked to hold still in-between the elementary movements. In this scenario, the amount of absolute optical flow drops almost to zero at the end of each movement segment. The third plot displays the variant in which the subject performed the variant “Fast”. Here, the temporal duration of the action is reduced to a short period with high amplitude values on the absolute optical flow. The lowest plot shows the “Deceleration” variant in which the subjects was asked to act as if the performed action would require physical strain. The plot shows a lower absolute optical flow than the other plots and a lower acceleration in the beginning of each repetition. The variants represent sub-classes of actions which can be additionally recognized if needed.

Overall 31 subjects were recorded with an average age of 28.77 ± 4.22 yr, an average height of 177.04 ± 11.57 cm and an average weight of 75.01 ± 13.79 kg. The male to female ratio is 77%. The dataset will be evaluated using leave-one-subject-out cross-validation.

The dataset will be used to study the HMM-GPD, CHMM-GPD and a related image-based state-of-the-art classifier architecture. The sequence of graphs was extracted similarly to the OAD using the Euclidean distances derived from the joints of the skeleton model. The self-occlusions was generated based on an artificial 3D avatars such that the graphs vary over time. In the following, details on the the generation of 3D avatar video sequences will be presented.

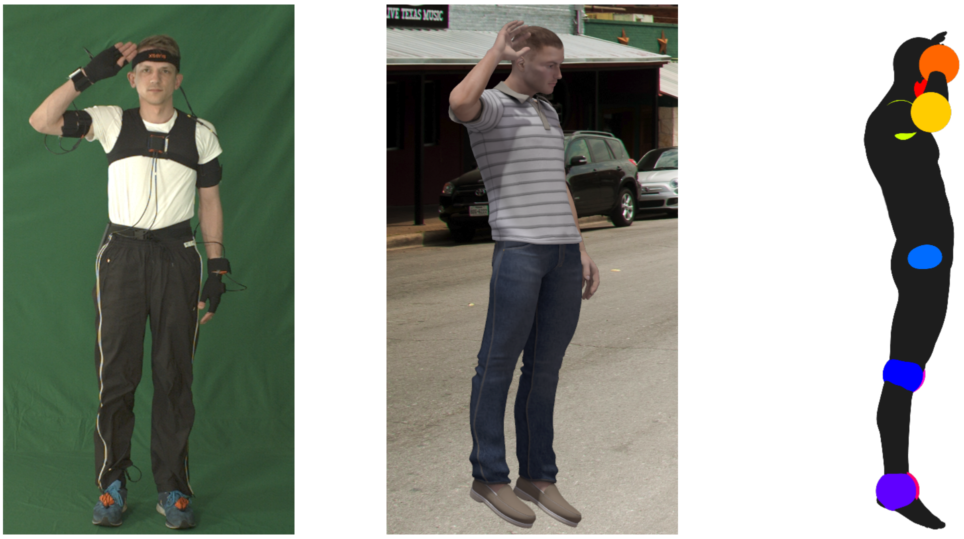
Generation of Self-occlusion Data Using 3D Avatars

To derive self-occlusion information, the motion capturing data were processed by a software named Poser[1] which is specialized for rendering human avatars. The software allows the generation of new action sequences in which a virtual human can be rendered from any perspective and under various illumination conditions. The shaders of the avatar were replaced such that the human body appeared to be solid black while important joints of the rendered body were augmented with uni-colored spheres. The colors were taken from equidistant locations of the hue channel in the hue-saturation-value (HSV) color space. Each action was then rendered using the same setup as the three cameras c1, c2 and c3. The information of the rendered images were used to create histograms of the sphere's colors. In case the number of pixels in one bin felt under a certain threshold, the joint was assumed to be occluded.

Figure 4.8 illustrates the generation of new video material. Figure 4.8a depicts an original video frame of camera c1 of the action "Hand wave" (EL1). Figure 4.8b shows the corresponding frame as generated by the software using the motion capturing data and a realistic avatar and background from the perspective of camera c3. Figure 4.8c shows the rendering output to derive the self-occlusion information from the viewpoint of camera c1.

For further processing, the skeleton model was normalized according to its total height. The visible joints were used to create a complete graph where each edge between two joints was labeled with a unique identifier and weighted by their Euclidean distance. In case all joints are visible the complete graph has 68 edges. The sequence of graphs for all actions and subjects were derived for four settings, i.e. the perspectives of camera c1, camera c2, camera c3 and all joints visible, with a frame rate of 5 Hz. Five graph datasets were created based on these four settings: one dataset for each camera perspective, a dataset containing all camera

[1] Poser$^{\mathrm{TM}}$is a 3D modeling software for human avatars by Smith Micro Software.



(a) Camera c1 **(b)** Camera c3 **(c)** Camera c2

Figure 4.8. Generation of additional rendered video material in the UUlm Multi-perspective Action Dataset. The motion capturing data can be processed by a rendering software to generate new action sequences in a post hoc fashion to change the perspective or the illumination condition. The figures show the action class "Hand wave" (EL1) from different perspectives: (a) original input image, (b) rendered human avatar with street background and (c) human avatar to derive self-occlusion data.

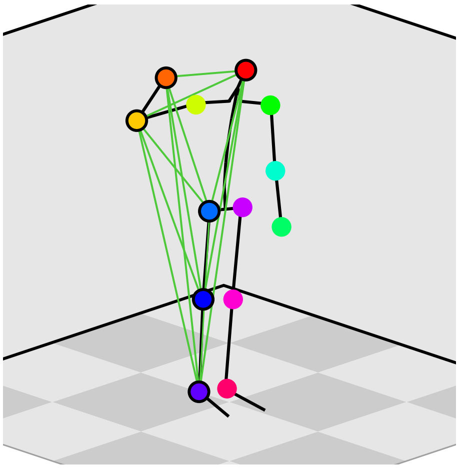
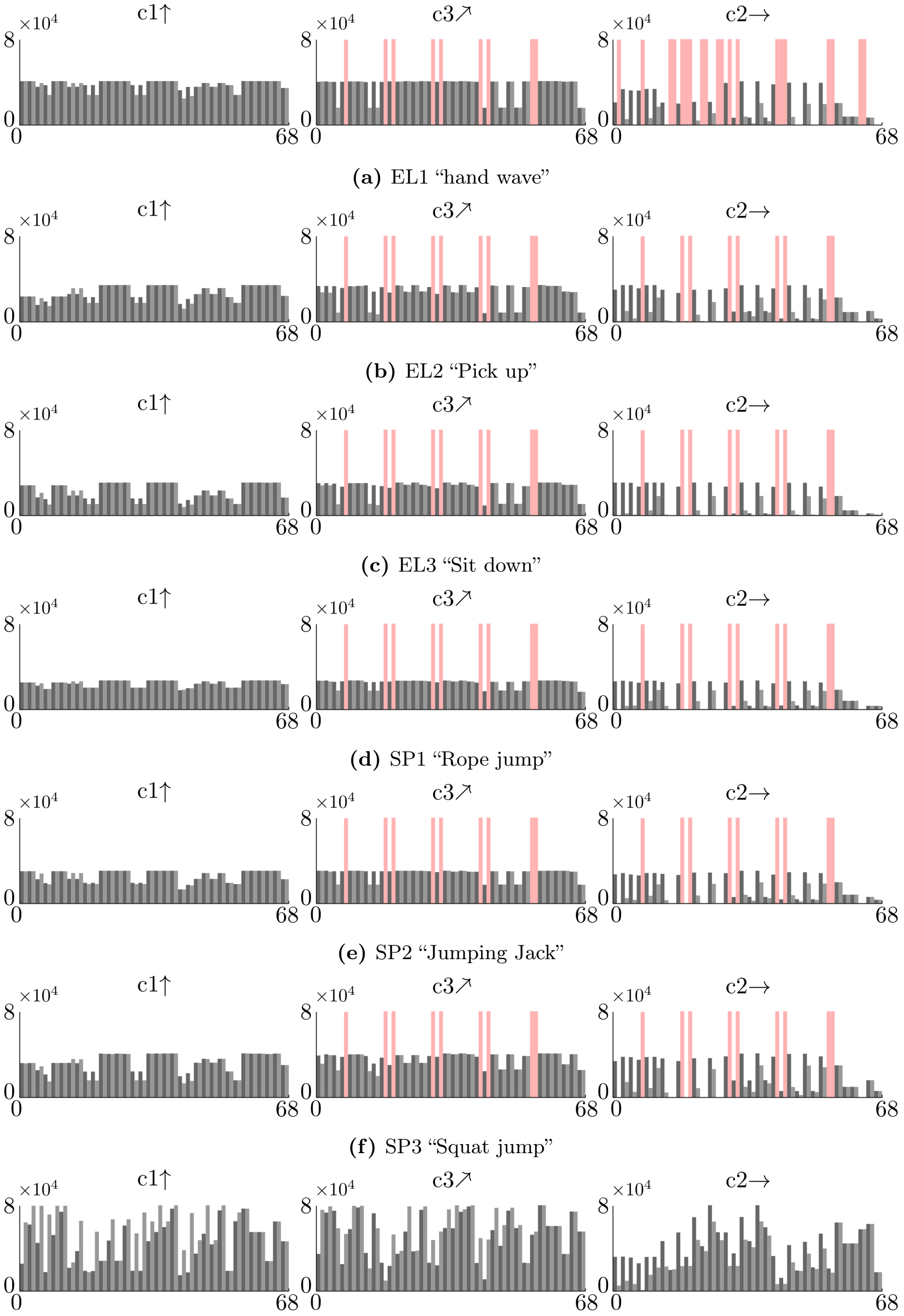


Figure 4.9. Extracted graph with occlusions derived from the rendered self-occlusion video of the action class "Hand wave" (EL1) in the UUlm Multi-perspective Action Dataset. The self-occlusion video is used to identify the visible joints which are then connected to create the graph. The joints are depicted by colored circles. Visible joints are additionally encircled by a black stroke. Each edge between two joints is labeled with unique identifier and weighted by the Euclidean distance between the joints.

perspectives and a complete graph dataset. Furthermore, an initialization dataset was created based on the first three seconds of the actions where the subjects waited in the neutral position. The initialization dataset was composed of complete graph sequences. Figure 4.9 depicts the occlusion information extracted from the rendered image of Figure 4.8c. The extraction of the self-occlusion from the perspective of camera c2 led to a number of bins being below the predefined threshold. The corresponding joints were encircled by a black stroke. Figure 4.10 and Figure 4.11 depict the distribution of the number of edges per action and camera perspective over all subjects and sequences. In case all edges of an action were observed for all frames, the maximal number of edges is constantly 68 for all edges. The overall maximal number of possible edges strongly depends on the camera perspective, but



(a) EL1 “hand wave”

(b) EL2 “Pick up”

(c) EL3 “Sit down”

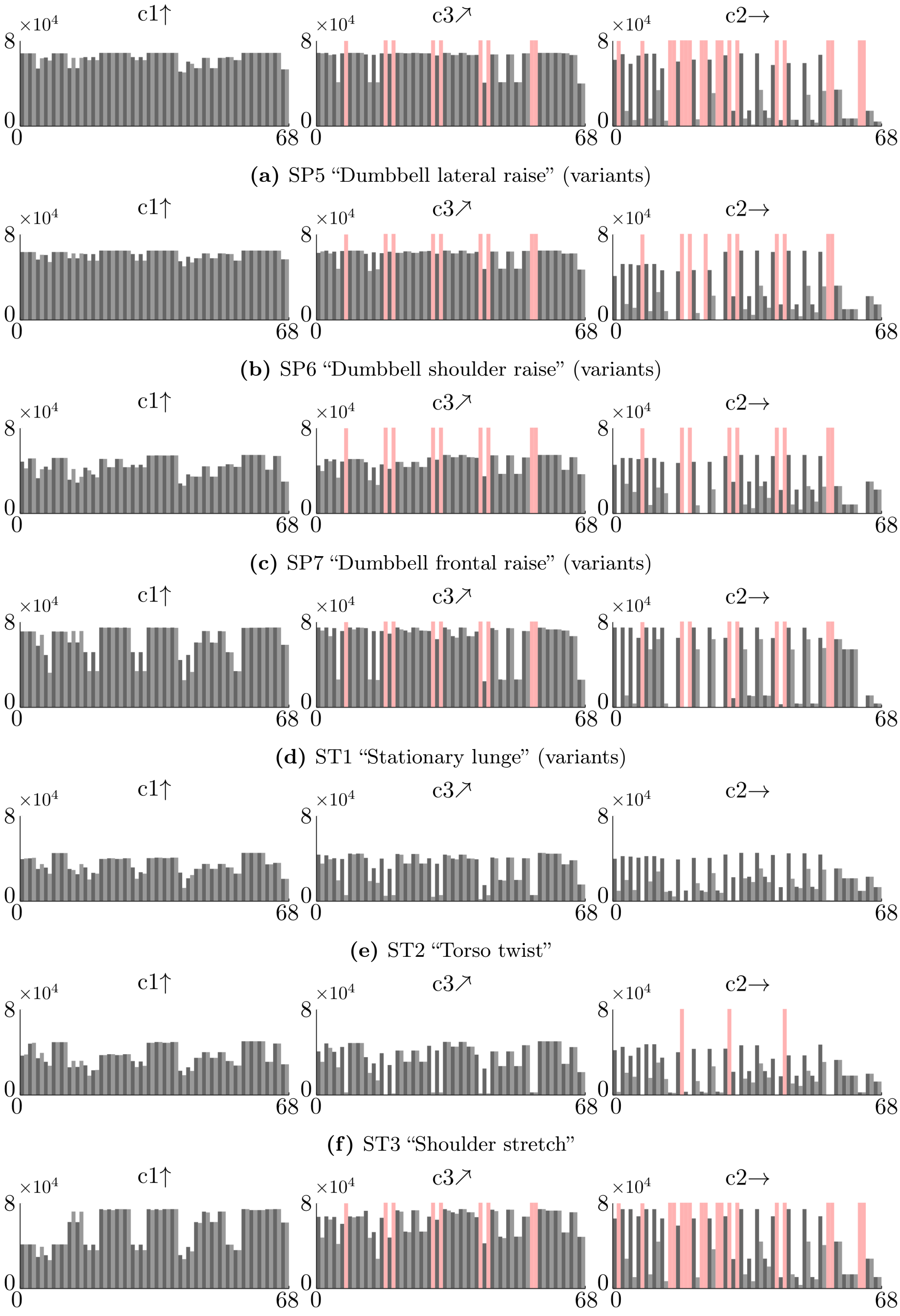
(d) SP1 “Rope jump”

(e) SP2 “Jumping Jack”

(f) SP3 “Squat jump”

(g) SP4 “Push up” (variants)

Figure 4.10. Bar chart showing the number of edges accumulated over all persons in the UUlm Multi-perspective Action Dataset for the cameras c1, c2 and c3. The number of edges depends on the self-occlusions which is heavily influenced by the camera perspective. A number of zero edges is marked by a red bar. The frontal camera c1 has the highest chance of observing all edges, whereas the lateral camera c2 usually observes the lowest number of edges.



(a) SP5 “Dumbbell lateral raise” (variants)

(b) SP6 “Dumbbell shoulder raise” (variants)

(c) SP7 “Dumbbell frontal raise” (variants)

(d) ST1 “Stationary lunge” (variants)

(e) ST2 “Torso twist”

(f) ST3 “Shoulder stretch”

(g) ST4 “Toe touch” (variants)

Figure 4.11. Bar chart showing the number of edges accumulated over all persons in the UUlm Multi-perspective Action Dataset for the cameras c1, c2 and c3. The number of edges depends on the self-occlusions which is heavily influenced by the camera perspective. A number of zero edges is marked by a red bar. The frontal camera c1 has the highest chance of observing all edges, whereas the lateral camera c2 usually observes the lowest number of edges.

also on the length of the sequences. Edges which do not occur in an action and a camera perspective are highlighted by a red bar. As to be seen from the figures, most edges were usually visible from the perspective of the frontal camera c1. In general, fewer edges were visible from the perspective of the diagonally-placed camera c3, and even lesser edges from the lateral perspective of camera c2. However, there were exceptions such as action SP4 (Figure 4.10g), action ST2 (Figure 4.11e) and action ST3 (Figure 4.11f). In case of these three actions, the subjects performed a rotation in which they faced the camera c2 for a short-duration. As a result, all edges are present at some time in the data of all camera perspectives. The opposed extreme is given by action EL1 (Figure 4.10a), action SP5 (Figure 4.11a) and action ST4 (Figure 4.11f). Here, the subjects performed movements in the direction of camera c3 such that the far side of their bodies was permanently occluded. As a result, these actions have a large number of totally missing edges.

4.1.3. Datasets for Affective State Recognition

Datasets containing affective human expressions provide a welcome domain to study MCS. Due to the nature of the classification problem, the data are often multimodal and uncertain (Kächele et al., 2014b; Schels et al., 2012a). Furthermore, classes are often annotated using fuzzy labels. In this work, three datasets will be studied: (1) the Freetalk Dataset, (2) the AVEC 2011 Dataset and (3) the AVEC 2012 Dataset. In the following, these three datasets will be presented in detail.

4.1.3.1. Freetalk Dataset

The *Freetalk Dataset*[1] was created with the objective to study multimodal temporal MCS for the recognition of laughter. The dataset comprises two separate 90 minute recordings of four people sitting around a table involved in a lively and unconstrained natural conversation. The recordings were done using a single unobtrusive 360° spherical camera and a microphone which was placed on the middle of the table. Figure 4.12 shows a video frame of the dataset to exemplify the recording setting. The faces were tracked in the video stream (Viola and Jones, 2002) and for each face the relative head movement was derived as a feature (MOV). From the audio channel two features were extracted: the modulation spectrum (MOD) and the perceptual linear prediction (PLP) (Hermansky and Morgan, 1994; Maganti et al., 2007). The MOD feature captures the dynamics in the channel and was derived from the 16 kHz audio signal using a window of 50 ms shifted by 20 ms and an inner window which spans over 200 ms such that a laughter can be adequately represented. The final feature vector representation is eight-dimensional. The PLP feature was extracted from a 32 ms window shifted by 10 ms rendering a 21-dimensional vector. Further details on the feature extraction

[1] The Freetalk Dataset is available at `http://www.speech-data.jp/` (last visited 10/06/2015)

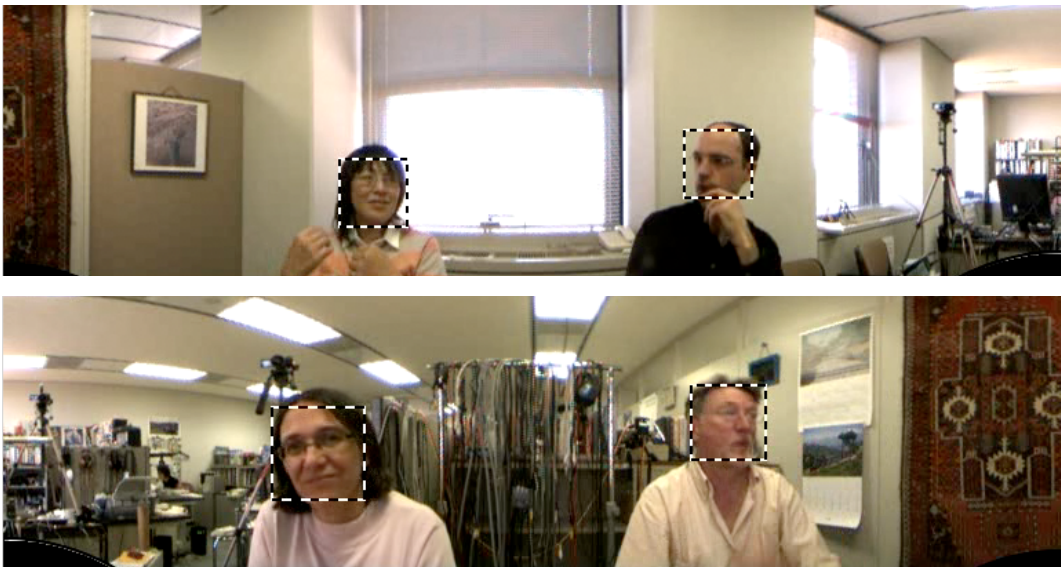


Figure 4.12. Video frame taken from the Freetalk Dataset. The recordings were done using a 360° spherical camera. The depicted video frame is overlayed by dashed lined squares marking the detected faces.

Table 4.3. Statistics on the Freetalk Dataset. Number of samples and average length of the pre-segmented and unsegmented datasets being part of the Freetalk Dataset. The pre-segmented dataset is composed of short snippets of single laughs and utterances. The unsegmented dataset is based on one continuous recording in which laughter and utterances have been annotated.

Property	**Pre-segmented**	**Unsegmented**
Number (Laughter)	256	194
Number (Utterance)	2270	195
Avg. length (Laughter)	0.95±0.53 s	2.02±1.94 s
Avg. length (Utterance)	1.57±0.96 s	22.98±31.79 s

can be found in (Scherer et al., 2012b). Both recordings were labeled for "Utterance" and "Laughter" with the annotator using both available modalities. Since only one microphone was used, the recording contains overlapping speech and laughs of all four persons which made the annotation difficult.

Table 4.3 provides statistics about the dataset derived from the two recordings. Based on the first recording a pre-segmented dataset was created with a number of 256 laughter and 2270 utterance samples. The annotation was performed on the level of sentences and laughters which were precisely isolated. Hence, the number of 2270 utterances is clearly higher than the number of 256 laughters. The average length of the laughters is 0.95 s while an average utterance lasts around 1.57 s The second recording will be used to examine unsegmented sequential data. Hence, it was not segmented into their target classes, but only the changes between laughters and utterances were annotated. As a result, no fine-grained laughters or sentences can be identified when testing on the unsegmented dataset. The average length of an annotated laugh has increased to 2.02 s with a standard deviation of 1.94 s. The average length of an utterance is 22.98 s. The number of laughters and utterances are balanced to 194 and 195, respectively.

4.1.3.2. AVEC 2011 Dataset

The *Audio/Visual Emotion Challenge* (AVEC) was used to study different MCS which implement temporal multimodal fusion. The first challenge dataset was the AVEC 2011 which was introduced as a workshop being part of the ACII 2011. It contains audio-visual recordings of 13 subjects interacting with four emotionally stereotyped artificial agents, each having a particular affective characteristic (Schuller et al., 2011; McKeown et al., 2010). The dataset was labeled with four affective dimensions: “Arousal”, “Expectancy”, “Power” and “Valence” (Fontaine et al., 2007). For each recording, the number of annotators range from 2 to 8. The annotations were averaged for each dimension resulting in a real-valued label for each time step. Subsequently, the labels were binarized using a threshold equal to the grand mean of each dimension. In addition to the sensor data and affect annotation, a word-by-word transcription of the spoken language and conversational turns was provided. Three sub-challenges were proposed: an audio challenge on word-level, a video challenge on frame-level and an audio-visual challenge on the frame-level of the video channel[1]. According to the official guideline of the challenge, the goal is to predict the four label dimensions using exclusively features derived from either audio, video or both modalities. The final ranking of the challenge submissions was done only on the label dimension of “Arousal”, since the submissions on other dimensions yielded poor results.

The dataset was divided into training, development and test sets. The labels of the test set have not been published. Table 4.4 shows the partitioning and statistic of the challenge dataset. For further studies, we re-partitioned the available datasets into 4 × 4 cross-validation folds in a way such that each fold has a disjunct set of test subjects. Besides the raw video and audio recordings, the challenge organizers provided a large range of pre-computed audio and video features. However, we decided to give preference to our own extracted features. In the following, a detailed description of the utilized features will be provided.

Audio Features

Three different kinds of features were extracted from the audio signal:

- The fundamental frequency values were extracted using the f_0 tracker tool available in the ESPS/*waves+* software package. Besides the f_0 track, the energy and the linear predictive coding (LPC) of the plain wave signal were derived (Rabiner and Juang, 1993). All three features were concatenated to form a ten-dimensional early fusion feature vector.

[1] A comprehensive description and the data can be found at `http://sspnet.eu/avec2011/` (last visited 10/06/2015).

Table 4.4. Statistics on the AVEC 2011 Dataset. The dataset was partitioned into training, development and test sets. The table specifies the number of recordings, frames and words, as well as the complete length and the average word length of each partition. Table adapted from (Schuller et al., 2011).

Property	Training	Development	Test	Total
Number of recordings	31	32	32	95
Frames	501,277	449,074	407,772	1,358,123
Words	20 183	16 311	13 856	50 350
Complete length (hh:mm:ss)	2:47:10	2:29:45	2:15:59	7:32:54
Avg. word length (ms)	262	276	249	263

- The representation of the mel frequency cepstral coefficients (MFCC) is inspired by the biological perceptual variations in the human auditory system. It is modeled using a linearly spaced filter bank in lower frequencies and logarithmically in higher frequencies in order to capture the phonetically important characteristics of speech. The MFCC were extracted as described in (Zheng et al., 2001).

- The perceptual linear predictive (PLP) analysis is based on two perceptually and biologically motivated concepts, namely the critical bands and the equal loudness curves. Frequencies below 1 kHz need higher sound pressure levels than the reference and sounds between 2 – 5 kHz need less pressure. This follows the way of human perception. The critical band filtering uses equally spaced filters based on the Bark scale which are trapezoidal shaped. After the critical band analysis and equal loudness conversion, the subsequent steps required for the relative spectral (RASTA) processing extension follow the implementation recommendations provided by Hermansky et al. (1992). Subsequent to the transforming the spectrum to the logarithmic domain and the application of RASTA filtering, the signal is transformed back using the exponential function. The PLP were extracted using the same parameters as described in the Freetalk Dataset.

Video Features

Two different video feature sets were extracted from the AVEC 2011 Dataset. The first set is composed of biologically-inspired video features which were kindly provided by the Vision and Perception Science Lab[1]. The second set contains high-level features extracted by the computer expression recognition toolbox (CERT) (Littlewort et al., 2011).

The biologically-inspired feature set models two mainly independent pathways. The first one is specialized for the processing of form and shape, while the second one is specialized for motion. Both pathways are hierarchically organized with larger scales of interaction

[1] http://www.informatik.uni-ulm.de/ni/staff/HNeumann/ (last visited 10/06/2015)

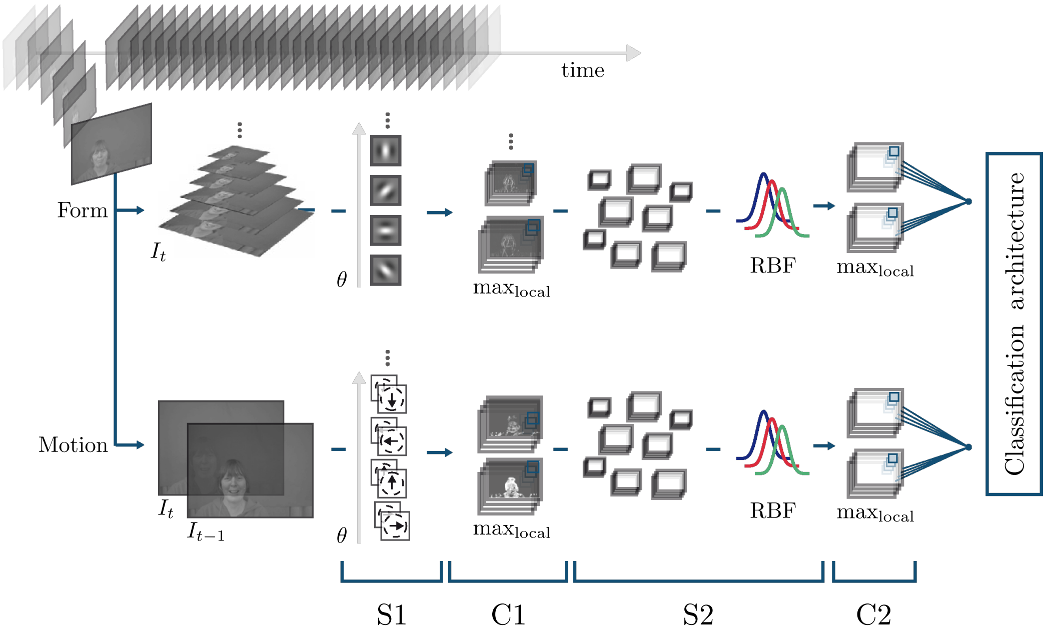


Figure 4.13. Visual features extracted from the AVEC 2011 Dataset. The feature extraction models two biologically-inspired mainly independent pathways: a form and shape pathway and a motion pathway. For a detailed description please refer to the text.

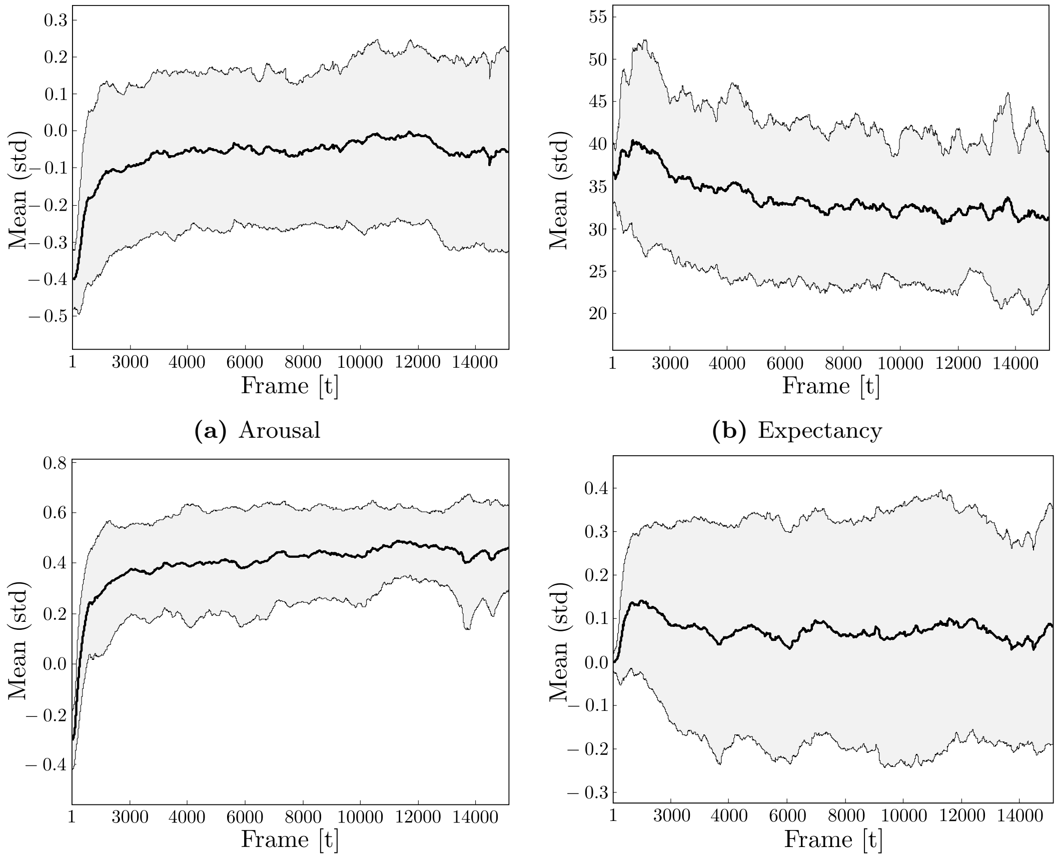
in higher states of processing (Rolls, 1994). The form pathway is basically orientation selective and combines activities to build representations of localized features and shape configurations. The motion pathway on the other hand is direction selective and combines activities to build representations of flow discontinuities as well as motion patterns. It is assumed that the hierarchical processing in both pathways is organized in a similar fashion. The features processing architecture is shown in Figure 4.13. The architecture is a modified variant of the object-recognition model proposed by (Mutch and Lowe, 2008; Riesenhuber and Poggio, 1999; Serre et al., 2005). Motion and form features are processed along two separate pathways, composed of alternating layers of filtering (S) and non-linear pooling (C) stages. In layer S1, different scale representations of the input image are convolved with two-dimensional Gabor filters of different orientations (form path) and a spatio-temporal correlation detector which is used to build a discrete velocity space representation (motion path). Layer C1 cells pool the activities of S1 cells of the same orientation (direction) over a small local neighborhood and two neighboring scales and speeds, respectively. The layer S2 is created by a simple template matching of patches of C1 activities against a number of prototype patches. These prototypes are randomly selected during the learning stage (Mutch and Lowe, 2008). In the final layer C2, the S2 prototype responses are again pooled over a limited neighborhood and combined into a single feature vector which serves as input to the successive classification stage. A more detailed description about the extraction of the feature set can be found in (Glodek et al., 2011d).

The second feature set was extracted using the well-known computer expression recognition toolbox (CERT) (Littlewort et al., 2011). The toolbox provides recognition modules for emotion-related facial features such as action units or basic emotions, but also general attributes such as gender or glasses. The output of modules “Basic Emotions 4.4.3”, “FACS 4.4”, “Unilaterals” and “Smile Detector” were extracted and concatenated which resulted in an overall 36-dimensional feature vector per frame. The tracking of prominent facial markers which are needed for the extraction of the features requires a successful detection of the face. However, the detection of face occasionally failed because of the unrestricted recording settings, e.g. subjects may turn away or leave the visual range of the camera. In about 8% of the video frames the face was not recognized such that no features are available for these frames.

4.1.3.3. AVEC 2012 Dataset

The follow-up event to the AVEC 2011 was the AVEC 2012[1]. The new task extended the previous challenge by replacing the binarized labels with continuous labels. The challenge was composed of two sub-challenges: the word-level sub-challenge (WLSC) and the fully continuous sub-challenge (FCSC). It was up to the participants to use either audio, video or both modalities. The winner of the challenge was determined by comparing the absolute value of the averaged correlation coefficients between the prediction and the ground truth label of each sessions. Figure 4.14 shows the means and standard deviations of the four continuous labels plotted against the video frames which were sampled at 50 Hz. The number of displayed frames was restricted to the median number of frames over all sessions in order to handle the varying length of the recordings. The labels have clear temporal characteristics with comparably low standard deviations. For instance, Figure 4.14a and Figure 4.14c show a quasi-logarithmic curve. The reason for these remarkable curve shapes can most probably be related to the utilized annotation tool Freetrace (Cowie et al., 2000) which, by default, initializes at a fixed label position. It is an open question, whether the prominent curve shapes superimpose the annotated emotions (Schels et al., 2013a). An impression about the strength of the superimposition can be gained by evaluating the correlation to the most obvious function, i.e. logarithmic function $\log(t)$ (Glodek et al., 2012c). Table 4.5 shows the absolute correlation coefficients (CC) of the baseline and the challenge winner and compares them to the absolute correlation coefficients derived using the logarithmic function and the annotated label computed according to the challenge guidelines. Since the logarithmic function was not learned, the results on the training set and development set are as expressive as the results on test set. The best results are marked using a **bold** typesetting.

[1] `http://sspnet.eu/avec2012/` (last visited 10/06/2015).



(a) Arousal

(b) Expectancy

(c) Power

(d) Valence

Figure 4.14. Real-valued labels over time in the AVEC 2012 Dataset. Average value and standard deviation of the continuous labels plotted against the frames. Plots are truncated to show only the median length of all frames. The gray area represents the standard deviation. Continuous labels taken from the training set and the development set.

The labels “Arousal” and “Power” which were already identified to have a logarithmic-inspired curve correlate very well with the logarithmic function. The label “Valence” has a low correlation with the function. Comparing the outcome with the best baseline results by Schuller et al. (2012), it became obvious that all efforts making use of pattern recognition to predict the continuous labels can easily be outperformed by exploiting the labels’ temporal artifacts. The most impressive performance of the logarithmic function is achieved in the FCSC with respect to the label “Power” which has an CC of 0.488. Opposed to that, the corresponding baseline achieved only an CC of 0.138. The results using the logarithmic functions in the WLSC with respect to the labels “Expectancy” and “Power” even outperform the AVEC 2012 winner Savran et al. (2012) with correlations of 0.304 and 0.302, respectively. However, the performance of the logarithmic function is generally lower than the winners of the AVEC 2012. The discovered artifacts do not put the dataset in question. As to be seen from the Figure 4.14, all label dimensions saturate after a certain amount of time once the annotators became familiar with the annotation tool. The findings rather show the

Table 4.5. Results of the challenge winner and the logarithmic function on the AVEC 2012 Dataset. The absolute correlation coefficients (CC) are obtained by taking the absolute value of the averaged correlation coefficients of each recording. Baseline and challenge winner results on the test set. Performance of logarithmic function based on the training set and development set.

	Arousal	**Expectancy**	**Power**	**Valence**	**Mean**
Dataset	↑CC	↑CC	↑CC	↑CC	↑CC
Baseline					
FCSC Test	0.149	0.110	0.138	0.146	0.136
WLSC Test	0.103	0.105	0.066	0.111	0.096
Logarithmic function					
FCSC Train	0.417	0.217	0.559	0.084	0.319
WLSC Train	0.303	0.258	0.411	0.006	0.245
FCSC Development	0.393	0.239	0.510	0.023	0.291
WLSC Development	0.327	0.250	0.323	0.116	0.254
FCSC Test	0.377	0.250	0.488	0.080	0.299
WLSC Test	0.230	**0.304**	**0.302**	0.044	0.220
Challenge winner					
FCSC Nicolle et al. (2012)	**0.612**	**0.314**	**0.556**	**0.341**	**0.456**
WLSC Savran et al. (2012)	**0.302**	0.194	0.293	**0.331**	**0.280**

difficulties to annotate affective states.

Analogously to the AVEC 2011, a large set of pre-computed features was provided. However, we extracted a custom set of features for both modalities. The studies in this work which are using the AVEC 2012 utilize the audio feature set as introduced in Section 4.1.3.2 and the set of high-level video features extracted with CERT as described in Section 4.1.3.2.

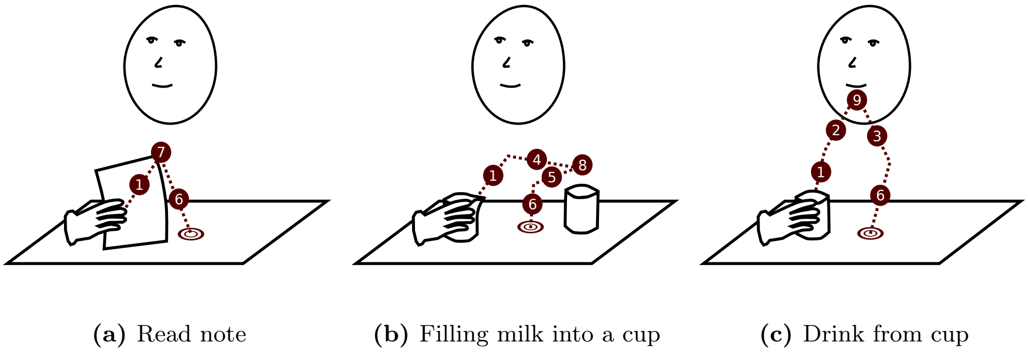
4.1.4. Layered Architectures Datasets

Two datasets were created in order to study the performance of the layered architectures presented in Section 3.7. Dataset and layered architecture form a close relationship such that many design decisions have to be set beforehand. According to the proposed architecture, layers of classes with increasing complexity need to be provided by the dataset. The uppermost layer must be composed of complex entities which require the use of probabilistic logical rules.

The first dataset which is named *Office Layered Architecture Dataset* (OLAD) is situated in an office scenario and comprise three layers. The idea is to recognize basic entities close to the sensors and to propagate the predictions to the higher layers which are more abstract. The final layer needs to be realized by a probabilistic symbolic AI component to infer the complex entities. The information flow of the first dataset is designed to be unidirectional. The second dataset which is called UUlm Generic Layered Architecture Dataset (UUlmGLAD) will be used to study the bidirectional exchange of information between the recognizers of basic and complex entities which was proposed in the BLA.

4.1.4.1. Office Layered Architecture Dataset

The *Office Layered Architecture Dataset* (OLAD) is based on recordings in which a subject performs a set of activities in front of a desk. The recorded patterns allow a decomposition into three layers in order to demonstrate the composition of basic entities to complex entities. The main goal was to create a meaningful and challenging dataset which incorporates adverse conditions, e.g. similarities between movements. The first layer is composed of the basic elementary actions: (1) "Pick up object" (PUP_1), (2) "Move object towards head" (MTH_1), (3) "Move object from head" (MFH_1), (4) "Move object towards table" (MTT_1), (5) "Move object from table" (MFT_1), (6) "Lay back object" (LBA_1), (7) "Manipulate object in hand" (HAM_1), (8) "Manipulate object at table" (TAM_1) and (9) "Manipulate object close to head" (HEM_1). The activities in the second layer are (1) "Drink from cup" (DC_2), (2) "Fill milk into a cup" (MC_2), (3) "Read note" (RN_2), (4) "Stir cup" (SC_2), (5) "Add sugar and stir" (SSC_2) and (6) "Write note" (WN_2) and are composed of a sequence of the first layer's actions. The activities are performed by manipulating the objects: "Cup", "Can", "Paper", "Spoon" and "Pencil". In addition to these five classes, the class "Bare hand" was annotated. Figure 4.15 illustrates how sequences of actions compose the activities "Read note", "Fill milk into a cup" and "Drink from cup". For instance, the activity "Read note" is composed of three actions, i.e. "Pick up object", "Manipulate object in hand" and "Lay back object". Certain activities are likely to be confused such as "Read note" and "Write note" since both comprise the same sequence of action patterns. Furthermore, activities may also have partially identical patterns such as "Write note" and "Stir cup" where in both sequences the subject raises the spoon to the mouth after stirring the cup. The sequence of recognized activities is used as the input to the third layer in which the user is categorized according to his preference: (1) "Black coffee" (BC_3), (2) "White coffee" (WC_3), (3) "Sweet coffee" (SC_3) and (4) "Sweet white coffee" (SWC_3). These categories follow rules, but can be encoded in



(a) Read note **(b)** Filling milk into a cup **(c)** Drink from cup

Figure 4.15. Schematic drawing of the decomposition of activities into actions in the Office Layered Architecture Dataset. Three examples of activities segmented into actions: (1) "Pick up object", (2) "Move object towards head", (3) "Move object from head", (4) "Move object towards table", (5) "Move object from table", (6) "Lay back object", (7) "Manipulate object in hand", (8) "Manipulate object at table" and (9) "Manipulate object close to head".

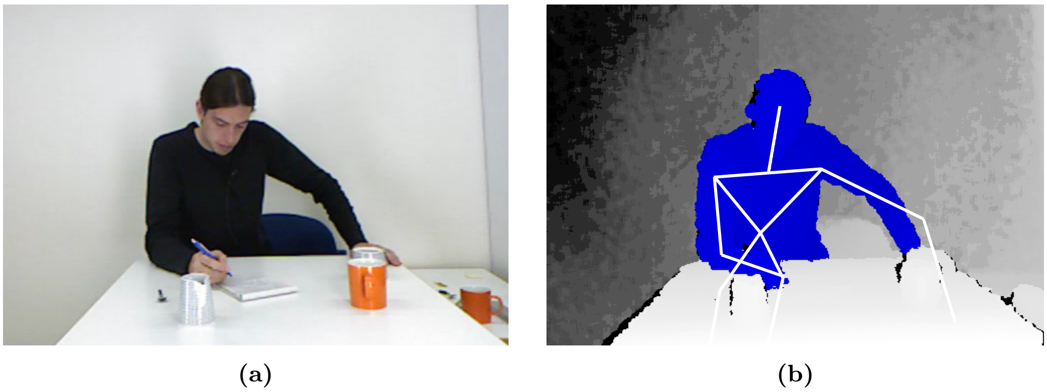


Figure 4.16. Scene from the Office Layered Architecture Dataset. (a) The RGB image shows the subject performing the activity “Write note” (WN_2) (b) The corresponding depth map shows the tracked user (colored in blue) overlaid with the fitted skeleton model (white lines).

sequence of activities with variable length. The activities “Read note” and “Write note” are not related the user preference categories and, therefore, can be repeated arbitrary often.

Figure 4.16 shows an exemplary frame of the dataset. The recordings were made using the Kinect™ camera. The video stream was sampled down to 20 Hz and the audio signal was recorded with 16 kHz and 32 bit depth. Figure 4.16a depicts the RGB image. The subject sits at the desk on which objects are placed while performing a sequence of activities. In the displayed frame, the subject performs the activity “Write note” (WN_2) and, hence, uses the pencil to write a note on a paper block. Figure 4.16b shows the corresponding depth map in which the tracked subject is colorized in blue and overlayed by the fitted skeleton.

The region of interest with the currently used object was determined by utilizing the image coordinates of the skeleton’s right hand. Based on this region of interest, a fixed-sized RGB sub-image was extracted. Although this method was quite accurate, the extracted image occasionally contained other objects located close to the hand, e.g. the milk can in Figure 4.16a. The recognition of objects was further complicated by fast movements of the hand which resulted in blurred sub-images or a minor delay in the skeleton fitting such that a wrong region of the image was extracted.

Three feature sets containing (1) histograms of orientation gradients (HOG) (Freeman and Roth, 1995), (2) histograms of colors (HOC) (Swain and Ballard, 1991) and (3) MFCC (Huang et al., 2001) were extracted from the sub-images of the hand and the audio channel The HOG was based on a grid of 2×2 and eight bins, each one covering 45°. Furthermore, the HOG was normalized by the sum over all bins. The HOC was created using the HSV color space and uses a number of ten bins for the hue dimension and three bins for the saturation dimension. The value dimension was discarded to realize light invariance. The MFCC (Huang et al., 2001) was extracted using a 20 ms window with an offset of 10 ms. Furthermore, the energy of the audio signal was extracted and utilized for silence detection.

Two recording sessions were conducted. The data of the first recording session were

Table 4.6. Statistics on the Office Layered Architecture Dataset. Number of samples and average length in seconds for all three layers: actions, activities and user preferences. For a detailed description please refer to the text.

Layer 1: Actions		#	Avg. length	Layer 2: Activities		#	Avg. length
Pick up object	(PUP_1)	334	0.89±.37 s	Drink from cup	(DC_2)	54	4.54±.38 s
Move object towards head	(MTH_1)	164	0.54±.18 s	Fill milk into a cup	(MC_2)	53	4.66±.81 s
Move object from head	(MFH_1)	164	0.38±.12 s	Read note	(RN_2)	57	5.41±1.26 s
Move object towards table	(MTT_1)	285	0.66±.22 s	Add sugar and stir	(SSC_2)	62	9.13±1.28 s
Move object from table	(MFT_1)	285	0.46±.21 s	Write note	(WN_2)	60	1.9±3.17
Lay back object	(LBA_1)	334	0.95±.25 s	Layer 3: User Preferences		#	Avg. length
Manipulate object in hand	(HAM_1)	176	1.29±1.42 s	Black coffee	(BC_3)	20	24.15±9.12 s
Manipulate object at table	(TAM_1)	288	2.91±2.73 s	White coffee	(WC_3)	20	41.94±10.49 s
Manipulate object close to head	(HEM_1)	164	0.95±.44 s	Sweet coffee	(SC_3)	20	39.11±9.24 s
				Sweet white coffee	(SWC_3)	20	51.21±15.29 s

cut into pre-segmented sequences which were used for training and testing the action and activity layers. The data of the second recording session were used for the evaluation of the third layer. Table 4.6 shows the number of the samples and the average length in seconds of all layers. In total, the first and the second layer consist of 2194 and 334 sequences, respectively. The third layer contains 20 randomly generated sequences for each category of user preferences which are based on the second recording session. The first recording session was partitioned into 10×10 cross-validation folds. The second recording session was used only for testing.

4.1.4.2. UUlm Generic Layered Architecture Dataset

The *UUlm Generic Layered Architecture Dataset* (UUlmGLAD) was recorded in the same setting as the UUlmMAD which was presented in Section 4.1.2.2. A table with a sport bag containing a dumbbell, a bottle and a towel was placed in front of the subject. Nine actions were performed in this setting: putting one of the three objects onto the table, returning one of the three objects to the bag and manipulating one of the three objects, i.e. “Lifting the dumbbell once”, “Drinking from the bottle” or “Wiping the face with the towel”. The sequences of actions were prompted on a monitor screen for the subjects to follow and were based on a fixed set of rules:

1. All object have to be placed in the bag in the beginning and at the end of the recording.
2. Objects are taken out of the bag only once.
3. Objects can be manipulated only if they are already placed on the table.
4. The bottle and towel can be used only after the dumbbell was used.
5. The bottle (or the towel) has to be used if the dumbbell was used twice.
6. The towel has to be used after the last use of the dumbbell.

The rules can be used to identify complex entities, e.g. beginning, course and end of the training. Figure 4.17 shows a representative video frame taken from the recording. The

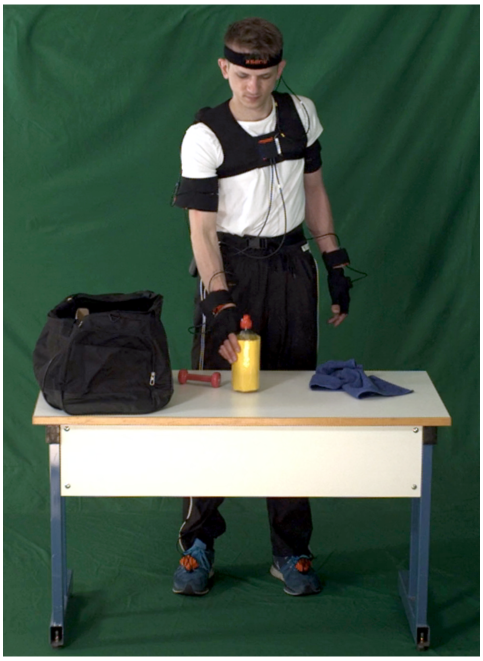


Figure 4.17. Scene from the UUlm Generic Layered Architecture Dataset. The subject is allowed to perform nine different actions using three objects, i.e. putting an object onto the table, returning an object to the bag or manipulating an object where the object can be a dumbbell, a bottle or a towel. The order of actions is randomized, but follows rules. In the scenery, the subject has already placed all objects onto the table and is about to perform the action “Drinking from the bottle”.

subject has placed the dumbbell, bottle and towel onto the table and is in act of grabbing the bottle in order to perform the action “Drink from the bottle”.

According to the rules, objects are either in the bag, on the table or in the hand of the subject, depending on the actual and past actions. As a result, every contradicting information generated from the sub-symbolic recognizers should be suppressed. Furthermore, the temporal order of actions can help to correct sub-symbolic and symbolic recognitions. An object which is assumed to be located in the bag cannot be involved in an action in which the object is manipulated or returned to the bag. Hence, it is possible to identify sub-symbolic recognitions which are contracting the symbolic rules. However, it is possible to revise symbolic decisions in the past.

The dataset was recorded using three video cameras and a motion capturing suit which were calibrated to each other in a post-processing step. In a first step, an intrinsic camera calibration, an extrinsic calibration across the cameras and a fitting of the three-dimensional motion capturing data to the image plane were performed. The camera calibration was achieved by using a separate recording of a calibration pattern which was made prior to the beginning of each recording session. The camera geometry was determined using standard computer vision approaches (Hartley and Zisserman, 2003). The fitting of the motion capturing data was performed by manually labeling a subset of prominent skeleton model joints in three frames in each camera view. These labels and the extrinsic calibration were used to derive the three-dimensional locations of the joints in the world coordinate system. In the last step, the motion capturing skeleton model was fitted to the three-dimensional locations of the joints. The procedure is visualized in Figure 4.18 using a frame from the UUlmMAD which shows the subject p28 performing the action “Dumbbell frontal raise” (SP7). The

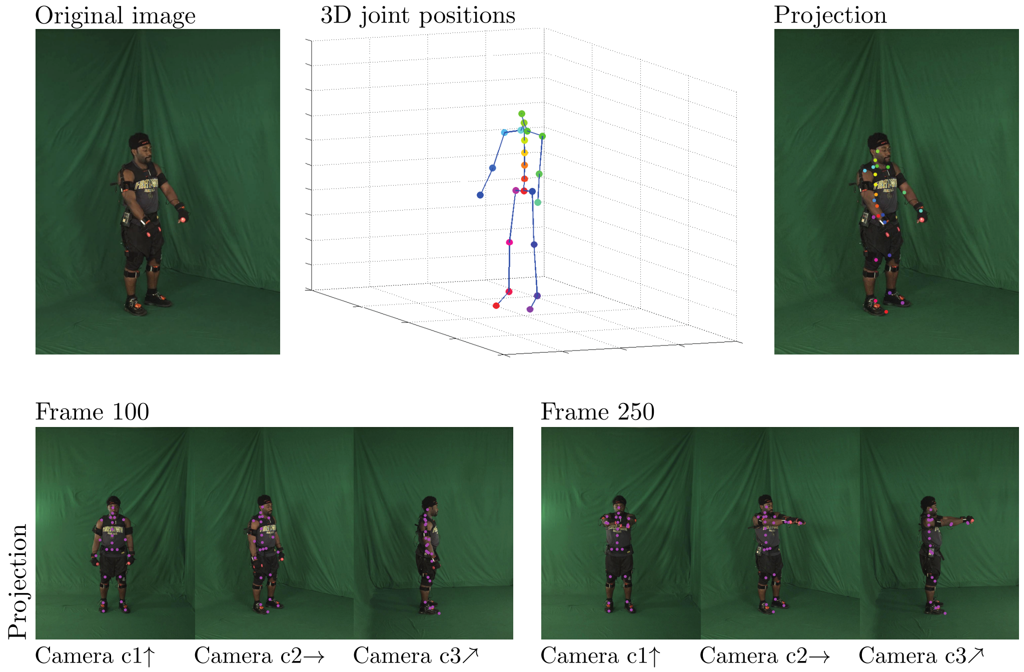
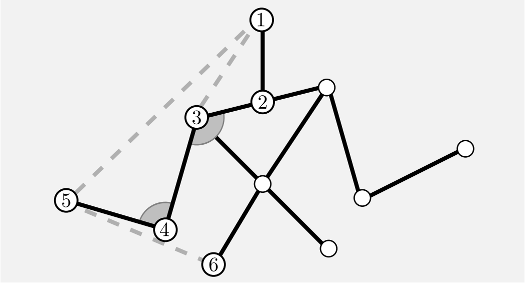


Figure 4.18. Projection of the motion capturing skeleton model into the camera image plane in the UUlm Multi-perspective Action Dataset. The upper part of the figure shows the original image, the motion capturing skeleton model and the projection of the skeleton model into the camera image plane. The lower part of the figure depicts two frames of the action "Dumbbell frontal raise" (SP7) performed by subject p28 overlayed with the post-processed skeleton model projection.

Figure 4.19. Features extracted from the skeleton model of the UUlm Generic Layered Architecture Dataset. The extracted feature vector has five elements containing distances and angles. The three distances marked by a dashed gray line: (1) head – hand, (2) head – shoulder and (3) hip – hand. The two angles marked by a gray sector of a circle: (1) neck – shoulder – elbow and (2) shoulder – elbow – hand.



upper part of the figure shows from left to right: the original image, the three-dimensional joints as provided by the motion capturing and the original image overlayed with the skeleton model after the fitting process. The lower part of the figure shows the action from all three camera perspectives at two different times. The left example is taken from frame 100 where the subject stands motionless in the neutral position. The right example is taken from frame 250 and shows the subject performing the movements of the action, i.e. the subject holds the dumbbell in his hands and his arms are passing each other with maximal speed. The displayed frame of the last example reveals a small lag in the motion capturing data. In camera c3, the skeleton model joints of the arms are running slightly after the actual location of the arm. However, with regard that both data originate from independent sources the calibration of the motion capturing data and the cameras worked very exact.

A sub-image of 96×96 pixel was specified for each frame based on the image coordinate of the right hand. For all sub-images, HOG and HOC features were extracted. The HOG features were based on a grid of 3×3 and eight bins each one covering an angle of 45°. The bins were subsequently normalized. The HOC made use of the HSV color space and utilized a number of eight bins for the hue dimension and four bins for the saturation dimensions. Again, the value dimension was discarded for light invariance. The skeleton model was used to extract a feature with five elements, i.e. three distances: (1) head – hand, (2) head – should and (3) hip – hand; and two angles: (1) neck – shoulder – elbow and (2) shoulder – elbow – hand. Figure 4.19 illustrates the extracted features using a schematic drawing of the skeleton model's torso. The three distances are marked by dashed gray lines and the two angles are marked by a gray sector of a circle.

4.2. Evaluation Criteria

Two kinds of measures will be presented in the following sections which are used to assess the performance of the proposed methods: (1) performance measures for crisp classifier decisions and (2) performance measures for density estimations.

4.2.1. Performance Measures of Crisp Classifier Predictions

The most frequently used performance measures in machine learning are based on the ratios derived from the confusions between the ground-truth classes and the predictions of a classifier. Given a dataset of N test samples, a classifier can be examined by determining the classifier's prediction for each sample and comparing it to the ground-truth label. Correct class assignments and errors are binned into the four categories of a confusion matrix, as shown in Figure 4.20. The four bins of the confusion matrix are: true positive n^{tp}, false negative n^{fn}, false positive n^{fp} and true negative n^{tn}. Different measures can be derived from the confusion matrix. A popular measure is the classifier *accuracy* which is computed by

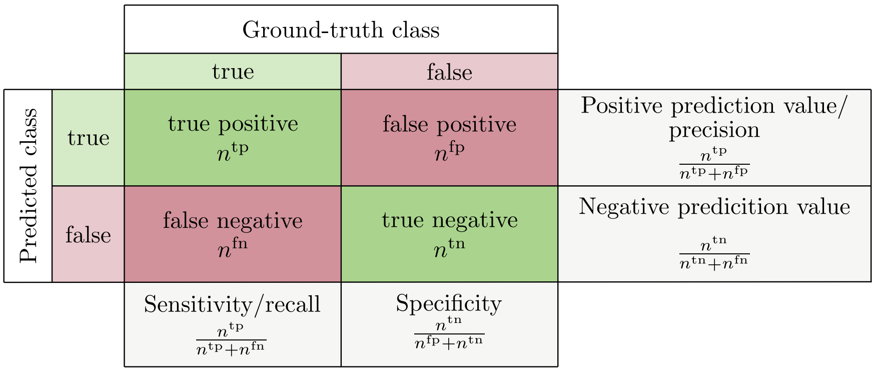
Accuracy and error rate

$$\text{accuracy} = \frac{n^{\mathrm{tp}} + n^{\mathrm{tn}}}{N}.$$

An alternative measure to the accuracy is the classification error which describes the fraction of misclassified samples:

$$\text{error} = 1 - \text{accuracy} = \frac{n^{\mathrm{fp}} + n^{\mathrm{fn}}}{N}.$$

Both, the accuracy and the *error rate*, are probably the most frequently used performance



		Ground-truth class		
		true	false	
Predicted class	true	true positive n^{tp}	false positive n^{fp}	Positive prediction value/ precision $\frac{n^{\mathrm{tp}}}{n^{\mathrm{tp}}+n^{\mathrm{fp}}}$
	false	false negative n^{fn}	true negative n^{tn}	Negative predicition value $\frac{n^{\mathrm{tn}}}{n^{\mathrm{tn}}+n^{\mathrm{fn}}}$
		Sensitivity/recall $\frac{n^{\mathrm{tp}}}{n^{\mathrm{tp}}+n^{\mathrm{fn}}}$	Specificity $\frac{n^{\mathrm{tn}}}{n^{\mathrm{fp}}+n^{\mathrm{tn}}}$	

Figure 4.20. Confusion matrix with different measures to rate the performance of classifiers. The matrix lists four types of errors which can be used to derive further performance measures, such as the recall, specificity, precision or the negative prediction value.

Precision and recall

measures for classifiers. However, these measures capture only a part of the actual performance of a classifier. For instance, the class distribution of a dataset might be unbalanced, i.e. samples of one class are more frequent. If the examined classifier turns out to be unilateral and favors the majority class, the accuracy and error rate will not be able to reveal this classifier deficit. Hence, it is beneficial to take a closer look at additional performance measures such as the *precision* and the *recall* which are given by

$$\begin{aligned} \text{precision} &= \frac{n^{\mathrm{tp}}}{n^{\mathrm{tp}} + n^{\mathrm{fp}}} \\ \text{recall} &= \frac{n^{\mathrm{tp}}}{n^{\mathrm{tp}} + n^{\mathrm{fn}}}. \end{aligned}$$

The precision is the ratio between the number of correctly predicted positive samples and the overall number of samples assigned by the classifier to the positive class. The recall describes the share of true positives and the overall number of samples labeled with *true*. The interpretation of which class is regarded as true and false depends on the viewpoint of the experimenter. The specificity and the negative prediction value are the negative counterparts of the precision and recall. In order to perform a comprehensive evaluation of the performance of a classifier, the precision and the recall should be provided for all available classes in the dataset. However, it might be confusing to provide four additional performance measures for a two-class problem. The F_1*-measure* can be used to reduce the number of performance measures and is given by

F$_1$-measure

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

The F$_1$-measure is defined as the weighted average of precision and recall and, thus, indicates the share of correct and false assignments. The examination of the F$_1$-measure for every class of the dataset provides a good intuition of the recognition balancedness.

4.2.2. Performance Measures of Density Estimation

The performance of probability density algorithms will be assessed in two ways: a classification scenario and the well-known bias-variance decomposition (Bishop, 2006, page 147ff.).

In case of the classification scenario, a probability density is determined for each class of a given classification problem. The densities are then utilized as parts of an NBC. The classifier performance obtained when testing on a disjunct test set serves as a performance measure for the estimated densities.

Bias-variance decomposition

The second way to rate the performance of a density estimation algorithm is to use the *bias-variance decomposition*. The starting point to derive the bias-variance decomposition is the expected squared loss function given by

$$\mathbb{E}(L) = \sum_{n=1}^{N} \{y(\mathbf{x}_n) - h(\mathbf{x}_n)\}^2$$

where $L(a) = a^2$ is the squared loss, the function $y(\cdot)$ represents the estimated density, $h(\cdot)$ is the target function which is assumed to be free of noise, and $\{\mathbf{x}_n\}_{n=1}^{N}$ are the data samples. The goal is to asses the uncertainty of models where the different models are given by their parameter configurations. Therefore, the loss function is expanded by the average over a specific model $\mathcal{M}$ and by adding and subtracting $\mathbb{E}_{\mathcal{M}}[y(\mathbf{x};\mathcal{M})]$:

$$\begin{aligned}
\{y(\mathbf{x}) - h(\mathbf{x})\}^2 &= \{y(\mathbf{x};\mathcal{M}) - \mathbb{E}_{\mathcal{M}}[y(\mathbf{x};\mathcal{M})] + \mathbb{E}_{\mathcal{M}}[y(\mathbf{x};\mathcal{M})] - h(\mathbf{x})\}^2 \\
&= \{y(\mathbf{x};\mathcal{M}) - \mathbb{E}_{\mathcal{M}}[y(\mathbf{x};\mathcal{M})]\}^2 + \{\mathbb{E}_{\mathcal{M}}[y(\mathbf{x};\mathcal{M})] - h(\mathbf{x})\}^2 \\
&= 2 \cdot \{y(\mathbf{x};\mathcal{M}) - \mathbb{E}_{\mathcal{M}}[y(\mathbf{x};\mathcal{M})]\}\{\mathbb{E}_{\mathcal{M}}[y(\mathbf{x};\mathcal{M})] - h(\mathbf{x})\}.
\end{aligned}$$

Once the term is expanded, the expectation of the expression with respect to $\mathcal{M}$ is taken. As a result, the cross-term vanishes and the final decomposition is obtained:

$$\mathbb{E}_{\mathcal{M}}[\{y(\mathbf{x}) - h(\mathbf{x})\}^2] = \underbrace{\{\mathbb{E}_{\mathcal{M}}[y(\mathbf{x};\mathcal{M})] - h(\mathbf{x})\}^2}_{\text{bias}^2} + \underbrace{\mathbb{E}_{\mathcal{M}}[\{y(\mathbf{x};\mathcal{M}) - \mathbb{E}_{\mathcal{M}}[y(\mathbf{x};\mathcal{M})]\}^2]}_{\text{variance}}.$$

Squared bias and variance

The expected squared loss function comprises two terms. The first term is referred to as the *squared bias* and represents the extent to which the average prediction differs from the desired density function. The second term which is called *variance* measures the extent to which the solution for individual models vary around their average. The reader will find further information on the bias-variance decomposition in the book written by Bishop (2006, page 147ff.).

Chapter 5

Empirical Evaluations

The current chapter presents the results of the empirical evaluations based on the proposed methods and datasets introduced in the previous chapters. The organization of this chapter follows the same structure as Chapter 4 addressing the datasets. Each study in this chapter is followed by a summary which outlines the main findings. A detailed discussion of the results will be provided in Chapter 6. The best results are marked in the tables by using a **bold** typesetting.

The evaluation begins with Section 5.1 which presents the results of the EGMM and related approaches. Section 5.2 provides the action recognition results to assess the performance of the HMM-GPD and CHMM-GPD. The results of the multimodal temporal fusion algorithms are presented in Section 5.3 and Section 5.4. Finally, Section 5.5 provides the results of the study performed using the ULA.

5.1. Ensemble GMM for Enhanced Density Estimation

The EGMM which was introduced in Section 3.1 promises to be more robust in terms of accuracy and less sensitive for singularities (Glodek et al., 2013c). Furthermore, the EGMM aims at reducing the effort in finding the best performing parameter configuration compared with conventional GMM. Figure 5.1 exemplifies the advantage which can arise from using the EGMM. Figure 5.1a shows the density of an exponential decay, i.e. $p(x) = \exp(-x)$, together with 250 data samples drawn from the corresponding distribution which was utilized for the training of the density models. Two individual GMM, one with two mixture components and the other with three mixture components, were derived from the data samples. Their densities are plotted in Figure 5.1b using dashed curves. The EGMM which was computed using the unweighted average of both individual GMM is depicted by the solid red curve. The

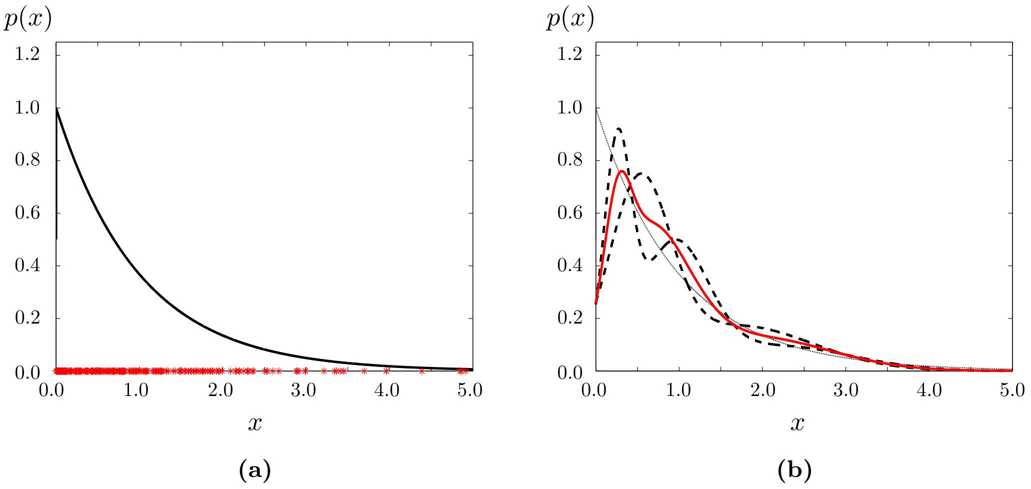


Figure 5.1. Comparison of classic GMM and EGMM using the example of a distribution generated from the exponential decay. (a) Shows the probability density (solid line) from which the data samples are derived (x-marks at the bottom of the plot). (b) The data samples are utilized to estimate two Gaussian mixture models with two and three mixture components (dashed lines). Both models give a good approximation of the original density. However, the EGMM created by averaging the two GMM (red line) has a closer fit to the original function.

accuracies of the three densities were measured by comparing the estimates to the ground truth function using the integral of the absolute difference ranging in $[0, 5]$. The GMM with two mixture components and the GMM with three mixture components resulted in an error of 0.29 and 0.22, respectively. The EGMM rendered a lower error of 0.19. Although the example is constructed, it shows that especially for non-Gaussian distributions a clear improvement in performance can be expected. In the following, we will fall back on the datasets already described in Section 4.1.1.

5.1.1. Ormoneit Dataset I and Ormoneit Dataset II

The first experiment examines the two artificial classification problems ODI and ODII. Ormoneit and Tresp (1998) evaluated six density esimtation approaches: (1) standard maximum likelihood; (2) simple averaging of multiple identical models trained on the same data; (3) subset averaging (resample from the dataset without replacement) of multiple identical models; (4) bagging (resample from the dataset with replacement); (5) penalized likelihood (which adds a regularizing term in form of conjugated priors to the log-likelihood function); and (6) a Bayesian approach which makes use of conjugated priors in combination with Gibbs sampling. The last two approaches regularize the estimation of the GMM and, therefore, it is difficult to related them to the EGMM presented in this work. All GMM used in the experimental evaluation have 20 mixture components. In case of subset averaging and bagging, the data were resampled by drawing 70% from the original data without and

Method	**Ormoneit Dataset I** ↑Acc.	**Ormoneit Dataset II** ↑Acc.
Maximum likelihood†	79.03±2.6 %	72.59±0.2 %
Simple averaging†	80.43±2.0 %	76.55±0.2 %
Subset averaging†	80.73±1.8 %	76.39±0.2 %
Bagging†	81.20±1.4 %	76.65±0.2 %
Penalized likelihood†	82.28±1.4 %	**81.70**±0.2 %
Bayesian†	82.70±1.3 %	64.76±0.9 %
EGMM	**84.55**±2.1 %	78.80±0.2 %
EGMM ($M = 25$)	**84.77**±2.5 %	77.40±0.2 %

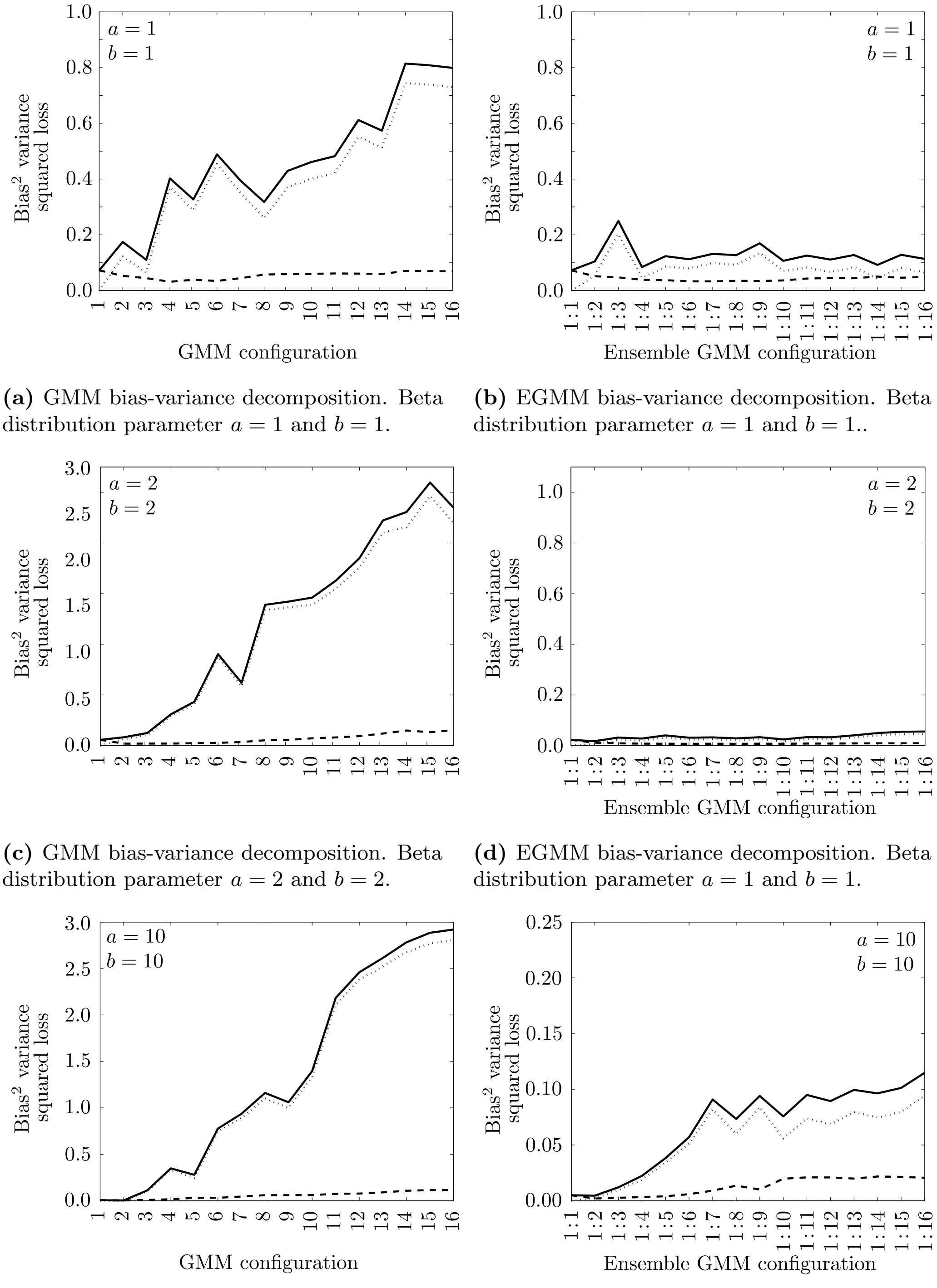
Table 5.1. Accuracies of the EGMM on the Ormoneit Dataset I and Ormoneit Dataset II in comparison to the methods proposed by Ormoneit and Tresp (1996). Results marked by (†) indicate the best performance achieved in the study of Ormoneit and Tresp (1996). All values averaged with standard deviation. Arrows indicate how to rate the measures.

with replacement, respectively. The number of replications was chosen to be 50 according to Breiman (1996).

The EGMM was analyzed in two settings: (1) using all evaluated GMM configurations and (2) using only the 25 best GMM rated against the model likelihood (pruned EGMM). The numbers of mixture components ranged from 5 to 14 with five replications resulting in 50 GMM configurations. Resampling was additionally performed by drawing 70% of samples from the original dataset without replacement.

The mean classification accuracies of all approaches together with the corresponding standard deviation over 20 simulations are presented in Table 5.1. The left part of the table shows the results of the ODI which comprises two-dimensional samples: Ormoneit and Tresp (1998) reported an accuracy of 79.03% for the standard maximum likelihood approach. The accuracies of the averaging and bagging approaches ranged from 80.43% to 81.20%. The penalized likelihood and the Bayesian approach using a conjugated prior achieved accuracies of 82.28% and 82.70%, respectively. The EGMM method achieved an accuracy of 84.55% and the pruned EGMM using only the 25 best GMM achieved the highest accuracy of 84.77%.

The results of the ODII are presented in the right part of the table. The ODII is based on ten-dimensional samples and, therefore, is clearly more challenging. Applying the classic maximum likelihood approach to ODII resulted in an accuracy of 72.59%. The accuracies of the averaging and bagging approaches ranged from 76.39% to 76.65%. The penalized likelihood achieved the highest accuracy of 81.70%. The Bayesian approach using the conjugated prior achieved only an accuracy of 64.76%. The EGMM and the pruned EGMM approaches had the second and third highest results with accuracies of 78.8% and 77.4%.



(a) GMM bias-variance decomposition. Beta distribution parameter $a = 1$ and $b = 1$.

(b) EGMM bias-variance decomposition. Beta distribution parameter $a = 1$ and $b = 1$..

(c) GMM bias-variance decomposition. Beta distribution parameter $a = 2$ and $b = 2$.

(d) EGMM bias-variance decomposition. Beta distribution parameter $a = 1$ and $b = 1$.

(e) GMM bias-variance decomposition. Beta distribution parameter $a = 10$ and $b = 10$.

(f) EGMM bias-variance decomposition. Beta distribution parameter $a = 10$ and $b = 10$.

Figure 5.2. Bias-variance decomposition of classic GMM and EGMM on the Beta Distribution Data. Bias-variance decomposition for different beta distribution parameters using the classic GMM (a, c, e) and the EGMM (b, d, f). Each plot shows the squared bias (dashed line), the variance (dotted line) and the test error given by the sum of the squared bias and variance (solid line). Please note, that the scale of the Figure 5.2e and Figure 5.2f differ from the other figures.

5.1.2. Beta Distribution Dataset

The second experiment examines the EGMM and classic GMM using the BDD. The experiment analyzes the bias-variance decomposition using the sub-Gaussian beta distributions which were described in Section 4.1.1 and Section 4.2.2. In case of the classic GMM, the models had different number of mixture components ranging from 1 to 16. The EGMM were evaluated with an increasing complexity: The number of mixture components range from one to the index of the current model. Hence, the ensembles $\{\{1\}, \{1, 2\}, \{1, 2, 3\}, \ldots, \{1 \ldots 16\}\}$ were evaluated. The loss function was computed using a numerical integration with equidistant samples generated in the interval $[-0.5, 1.5]$ and a step size of 10^{-3}. For each value, the model was estimated 20 times in order to derive the squared bias and the variance. The results of the bias-variance decompositions are shown in Figure 5.2. The conventional maximum likelihood GMM results are shown on the left-hand side in the Figures 5.2a, 5.2c and 5.2e, while the results of the EGMM are shown in the Figures 5.2b, 5.2d and 5.2f. The plots which show the beta distributions with $a = b = 10$ have different scales on the y-axis. The squared bias (dashed curve) captures the average prediction difference from the desired density function. As to be seen in the figure, the squared bias had over a wide range only a minor contribution to the squared loss (solid line). The variance (dotted curve) which measures the extend to which individual solutions vary around the average generally had a bigger impact to the squared loss. Figure 5.2a, Figure 5.2c and Figure 5.2e show the performance of classic maximum likelihood GMM for different beta distributions starting from a non-Gaussian distribution to distributions which are closer to the Gaussian distribution. For all evaluations, the variance and squared bias increased with higher model complexities. In Figure 5.2a, the squared bias dropped slightly for GMM having around four mixture components. In general, the squared loss significantly increased for all plots. The Figure 5.2b, Figure 5.2d and Figure 5.2f depicting the EGMM showed a clearly different behavior. In contrast to the classic GMM, the squared loss of the EGMM remained stable in case the complexity of the models increased. Regarding the non-Gaussian beta distribution ($a = b = 1$) the squared loss error even started to decrease, mostly due to the low variance.

5.1.3. Summary

The study on EGMM comprise of two experiments which showed that the EGMM provides robust and accurate density estimations. The first experiment compares the EGMM with conventional and partially problem-tailored approaches (Table 5.1). The EGMM is the best performing algorithm on the ODI and the second best performing algorithm on the ODII. The penalized likelihood which had the best performance on the ODII based on a regularized density with a conjugated prior such that the estimations can be regarded as being tailored to the problem setting.

The second experiment examines the robustness of the EGMM approach using the BDD. The classic GMM was compared with the EGMM with the help of the bias-variance decomposition. The results showed that with increasing model complexity, the test errors of the GMM increased, while the test errors of the EGMM remained stable (Figure 5.2). This behavior was further examined by decomposing the test error (squared loss) into the squared bias and variance. It turned out, that for both approaches the squared biases stayed stable. However, the variances of the GMM and EGMM showed different behaviors. The variances of the GMM increased which indicates that the variation between the models was higher for more complex GMM. In contrast to that, the variances of the EGMM did not increase. This proves that the EGMM approach can provide more robust models with lower variations.

5.2. HMM Graph Probability Densities for Action Recognition

This section studies the HMM-GPD using the two datasets presented in Section 4.1.2. First, the results on the OAD are presented followed by the results on the UUlmMAD.

5.2.1. Occlusion Action Dataset

The classification using the OAD was performed by training one HMM-GPD for each of the three actions classes “Drink”, “Eat” and “Read”. The class assignment given a new sample sequence was done by choosing the model with the highest likelihood. The meta-parameters of the HMM, e.g. number of hidden states, shape of the covariance matrix and type of transition matrix, were determined using the inner cross-validation folds.

As already described in Section 4.1.2.1, the training and test sets of the OAD comprise six different occlusion configurations which were introduced in Figure 4.4 and match the letters from a to f. Table 5.2 reports the performances on compilations of occlusion configurations. Two settings were evaluated: (1) a training set without occlusion and (2) a training set with samples of mixed occlusions. The upper half of the table shows the results when performing the training without sequences containing incomplete graphs. In this setting, each HMM-GPD was trained using four hidden states and a full transition and covariance matrix. According to the second setting which is presented in the lower part of the table the models were trained using all occlusion configuration sets. The corresponding HMM-GPD were trained with five hidden states and a full transition and covariance matrices.

In the first setting, the classification performances decreased with the degrees of occlusion. The reference experiment testing the occlusion configuration (a) which was free of any

Table 5.2. Accuracies and $\mathbf{F_1}$-measures for HMM using graph probability on the Occlusion Action Dataset. The first column and second column show the occlusion configurations used for training set and test sets. All values averaged with standard deviation. Arrows indicate how to rate the measures.

Training	**Testing**	↑**Accuracy**	↑$\mathbf{F}_1$(Drink)	↑$\mathbf{F}_1$(Eat)	↑$\mathbf{F}_1$(Read)
a	a	**80.1±2.6 %**	75.4±4.9 %	**79.6±8.6 %**	**86.8±7.6 %**
a	b	78.1±7.5 %	**77.3±11.6 %**	73.6±4.9 %	84.7±9.9 %
a	c	72.2±6.5 %	64.7±5.9 %	79.2±6.6 %	71.2±9.0 %
a	d	70.4±4.2 %	62.9±4.0 %	76.9±2.8 %	70.4±11.3 %
a	e	68.7±3.0 %	63.6±5.3 %	74.5±3.1 %	68.0±5.6 %
a	f	69.0±5.3 %	63.0±10.1 %	70.7±5.0 %	74.2±5.3 %
a	abcdef	72.7±2.7 %	67.6±5.2 %	74.4±3.2 %	77.0±5.6 %
abcdef	a	**81.5±6.8 %**	**79.6±10.2 %**	**81.6±6.8 %**	**83.0±10.6 %**
abcdef	b	71.7±9.2 %	75.7±7.1 %	71.4±7.5 %	65.1±20.1 %
abcdef	c	74.2±7.1 %	69.7±11.1 %	77.6±6.8 %	74.7±8.5 %
abcdef	d	73.2±3.5 %	68.7±7.9 %	77.3±3.6 %	73.3±7.4 %
abcdef	e	69.5±3.3 %	65.8±3.8 %	72.6±3.3 %	70.1±6.7 %
abcdef	f	66.8±4.1 %	63.8±3.9 %	68.9±2.7 %	68.3±12.2 %
abcdef	abcdef	72.8±5.9 %	71.0±7.9 %	74.9±5.4 %	71.4±6.0 %

occlusions led to a three-class classification accuracy of 80.1%. Testing with the occlusion configuration (b), i.e. all edges containing the head node were removed, yielded only in a slightly decreased accuracy of 78.1%. Removing the head node preserved the information of both hands which explains the good performance in the sparse graph. Removing the left or the right hand nodes from the samples of the test set, i.e. occlusion configuration (c) and (d), led to accuracies of 72.2% and 70.4$ being clearly lower than the baseline accuracy of the occlusion configuration (a). In case of occlusion configuration (e) and (f), i.e. the complete left side or right side was removed from the samples of the test set, the three-class accuracies dropped down to 68.7% and 69.0%. The last experiment shows the performance when training on sequences containing exclusively complete graphs and testing on all possible occlusion configurations. The experiment achieved an accuracy of 72.7%.

The lower part of Table 5.2 shows the results of the second experiment in which the training was performed using a mixed sample set containing all sorts of occlusion configurations. Performing the test on the data of occlusion configuration (a) led to the overall highest accuracy of 81.5%. The test set in which the head node and the adjacent edges were removed, i.e. occlusion configuration (b), yielded in a reduced accuracy of 71.7%. The low accuracy can be explained by the low performance of the third class "Read note" which is revealed by the F_1-measures. Occlusion configuration (c) and (d) had slightly increased performances compared with the results of the upper half of the table and achieved accuracies of 74.2% and 73.2%, respectively. Removing a complete side of the skeleton model, i.e. configuration (e) and (f), resulted in the overall lowest accuracies of 69.5% and 66.8%. The last experiment in which training and testing was performed using the mixed set of occlusion configurations led to an accuracy of 72.8%.

In addition to the accuracies, per-class F_1-measures were provided for all experiments. The values differed only slightly which indicates that the recognition was not unilateral, but balanced over all classes. The only exception constituted the setting in which the training was performed on the mixed set of occlusion configuration and testing was conducted on data of occlusion configuration (b). Here, the class “Read note” has a noticeable low F_1-measure.

5.2.2. UUlm Multi-perspective Action Dataset

This section presents three experiments which were conducted using the UUlmMAD. The first experiment serves as a baseline. It provides comparative results by evaluating a conventional approach. The second experiment and third experiment examine the HMM-GPD and CHMM-GPD, respectively.

5.2.2.1. Baseline Experiment

The baseline experiment examines the state-of-the-art architecture which is shown in Figure 5.3 and was proposed by Laptev et al. (2008). The architecture is composed of a feature extraction step including bag of words generation and a classification step. In a first step, the high-resolution RGB videos are converted into gray scale videos having a resolution of 240×320. Prior experiments showed that this conversion is optimal for the given recognition task. A Harris 3D interest point detector was then used to find informative locations in the spatial and temporal domain of a video (Harris and Stephens, 1988; Laptev, 2005). Scale-invariance was achieved by using three different scales where the image size was halved at each scale. The number of detected interest points varied strongly from 240 to 1500 depending on the motions and contours present in the video. Histogram of oriented gradients (HOG) and histogram of optical flow directions (HOF) features were extracted from the patches covering the pixel neighborhoods of the interest point locations. The sizes of these patches were based on the detection scale. Each patch was subdivided into a 3×3 spatial grid and two cubes in the temporal domain. In every cube, a HOG with four bins of orientations and a HOF with five bins of flow directions were computed. The HOG and HOF features were then concatenated to a hybrid histogram of 162 dimensions, i.e. $4 \times 3 \times 3 \times 2 = 72$ HOG features and $5 \times 3 \times 3 \times 2 = 90$ HOF features. Subsequently, the distribution of the hybrid histograms of all detected interest points was transformed into a fixed-length representation using the bag-of-words approach (Laptev et al., 2008). To do so, a codebook using a clustering algorithm was generated. The fixed-length representation was created by identifying the cluster assignments and binning them into a histogram, the so-called signature. The codebook was generated using a training set composed of 200 randomly chosen hybrid histograms from each video. A number of 2000 codebook clusters, i.e. the words, were learned using the k-means algorithm. The signature of a new test recording was

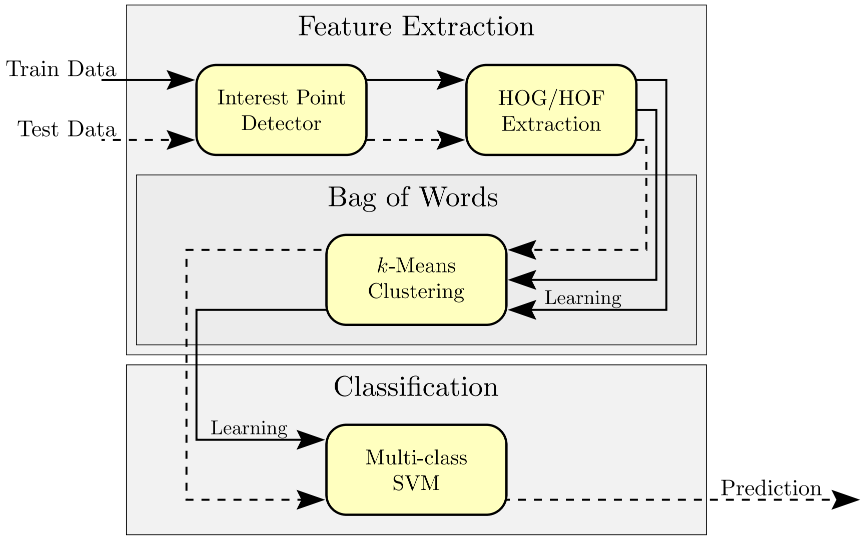


Figure 5.3. State-of-the-art classifier architecture used for the baseline experiment on the UUlm Multi-perspective Action Dataset. The feature extraction was based on an interest point detector. HOG and HOF features were extracted in the neighborhood of these interest points which were then transformed into a bag-of-words representation. The classification was then performed using a multi-class SVM. The solid arrows show the information flow of the training procedure, whereas the dashed line represents the information flow of the test procedure. A partition of the data for training was used to perform the k-means clustering.

created using 200 randomly chosen hybrid histograms. Classification was realized using a one-vs-one L2 soft-margin SVM with linear kernels. The generation of the codebook and the SVM training and the related parameter search was conducted using leave-one-subject-out cross-validation.

Table 5.3 shows the results obtained using four training configurations. The first three training configurations used data from one of the three camera perspectives. The fourth training configuration utilized all three camera views for training. Testing was performed using all three camera perspectives independently.

Table 5.4a shows the results in which the training was performed using the data of the frontal camera c1. Testing on the data from the same camera c1 rendered an accuracy of 99.3% which turned out to be the highest accuracy in the complete study. In this setting, “Hand wave” (EL1) and “Pick up” (EL2) were occasionally confused which is also reflected in the slightly weaker F$_1$-measures of 96% and 97%. This can be related to the fact that both actions have a similar downward motion in common. Testing on the data of the side-view camera c2 resulted in a low accuracy of only 27.7%. The numerous class confusions were caused by the change of the camera perspective. The class “Torso twist” (ST2) having a high F$_1$-measure constituted an exception. The good performance can be explained by the fact that in action ST2, the subjects turns from camera c1 to camera c2. This results prominent and similar movements in both views which in turns led to similar features. The class “Stationary lunge” (SP1) had the second highest F$_1$-measure. It was recognized

Table 5.3. Accuracies and F_1-measures of the baseline architecture on the UUlm Multi-perspective Action Dataset. (a, b, c) Training performed using data derived from only one camera perspective and testing on data derived from all three cameras perspectives. (d) Training and testing performed on data derived from all three cameras perspectives. All values averaged with standard deviation.

(a)

Training		c1		
Testing		c1↑	c2→	c3↗
↑Acc.		**99.3±1.3 %**	27.7±6.5 %	56.4±8.0 %
↑F_1-measures per class	EL1	1.00±.00	0.00±.00	0.87±.29
	EL2	0.96±.11	0.00±.00	0.06±.16
	EL3	0.97±.07	0.01±.06	0.25±.38
	SP1	1.00±.00	0.76±.17	0.91±.12
	SP2	1.00±.00	0.45±.39	0.79±.32
	SP3	1.00±.00	0.20±.31	0.65±.39
	SP4	1.00±.00	0.24±.20	0.82±.15
	SP5	0.99±.03	0.00±.00	0.42±.31
	SP6	0.99±.03	0.01±.08	0.32±.32
	SP7	0.99±.03	0.02±.06	0.10±.17
	ST1	1.00±.02	0.21±.19	0.46±.10
	ST2	1.00±.00	0.91±.20	0.99±.05
	ST3	1.00±.00	0.48±.24	0.64±.08
	ST4	0.99±.03	0.00±.00	0.03±.10

(b)

Training		c2		
Testing		c1↑	c2→	c3↗
↑Acc.		28.9±5.8 %	**96.9±2.8 %**	69.0±6.7 %
↑F_1-measures per class	EL1	0.00±.00	0.94±.14	0.03±.16
	EL2	0.00±.00	0.79±.29	0.33±.31
	EL3	0.00±.00	0.99±.04	0.86±.20
	SP1	0.66±.19	0.99±.04	0.82±.15
	SP2	0.26±.39	0.99±.03	0.68±.34
	SP3	0.38±.36	1.00±.00	0.99±.04
	SP4	0.25±.03	1.00±.02	0.57±.15
	SP5	0.03±.12	0.98±.07	0.12±.21
	SP6	0.20±.23	0.95±.09	0.70±.19
	SP7	0.00±.00	1.00±.00	0.67±.27
	ST1	0.00±.00	1.00±.00	0.99±.03
	ST2	0.85±.27	1.00±.00	1.00±.00
	ST3	0.55±.30	0.99±.04	0.78±.20
	ST4	0.00±.00	0.87±.16	0.47±.36

(c)

Training		c3		
Test		c1↑	c2→	c3↗
↑Acc.		58.3±7.5 %	76.0±5.6 %	**99.2±2.0 %**
↑F_1-measures per class	EL1	0.76±.36	0.02±.09	1.00±.00
	EL2	0.23±.34	0.67±.30	0.96±.09
	EL3	0.02±.09	0.97±.09	1.00±.00
	SP1	0.83±.13	0.85±.15	0.98±.09
	SP2	0.71±.31	0.72±.34	0.99±.05
	SP3	0.77±.15	0.97±.08	1.00±.00
	SP4	0.45±.08	0.95±.09	1.00±.00
	SP5	0.83±.27	0.00±.00	0.99±.04
	SP6	0.58±.18	0.52±.19	0.99±.03
	SP7	0.04±.13	0.79±.17	1.00±.00
	ST1	0.06±.14	0.98±.08	1.00±.00
	ST2	0.95±.11	0.98±.06	1.00±.00
	ST3	0.91±.12	0.73±.17	1.00±.00
	ST4	0.24±.30	0.81±.21	0.96±.12

(d)

Training		c1 c2 c3		
Testing		c1↑	c2→	c3↗
↑Acc.		**97.8±2.6 %**	**95.4±3.4 %**	**98.7±2.7 %**
↑F_1-measures per class	EL1	1.00±.00	0.93±.12	0.99±.04
	EL2	0.92±.12	0.82±.23	0.93±.13
	EL3	0.97±.07	0.99±.05	1.00±.00
	SP1	0.97±.10	0.97±.11	0.96±.19
	SP2	0.98±.06	0.98±.06	0.99±.06
	SP3	1.00±.00	1.00±.00	1.00±.00
	SP4	1.00±.02	0.99±.03	1.00±.00
	SP5	0.98±.04	0.94±.13	0.99±.04
	SP6	0.97±.07	0.94±.08	0.99±.03
	SP7	0.99±.03	1.00±.00	1.00±.00
	ST1	0.99±.03	1.00±.00	1.00±.00
	ST2	1.00±.00	1.00±.00	1.00±.00
	ST3	0.99±.05	1.00±.00	1.00±.00
	ST4	0.93±.10	0.89±.12	0.94±.13

with a high precision, however, “Torso twist” (SP2) was often confused with SP1 which resulted in a low recall and, hence, in a lower F_1-measure. In both actions the subjects were asked to jump and raise their arms. The test performed on the diagonally-placed camera c3 achieved an accuracy of 56.4%. A large number of classes had a high recall, e.g. “Rope jump” (SP1), “Push up” (SP4), “Stationary lunge” (ST1), “Torso twist” (ST2) and “Shoulder stretch” (ST3). However, there were many confusions leading to a low precisions which in turn yielded lowered F_1-measures. The action SP2 which had a recognition rate of 75% constitutes as a representative example. An error of 23% was related to a confusion with action SP2.

Table 5.4b shows the results when performing the training on the data of the lateral camera c2. Testing on the data of the same camera c2 resulted in a slightly lower accuracy of 96.9% than the previous training configuration. The accuracies based on data generated for the camera c1 and camera c3 were 28.9% and 69.0%, respectively. The action “Hand wave” (EL2) had the lowest F_1-measure of 79% when testing on data derived from the camera c2. The corresponding recognition rate achieved only 80% which can be related to the confusion with “Toe touch” (ST4). In turn, the action ST4 had the second lowest F_1-measure of 87% because it was mainly confused with EL2 10%. This can be accounted to the fact that both actions are composed of a similar downward motion. The testing on data derived from camera c1 had the second lowest accuracy in this experiment which can be explained by the dramatic change from one camera perspective to the other. In this setting, many samples from different classes were attracted by the multifariousness movements of the class “Push up” (SP4), such as EL1, EL2, EL3, SP3, SP4, SP6, ST1 and ST4. The action SP4 contains: a turn to the right, the transition to the push up starting position, the downwards and upwards motions of the exercise, the uprise and, finally, a turn to the left. In the third setting in which testing was performed using data from camera c3 the highest F_1-measure of 100% was achieved by the action class “Torso twist” (ST2). This can again be attributed to the advantagous movements which can be observed in all three cameras.

Table 5.4c holds the results of the training configuration in which the training was performed on data derived from the diagonally-placed camera c3. This setting achieved the most robust accuracies so far. Using the data derived from the same camera for testing yielded in an accuracy of 99.2%. Testing on data derived from camera c1 and c2 achieved only accuracies of 58.3% and 76.0%, respectively. The testing on data obtained from camera c3 resulted in a symmetric confusion of 4% between the actions “Pick up” (EL2) and “Toe touch” (ST4). Testing on the data extracted from camera c1 shows that almost no samples were attracted to the classes “Pick up” (EL2), “Dumbbell frontal raise” (SP7) and “Rope jump” (SP1). Furthermore, a large set of classes were frequently misclassified, e.g. “Dumbbell frontal raise” (SP7) was confused with “Dumbbell shoulder press” (SP6) with about 89% and “Stationary lunge” (ST1) was confused with “Push up” (SP4) with about 87%. Using the data of camera c2 for testing resulted in a better performance than using the data of camera c1. In this setting, the actions “Hand wave” (EL1) and “Dumbbell lateral raise” (SP5) had low recognition rates of about 1%. EL1 was often confused with SP6 (40%) SP7 (21%) ST3 (32%), wheras SP5 was often confused with SP6 (49%) SP7 (29%) ST3 (16%).

The results of the last training configuration are presented in Table 5.4d. In this setting, the training was performed using the data derived from all three camera views. Evaluating the test data led to accuracies of 97.8% for the data of camera c1, 95.4% for camera c2 and camera c3 98.7%. Analogously to the accuracies, the F_1-measures of this training configuration were exceptionally good, although slightly lower performances were achieved compared with the first three training configurations which used data derived from only one camera perspective.

In summary one can say that a good action recognition can be achieved in case data from the perspective to be test is already part of the training dataset. The experiments show that view-invariance cannot be expected from the studied state-of-the-art approach as proposed by (Laptev et al., 2008). A reliable recognition was only possible in case the training data comprised the perspectives of the target test data.

5.2.2.2. HMM Graph Probability Densities Experiment

The HMM-GPD approach is examined in the second experiment. All HMM-GPD were trained using five hidden states and two GMM mixture components for each edge. The results of the four training configurations are provided in Table 5.5. The first three training configurations were based on data using only one camera perspective for training. The fourth training configuration used all three camera views for training. Testing was performed using all camera perspectives individually. A successful classification using the HMM-GPD requires that the training data covers all edges sufficiently such that good and representative statistics can be learned.

Table 5.6a shows the results obtained when training on the data generated using the viewpoint of the frontal camera c1. This perspective had almost no self-occlusions such that the HMM-GPD was able to learn a good representation of all actions. Hence, the approach achieved good results when performing the recognition on each of the three test sets. The perspectives of camera c1, c2 and c3 achieved accuracies of 98.0%, 87% and 95.7%, respectively. Testing the data of the camera c2 perspective constituted an exception by having a very low accuracy. Taking a closer look at the F_1-measures reveals that some actions were recognized with low rates, i.e. “Rope jump” (SP1), “Squat jump” (SP3), “Push up” (SP4) and “Toe touch” (ST4). Especially, the action SP4 stood out by an $F\text{-}_1$-measure of only 5%. This can be related to the unique characteristics of the action SP4 in which the subjects turns from camera c1 towards camera c2 in order to perform the push up exercise. As a result, some parts of the body are hidden during the sport exercise from the camera perspective c1 such that no proper statistics for these edges were learned. However, these parts became visible from the perspective of camera c2. The insufficient statistics learned for action SP4 led to a low model likelihood such that the samples of this action were attracted by samples of other classes, such as SP1 and SP3. For instance, 75.8% of the decisions which should be assigned to the class SP4 were recognized as the class SP1. The class ST4 was further frequently confused with the class SP3. The reason for this confusions can be related to similar extreme positions of the exercise and insufficient data for fitting the observations.

Table 5.6b shows the results obtained when training on data generated using the viewpoint of the lateral camera c2. The data of this perspective are associated with heavy self-occlusions for most of the actions. Generally, the far side of the body is permanently

Table 5.5. Accuracies and F_1-measures of HMM-GPD on the UUlm Multi-perspective Action Dataset. (a, b, c) Training performed using data derived from only one camera perspective and testing on data derived from all three cameras perspectives. (d) Training and testing performed on data derived from all three cameras perspectives. All values averaged with standard deviation.

(a)

	Training	c1		
	Testing	c1↑	c2→	c3↗
	↑Acc.	**98.0**±**2.4 %**	87.0±5.0 %	**95.7**±**4.1 %**
↑F_1-measures per class	EL1	1.00±.00	0.96±.13	1.00±.00
	EL2	0.97±.10	0.81±.31	0.80±.34
	EL3	0.98±.07	0.98±.06	0.98±.06
	SP1	0.93±.14	0.64±.15	0.93±.13
	SP2	0.97±.07	0.98±.07	0.98±.06
	SP3	0.96±.09	0.77±.19	0.85±.17
	SP4	0.96±.18	0.05±.15	0.96±.11
	SP5	0.98±.07	0.98±.07	0.98±.07
	SP6	1.00±.03	0.97±.08	0.99±.04
	SP7	0.99±.03	0.99±.03	1.00±.03
	ST1	0.99±.06	1.00±.00	1.00±.03
	ST2	0.99±.04	0.99±.04	0.99±.04
	ST3	1.00±.00	1.00±.00	1.00±.00
	ST4	0.97±.05	0.74±.39	0.84±.31

(b)

	Training	c2		
	Testing	c1↑	c2→	c3↗
	↑Acc.	11.3±2.9 %	**97.0**±**3.9 %**	31.8±5.5 %
↑F_1-measures per class	EL1	0.00±.00	0.96±.08	0.00±.00
	EL2	0.00±.00	0.96±.08	0.00±.00
	EL3	0.00±.00	0.95±.13	0.00±.00
	SP1	0.00±.00	0.96±.10	0.00±.00
	SP2	0.02±.09	0.97±.06	0.52±.14
	SP3	0.00±.00	0.95±.10	0.31±.21
	SP4	0.16±.00	0.92±.18	0.27±.05
	SP5	0.00±.00	0.97±.07	0.00±.00
	SP6	0.00±.00	0.95±.13	0.00±.00
	SP7	0.00±.00	0.97±.11	0.66±.32
	ST1	0.00±.00	0.99±.04	0.03±.10
	ST2	0.49±.45	0.99±.05	0.96±.11
	ST3	0.00±.00	0.98±.05	0.37±.36
	ST4	0.00±.00	0.99±.03	0.00±.00

(c)

	Training	c3		
	Testing	c1↑	c2→	c3↗
	↑Acc.	18.1±3.6 %	**98.3**±**2.5 %**	**98.2**±**2.5 %**
↑F_1-measures per class	EL1	0.09±.21	0.97±.10	1.00±.00
	EL2	0.00±.00	0.99±.04	0.96±.13
	EL3	0.08±.22	0.99±.04	0.99±.05
	SP1	0.00±.00	0.96±.18	0.96±.10
	SP2	0.02±.09	0.98±.07	0.99±.06
	SP3	0.00±.00	0.97±.07	0.96±.09
	SP4	0.17±.01	0.96±.18	0.95±.18
	SP5	0.00±.00	0.97±.07	0.98±.07
	SP6	0.00±.00	0.97±.06	0.99±.04
	SP7	0.00±.00	0.99±.04	0.99±.05
	ST1	0.00±.00	1.00±.00	1.00±.00
	ST2	0.98±.06	0.99±.04	0.99±.04
	ST3	0.50±.34	1.00±.00	1.00±.00
	ST4	0.01±.07	1.00±.00	0.98±.07

(d)

	Training	c1c2c3		
	Testing	c1↑	c2→	c3↗
	↑Acc.	**98.3**±**2.4 %**	**98.3**±**2.6 %**	**98.1**±**2.6 %**
↑F_1-measures per class	EL1	1.00±.00	0.97±.10	1.00±.00
	EL2	0.96±.08	0.98±.06	0.93±.14
	EL3	0.98±.05	0.99±.05	0.99±.04
	SP1	0.96±.10	0.96±.18	0.96±.11
	SP2	0.98±.07	0.98±.07	0.98±.07
	SP3	0.99±.04	0.98±.05	0.97±.07
	SP4	0.94±.19	0.96±.18	0.95±.18
	SP5	0.98±.07	0.98±.07	0.98±.07
	SP6	1.00±.02	0.97±.06	0.99±.04
	SP7	0.99±.04	0.99±.04	0.99±.04
	ST1	1.00±.00	1.00±.03	1.00±.00
	ST2	0.99±.04	0.99±.04	0.99±.04
	ST3	1.00±.00	1.00±.00	1.00±.00
	ST4	0.98±.07	1.00±.00	0.98±.07

hidden such that no statistics on related edges were learned. This is also reflected in the results: The recognition based on the same perspective achieved an accuracy of 97.0%, whereas the recognition based on the diagonally-placed camera c2 has a clearly lowered accuracy of 31.8% and the recognition based on the frontal camera c1 achieved only an accuracy of 11.3%. In both cases, the low accuracies can be related to the low number of observed edges. In this setting, the class “Push up” (SP4) attracted basically all samples of the other classes.

Table 5.6c shows the results when using the data generated using the diagonally-placed camera c3 for training. The testing using data rendered from the same camera perspective c3 and the lateral camera c2 yielded high accuracies of 98.2% and 98.3%, respectively. However, the recognition based on the frontal camera c1 achieved only an accuracy of 18.1%. The highest F_1-measures in this setting were achieved by the classes “Torso twist” (ST2), “Shoulder stretch” (ST3) and “Push up” (SP4). In all of these three actions, the subjects were asked to perform one or multiple turns towards the camera c2 such that the sequences representing these actions get a good coverage of all edges. This in turn leads to informative training and test data.

Table 5.6d shows the last training configuration in which the training was performed on the mixed dataset generated using all three camera perspectives. In this setting, the testing on data generated using the viewpoints of camera c1, camera c2 and camera c3 generally yielded good accuracies of 98.3%, 98.3% and 98.1%, respectively.

The experiment clearly shows that the recognition is less dependent on the viewpoint of the test data. However, a sufficient coverage of all edges for all actions in the training data is still mandatory. This issue is addressed in the next section by replacing the HMM with the CHMM. In a CHMM, hidden states which are associated with GPD can be shared over actions. Hence, edges with insufficient coverage can be learned with the help of related actions. The next experiment presents the results obtained using the CHMM-GPD.

5.2.2.3. CHMM Graph Probability Densities Experiment

The last experiment on the UUlmMAD combines the benefits of the GPD with the benefits of the CHMM. The sharing of hidden states in the CHMM-GPD allows to set up statistics on edges for action which are observed by other actions. All CHMM were trained using 50 hidden states and edges being modeled using two GMM mixture components. The results of the four training configurations are provided in Table 5.7.

Table 5.8a shows the accuracies and F_1-measures in case the training was done with data generated from the viewpoint of the frontal camera c1. Testing data of all three camera perspectives c1, c2 and c3 led to a view-invariant recognition with accuracies of 97.1%, 94.1% and 95.5%, respectively. The lowest F_1-measure of 93% was obtained regarding the class “Toe touch” (ST4) when using test data generated from the perspective of c1. The performance of the class ST4 can be related to the frequent confusion with the classes “Squat jump” (SP3) 6.2% and “Sit down” (EL3) 3.1%. This result is not surprising given that no orientation and location with respect to the world coordinate system is presented in the graph data. In case the data were derived from the viewpoint of camera c2, the class “Pick up” (EL2) has the lowest F_1-measure of 83%. The strongest confusion took place with the classes “Squat jump” (SP3) 9.4%, “Push up” (SP4) 6.2%, “Sit down” (EL3) 4.2% and “Toe touch” (ST4) 3.1%. Again the confused actions were characterized by similar movements. The reasons for this low performance are most probable the limited observability of the graphs due to the perspective and the coarse modeling of the exact movement execution times in the Markov model. The lowest F_1-measure of 80% regarding the data generated for the viewpoint of camera c3 was as well achieved by the class “Pick up” (EL2). The confusions were exactly the same as with the data generated for the viewpoint of camera c2.

Table 5.7. Accuracies and F$_1$-measures of CHMM-GPD on the UUlm Multi-perspective Action Dataset. (a, b, c) Training performed using data derived from only one camera perspective and testing on data derived from all three cameras perspectives. (d) Training and testing performed on data derived from all three cameras perspectives. All values averaged with standard deviation.

(a)

Training		c1		
Testing		c1↑	c2→	c3↗
↑Acc.		**97.1±3.3 %**	**94.1±5.8 %**	**95.5±4.5 %**
↑F$_1$-measures per class	EL1	1.00±0.00	0.86±0.27	1.00±0.00
	EL2	0.95±0.11	0.83±0.30	0.80±0.34
	EL3	0.95±0.10	0.89±0.19	0.94±0.10
	SP1	0.97±0.08	0.97±0.08	0.97±0.08
	SP2	1.00±0.03	0.99±0.04	1.00±0.03
	SP3	0.95±0.08	0.93±0.11	0.92±0.11
	SP4	0.93±0.12	0.89±0.19	0.91±0.19
	SP5	0.98±0.07	0.95±0.09	0.98±0.07
	SP6	0.99±0.04	0.96±0.06	0.99±0.04
	SP7	0.99±0.03	0.99±0.05	0.99±0.03
	ST1	0.99±0.06	0.98±0.07	0.98±0.08
	ST2	0.95±0.13	0.96±0.13	0.94±0.14
	ST3	1.00±0.00	1.00±0.00	1.00±0.00
	ST4	0.93±0.19	0.86±0.23	0.90±0.16

(b)

Training		c2		
Testing		c1↑	c2→	c3↗
↑Acc.		46.4±7.8 %	**96.5±4.1 %**	78.7±7.9 %
↑F$_1$-measures per class	EL1	0.00±0.00	0.92±0.14	0.13±0.28
	EL2	0.55±0.35	0.90±0.25	0.79±0.32
	EL3	0.74±0.31	0.97±0.07	0.87±0.21
	SP1	0.38±0.15	0.98±0.06	0.84±0.21
	SP2	0.51±0.34	0.96±0.18	0.95±0.18
	SP3	0.69±0.20	0.94±0.11	0.69±0.17
	SP4	0.44±0.15	0.93±0.19	0.81±0.22
	SP5	0.01±0.07	0.94±0.11	0.76±0.32
	SP6	0.05±0.18	0.97±0.07	0.55±0.38
	SP7	0.17±0.34	0.99±0.03	0.92±0.10
	ST1	0.65±0.39	0.98±0.07	0.87±0.24
	ST2	0.92±0.15	0.97±0.10	0.96±0.11
	ST3	0.44±0.41	0.99±0.04	0.75±0.25
	ST4	0.16±0.32	0.99±0.04	0.58±0.41

(c)

Training		c3		
Testing		c1↑	c2→	c3↗
↑Acc.		73.8±9.4 %	**95.9±4.7 %**	**96.8±3.6 %**
↑F$_1$-measures per class	EL1	0.50±0.41	0.82±0.32	1.00±0.00
	EL2	0.64±0.34	0.92±0.15	0.89±0.20
	EL3	0.93±0.13	0.93±0.20	0.95±0.11
	SP1	0.56±0.21	0.98±0.06	0.98±0.06
	SP2	0.80±0.28	0.99±0.04	1.00±0.03
	SP3	0.83±0.17	0.98±0.05	0.95±0.09
	SP4	0.67±0.18	0.90±0.19	0.91±0.14
	SP5	0.43±0.45	0.95±0.09	0.99±0.06
	SP6	0.56±0.43	0.96±0.09	0.99±0.04
	SP7	0.51±0.46	0.99±0.03	0.99±0.03
	ST1	0.92±0.15	0.98±0.07	0.98±0.07
	ST2	0.83±0.25	0.96±0.13	0.95±0.15
	ST3	0.95±0.11	1.00±0.00	1.00±0.00
	ST4	0.78±0.27	0.97±0.06	0.96±0.09

(d)

Training		c1c2c3		
Testing		c1↑	c2→	c3↗
↑Acc.		**97.2±3.7 %**	**97.3±3.7 %**	**96.9±3.7 %**
↑F$_1$-measures per class	EL1	0.99±0.04	0.97±0.10	0.96±0.18
	EL2	0.94±0.12	0.95±0.11	0.93±0.12
	EL3	0.98±0.06	0.95±0.19	0.98±0.05
	SP1	0.98±0.06	0.99±0.05	0.98±0.06
	SP2	0.99±0.04	1.00±0.03	0.99±0.04
	SP3	0.98±0.08	0.97±0.07	0.96±0.10
	SP4	0.91±0.19	0.92±0.19	0.93±0.19
	SP5	0.98±0.07	0.97±0.08	0.98±0.07
	SP6	0.99±0.03	0.99±0.04	0.98±0.05
	SP7	0.99±0.03	0.99±0.04	0.99±0.03
	ST1	0.97±0.09	0.97±0.08	0.97±0.09
	ST2	0.95±0.13	0.96±0.13	0.95±0.13
	ST3	0.99±0.04	1.00±0.00	0.99±0.06
	ST4	0.95±0.09	0.96±0.08	0.93±0.13

Table 5.8b shows the performance when training with data generated from the viewpoint of the lateral camera c2. The data of this perspective are strongly affected by self-occlusion since most actions were performed in the direction of camera c1. Hence, the achieved accuracies of 46.4%, 96.5% and 78.7% for the testing on data generated using the viewpoint of camera c1, c2 and c3 have only a mid-field performance. Regarding the test data rendered from the perspective of camera c1, samples of almost all classes were occasionally confused with either “Rope jump” (SP1) or “Push up” (SP4). This is most probably related to the fact that these actions have a high chance to provide sequences of complete graphs during training such that they were well represented in the learned CHMM-GPD. As excepted, testing on the data using the same perspective of camera c2 performed good. The class “Pick up” (EL2) had the lowest F$_1$-measure of 90% since it was occasionally confused with the class “Squat jump” (SP3) 9.4% and class “Push up” (SP4) 2.1%. In case of testing on

the data created from the viewpoint of camera c3, the classes “Squat jump” (SP3) and “Push up” (SP4) attracted a large number of samples from other classes. The class “Hand wave” (EL1) constituted an exception since it achieved only an F_1-measure of 13%. In this special case, most of the class samples were assigned to the action “Shoulder stretch” (ST3) 53.1% and “Dumbbell frontal raise” (SP7) 19.8%. The low performance is most likely related to the extremely low number of visible edges. In the worst cases regarding the action “Hand wave” (EL1), only the hand and head nodes of the upper skeleton model can be observed from the viewpoint of camera c2.

Table 5.8c shows the results when training with data using the viewpoint of the diagonally-placed camera c3. The testing on data generated using the perspectives of camera c1, c2 and c3 achieved accuracies of 73.8% 95.9% and 96.8%, respectively. The lowest accuracy was obtained when testing on data created using the perspective of camera c1. In this setting, a large number of samples was attracted by the classes “Rope jump” (SP1) and “Push up” (SP4). The lowest F_1-measure of 43% was achieved by the class “Dumbbell lateral raise” (SP5). Here, a major share of 51.5% of the samples were attracted by the class SP1. Testing on data generated from the perspectives of camera c2 and c3 achieved very good recognition rates. The class “Hand wave” (EL1) has the lowest F_1-measure of 82% using the data created for the viewpoint of camera c2. The samples of the class were confused with “Dumbbell lateral raise” (SP5) 11.5% and “Dumbbell shoulder press” (SP6) 9.6%. Regarding the data generated for the viewpoint of camera c3 the lowest F_1 measure of 89% was achieved by class “Pick up” (EL2) because of a confusion with the classes “Push up” SP4 7.3% and “Squat jump” (SP3) 6.2%.

Table 5.8d shows the performance of the last configuration in which the training was performed using samples generated from all three perspectives. The testing on data created for the individual camera perspectives c1, c2 and c3 achieved accuracies of 97.2%, 97.2% and 96.9%, respectively. The action “Push up” (SP4) has the lowest F_1-measure of 91% among all actions when testing on data generated using the viewpoint of camera c1. The class attracted a large number of samples from other classes such as “Stationary lunge” (ST1) 5.5% and “Torso twist” (ST2) and “Pick up” (EL2), both with 4.2%. Given that the data were rendered from the perspective of camera c2, the action “Pick up” (EL2) had the lowest F_1-measure of 95%. In this setting, the samples of class EL2 were mainly confused with “Squat jump” (SP3) 3.1% and “Push up” (SP4) 2.1%. Testing with data derived from the viewpoint of camera c3 led to low F_1-measures of 93% for the class “Pick up” (EL2), “Push up” (SP4) and “Toe touch” (ST4). These F_1-measures were largely related to the classes “Squat jump” (SP3) and “Push up” (SP4) which attracted samples of other classes. Furthermore, EL2 and ST4 were occasionally confused in both directions.

The CHMM-GPD showed the best view-invariance performance of all three examined

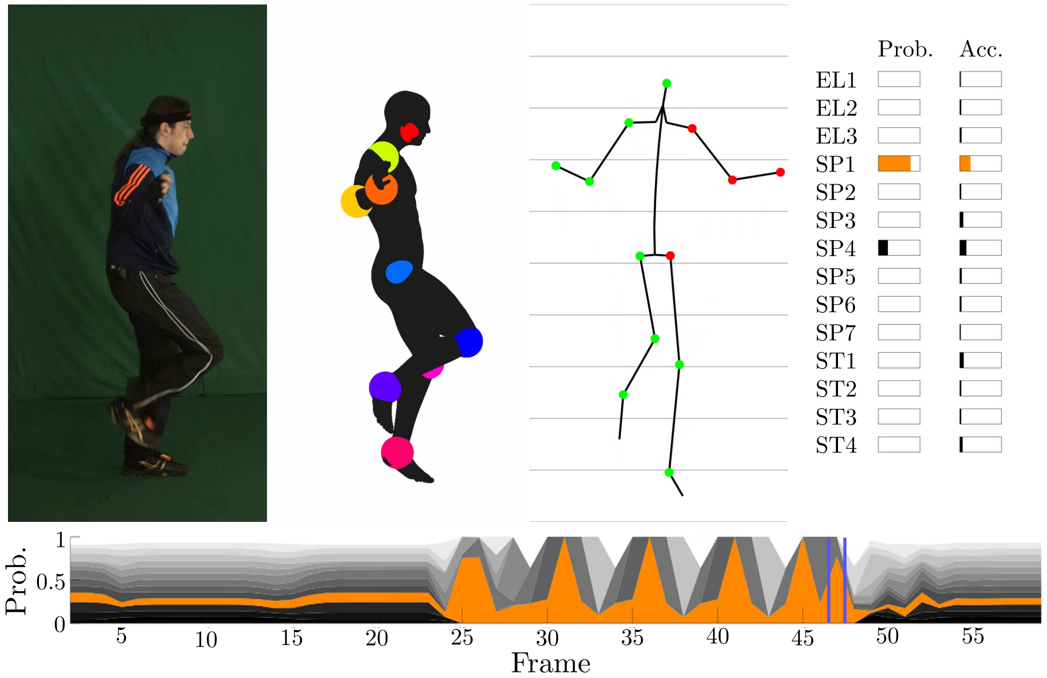


Figure 5.4. Probability distribution over time derived performing exact inference on the CHMM-GPD. The figure shows the process of data extraction and the output probability distribution. The upper part of the figure shows from left to right: the frame of the camera c2, the rendered human avatar to derive the self-occlusion information, the skeleton model in which the red markers indicate the joints to be removed and the probability distribution of the current frame and the accumulated probability distribution. The lower plot shows the class probability over time, whereas the orange area denotes the actual action class. For a detailed description please refer to the text.

approaches. Recognitions using the CHMM-GPD can be performed from any perspective once a sufficient statistics was trained. This was for instance the case in the first training configuration.

Figure 5.4 depicts the inferred probability distribution of the random variables of the causal sequence using a uniform prior. The picture on the left-hand side shows a frame of the camera c2 depicting the subject in the middle of the action “Rope jump” (SP1). The second picture from the left shows the rendered human avatar generated using the motion capturing data which was used to derive the self-occlusion information. The second picture from the right shows the skeleton model itself. The green markers show the joints visible, whereas the red markers represent the joint which cannot be observed due to self-occlusion. The occluded joints were removed from the sequence of skeleton models past to the CHMM-GPD. The picture on the right-hand side shows the probability distribution of the current frame and the accumulated probability distributions so far. The currently highest probability is assigned to the action “Rope jump” (SP1), while the second highest probability is given to the action “Push up” (SP4). This can be explained by the fact that the orientation of skeleton model was not considered. The current pose has a similarity with the pose of a subject facing the floor and performing the push up exercise. The accumulated probability distribution shows that the action “Rope jump” (SP1) has gathered the highest amount of probability so far. The distribution over time is depicted in the plot at the bottom of the figure. The beginning of the sequence, contains a long phase in which the subject remains in the neutral position. The orange area represents the probability assigned to the actual action class. Whenever the subject’s rope is in the back and gains momentum for the next swing, the probability over time forms a peak. Obviously having the arms pushed back as far as possible is a good indicator for the actual action.

5.2.3. Summary

The study to examine the HMM-GPD and CHMM-GPD is composed of two parts. The first part using the OAD shows that the HMM-GPD approach can handle datasets of incomplete features during training and testing. The second part which was performed using the UUlmMAD shows that the graph sequences can be composed of graphs with variable appearances. It examines a state-of-the-art architecture (Laptev et al., 2008), the HMM-GPD and the CHMM-GPD which uses a common representation across classes.

In the first part, the HMM-GPD was used to train and test on sets created from different occlusion configurations of the OAD. The results show that despite incomplete feature sets, a training and classification can still be performed (Table 5.2). In both cases, the performances decreased only marginally with the degree of occlusions. Within the OAD, each sequence was composed of graphs with varying weights, but with the same structure.

The second part uses the UUlmMAD which features sequences of graphs having variable size. Results of three experiments were provided: (1) a state-of-the-art image-based action recognition architecture, (2) HMM-GPD and (3) CHMM-GPD. In all experiments, four training configurations were analyzed: (1,2,3) training on data generated using only one of the three perspectives and testing on data of each single perspective and (4) training on mixed data generated using all perspectives and testing on data of each single perspective.

The first experiment shows that the image-based action recognition architecture is sensible for changes in the perspective. In case the training was performed using data from only one perspective, the recognition rates severely dropped when testing on data recorded from other perspectives. Only in the case training was performed on data of all three camera perspectives, good recognition rates was achieved for the very same perspectives (Table 5.3). Hence, actions have to be recorded for all perspectives which are intended to be recognized.

The second experiment examines the HMM-GPD. The major requirement for a successful application of the HMM-GPD is to obtain sufficient statistics on all edges of the graphs. As pointed out in Section 4.1.2.2, there are specific perspectives under which the data cover certain actions insufficiently. The results show that if a representation based on one perspective was learned, it is likely that the HMM-GPD approach is able to recognize the actions given data generated using the same perspective (Table 5.5). Complete view-invariance can be achieved in case the training data sufficiently cover all edges. This is already the first step towards view-invariance.

In the third experiment, the HMM-GPD was replaced by the CHMM-GPD in order to find a common representation for all actions. The CHMM-GPD shares the hidden states for all actions and, hence, clusters similar body poses. Actions which cover edges insufficiently can benefit from the distributions of edges of other related actions. The results show that it is clearly more likely to learn a sufficient representation compared with the HMM-GPD. The CHMM-GPD achieved higher recognition rates which confirm an improved view-invariance (Table 5.7). In three cases low accuracies of 46.4%, 73.8% and 78.7% were achieved for the 14-class classification problem. These cases can be explained by the insufficient coverage of edges in the training data. In all other nine cases, the classifier accuracies were over 94%. Hence, the CHMM-GPD outperforms the HMM-GPD and the state-of-the-art architecture.

5.3. CHMM MCS for Laughter Spotting

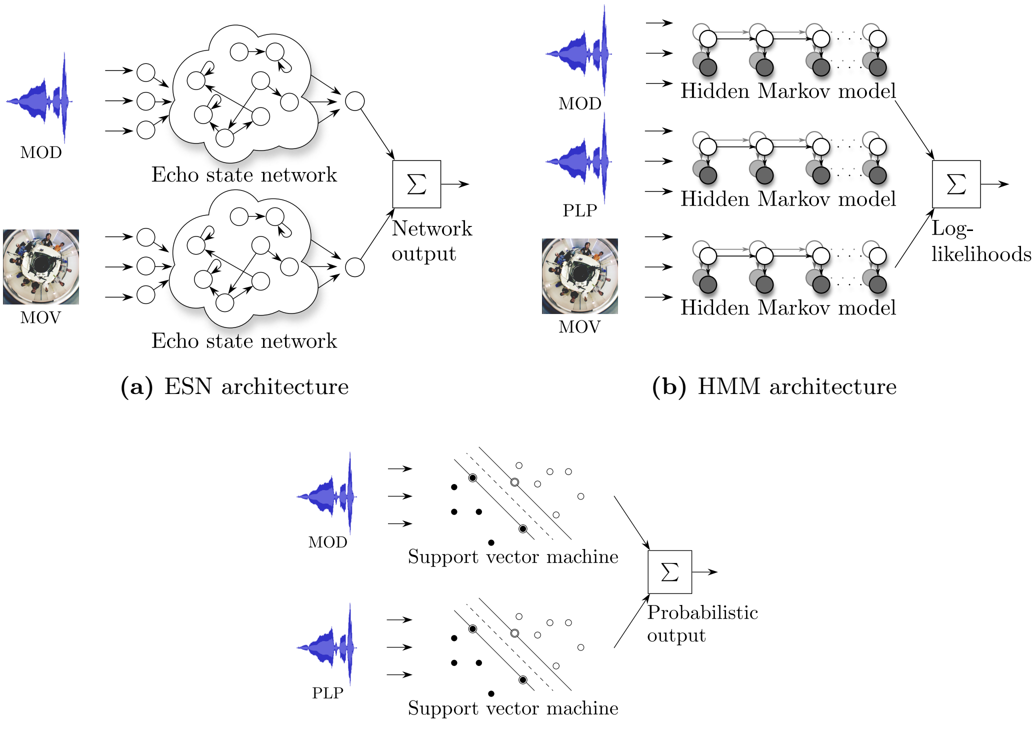
The next study examines three temporal classifier architectures operating on unsegmented data and the HMM-MCS and CHMM-MCS in particular. It is composed of two experiments which base on the Freetalk Dataset. The first baseline experiment compares the three MCS based on the different classifiers (Scherer et al., 2012b), whereas the second experiment examines the HMM-MCS and the CHMM-MCS (Glodek et al., 2011c).

5.3.1. Baseline Experiment

The experiment examines three MCS using different base classifiers, namely (1) the neuronal-motivated echo state network (ESN) (2) the probabilistic HMM and (3) the L2 soft-margin SVM from the statistical learning theory. All MCS are depicted in the Figure 5.5. In the following, short outlines of these classifier architectures are given.

5.3.1.1. ESN architecture

The first classifier architecture makes use of the ESN and is depicted in Figure 5.5a. An ESN is a recurrent neuronal network which can process sequential patterns (Jaeger, 2002). The key idea is to generate a sparse random matrix $\mathbf{W}$ which is called the reservoir and a random input neuron matrix $\mathbf{W}_{\text{in}}$. The input and output of the network are given by $\mathbf{x}_t \in \mathbb{R}^M$ and $\mathbf{y}_t \in [0,1]^I$ where t denotes the time step, M denotes the feature dimension and I denotes the number of classes to be predicted. The state of the neural network $\mathbf{z_{t+1}}$ is given by $f(\mathbf{W}^{\text{in}}\mathbf{x}_{t+1}+\mathbf{W}\mathbf{z}_t)$ where f is a sigmoid function. The network output is derived by



(a) ESN architecture

(b) HMM architecture

(c) SVM architecture

Figure 5.5. MCS architectures studied using the Freetalk Dataset.

$\mathbf{y}_{t+1} = f^{\text{out}}(\mathbf{W}^{\text{out}}\mathbf{z}_{t+1})$ where f^{out} is as well a sigmoid function. Mapping from the network state to the target label is performed by the matrix $\mathbf{W}^{\text{out}}$. It is trained by collecting the reservoir states in a matrix $\mathbf{Z}$ and using the pseudo-inverse $\mathbf{Z}^{+}$ to obtain the weighting matrix $\mathbf{W}^{\text{out}} = (\mathbf{Z}^{+}\hat{\mathbf{Y}})^{\mathrm{T}}$ where $\hat{\mathbf{Y}}$ is the matrix of inverted sigmoid teacher signals. The level of reservoir activation is controlled using the spectral radius of the matrix $\mathbf{W}$. The network requires a number of iteration to adjust itself to the input data. Therefore, the input sequences must have a sufficient length in order to perform a successful training. The training on Freetalk Dataset is done by concatenating the pre-segmented data samples. The operations to perform the matrix calculations and the collecting of the intermediate reservoir states led to high requirements to the computer's working memory such that only the MOV and MOD features were able to be processed. The multimodal fusion was achieved by buffering the audio network output and then downsampling it to the rate of the video network output. The final outputs were obtained by using a weighted average. The audio and video modalities were weighted with 0.7 and 0.3, respectively.

5.3.1.2. HMM architecture

The second architecture uses the HMM as the base classifier and is depicted in Figure 5.5b. All HMM utilized left-to-right transition matrices and eight hidden states. The

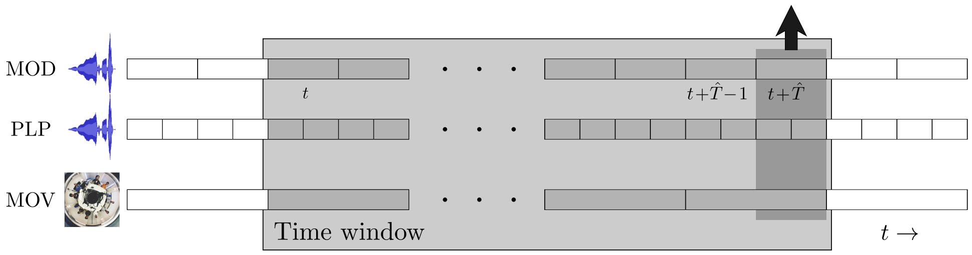


Figure 5.6. Time window used by the HMM architecture to handle of different sample rates. The utilized time window has a fixed temporal expansion based on the average length of the utterances and laughters in the pre-segmented dataset. The combined output of the HMM is considered to be aligned at the end of the time window.

MOD, PLP and MOV feature sets were utilized and modeled using GMM having eight mixture components with full covariance matrices. The length of the window which was shifted over the sequential data was based on the average duration of the utterances and laughters in the pre-segmented dataset. The outputs of all three models were combined by summing up the log-likelihoods of the same class. The combination was performed by multiplying the model probabilities. The different sample rates of the feature sets were addressed by normalizing the likelihoods using the length of the individual sequences. Decisions of the HMM were assumed to be observed at the end of the utilized window, as shown in Figure 5.6.

5.3.1.3. SVM architecture

The SVM architecture which is shown in Figure 5.5c makes use of an additional preprocessing step. A universal background model (UBM) was created to obtain a fixed-length feature vector providing an alternative view on the data. The UBM was derived using the Gaussians being part of a GMM (Scherer et al., 2012b) to form the so-called super-vector. The generation and adaption process of the UBM GMM is illustrated in Figure 5.7. The lower left-hand side of the figure shows the generation of the UBM in which a GMM is fitted to features extracted from training data. The generation of the UBM discards the temporal order and, hence, treats all samples independently. Test data was provided to the algorithm in the form of sequences created utilizing a window which was shifted over the recording. The features extracted from this sequence were again treated independently and passed to the EM algorithm to perform a fixed number of iterations. The means of the adapted GMM were then concatenated to obtain the output super vector. Due to the fixed number of EM adaptations, the means of the GMM representing the super vector moved towards the distribution of the given test features. Hence, the super vector encodes the difference of the input sequence to the UBM. The application of the UBM GMM leads to a representation which has a decreased dimensionality when compared to the original complete sequence of extracted features. The UBM representing the eight-dimensional MOD features used 20 mixture components and, hence, encoded a 160-dimensional super vector. The UBM which

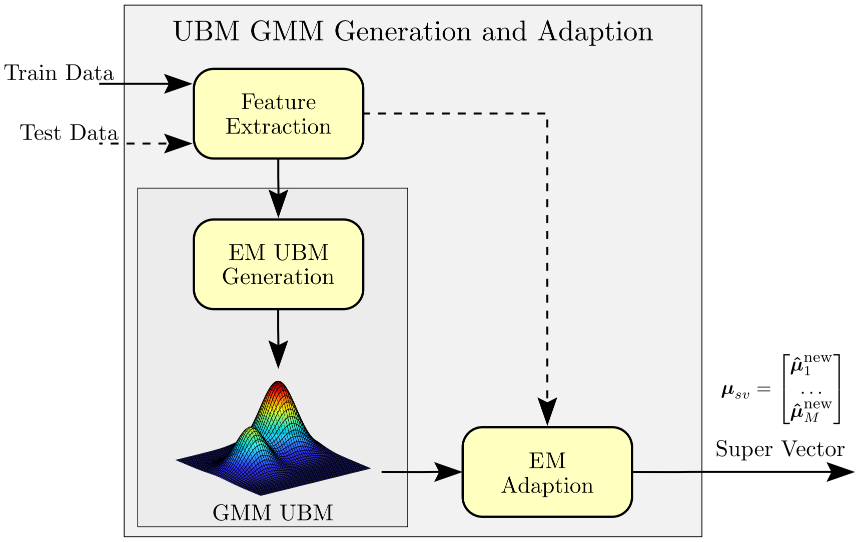

UBM GMM Generation and Adaption
Train Data
Test Data
Feature Extraction
EM UBM Generation
GMM UBM
EM Adaption
$\boldsymbol{\mu}_{sv} = \begin{bmatrix} \hat{\boldsymbol{\mu}}_1^{\text{new}} \\ \dots \\ \hat{\boldsymbol{\mu}}_M^{\text{new}} \end{bmatrix}$
Super Vector

Figure 5.7. Generation and adaption of the GMM UBM in the SVM architecture applied to the Freetalk Dataset. In the first step, the classical feature extraction is performed. The left-hand side of the figure shows the generation of the GMM UBM based on the features extracted from the training data. The super vector is obtained by performing a limited number of EM adaption steps on the extracted features of the test data and then concatenating the mean vectors of the GMM.

was trained to represent the 21-dimensional PLP features was composed of eight mixture components which in turn led to a 168-dimensional super vector. No UBM was created for the MOV features because no suitable GMM parameter configuration could be found. The fixed-length and time-independent super vector was used to train an L2 soft-margin SVM having a Gaussian radial basis function kernel and probabilistic outputs (Campbell et al., 2006b; Platt, 2000). The combination of the decisions from different channels was achieved by taking the average of the probabilistic outputs.

5.3.1.4. Results

Table 5.9 shows the result of the empirical evaluation of the three architectures in the setting of pre-segmented and unsegmented pattern recognition. The pre-segmented dataset was evaluated using 10 × 10 cross-validation. The unsegmented dataset was analyzed using the independent test recording and was evaluated on frame-level. The table provides classification performances on the pre-segmented and unsegmented datasets. The first two columns specify the utilized classifier, i.e. ESN, HMM or SVM, and the utilized feature sets, i.e. MOD, PLP and MOV. Additional columns provide the F_1-measure of the class laughter, as well as the precision and the recall. Quantities which could not be evaluated are denoted as being not available “n/a”.

The first part of the experiment is shown on the left-hand side of the table and provides the results in the pre-segmented setting. The network could not be initialized properly,

Table 5.9. Error rates, F_1-measures, precisions and recalls for different temporal MCS on the pre-segmented and unsegmented datasets of the Freetalk Dataset. The first column assigns the utilized classifier architecture, whereas the second column denotes which feature set was used. The third column shows the error rates obtained on the pre-segmented dataset. The remaining columns cover the results derived on the unsegmented dataset. The columns show the error rates, F_1-measures, precisions and recalls. All values averaged with standard deviation. Arrows indicate how to rate the measures.

Pre-segmented			**Unsegmented**			
Architecture	**Feature set**	**↓Error**	**↓Error**	**↑F_1**	**↑Precision**	**↑Recall**
ESN	*(MOD)*	n/a	13.4%	62±16 %	50±16 %	81±15 %
ESN	*(MOV)*	n/a	17.8%	46±06 %	38±10 %	58±19 %
ESN	*(MOD,MOV)*	n/a	9.1%	63±10 %	52±12 %	81±15 %
HMM	*(MOD)*	9.4%	6.7%	68±17 %	63±18 %	75±18 %
HMM	*(MOV)*	30.2%	26.8%	56±11 %	42±10 %	**85±10** %
HMM	*(PLP)*	10.1%	6.8%	70±18 %	66±21 %	75±19 %
HMM	*(MOD,MOV)*	8.2%	7.4%	62±15 %	59±21 %	65±15 %
HMM	*(PLP,MOV)*	7.3%	7.3%	69±17 %	67±21 %	71±19 %
HMM	*(MOD,PLP)*	6.5%	**6.5**%	**72±18** %	64±20 %	80±18 %
HMM	*(MOD,MOV,PLP)*	6.3%	7.1%	69±16 %	70±18 %	68±18 %
SVM	*(MOD)*	4.1%	7.6%	48±16 %	93±24 %	32±11 %
SVM	*(PLP)*	7.0%	8.1%	44±12 %	53±15 %	38±11 %
SVM	*(MOD,PLP)*	**3.7**%	7.2%	38±15 %	**95±19** %	24±10 %

because of the shortness of the pre-segmented test sequences. The lengths of the sequences were insufficient to carry out the adaptation phases of the ESN. Hence, no ESN architecture results were provided in the pre-segmented setting. The HMM architecture was evaluated for all feature combinations. In general, the unimodal HMM architectures achieved the highest error rates of 30.2%, 10.1% and 9.4% for the MOV, PLP and MOD feature sets, respectively. The combinations of two feature sets achieved notably lower error rates of 8.2%, 7.3% and 6.5% for the (MOD, MOV), (PLP, MOV) and (MOD, PLP) feature sets, respectively. Combinations with the MOV feature yielded consistently higher error rates than the combination of the two best unimodal features MOD and PLP. The HMM architecture which used all three feature sets has the lowest error rate of 6.3% so far. Only three SVM architecture were empirically evaluated since only the MOD and PLP feature sets were available. Analogously to the HMM architectures, the multimodal feature set combination outperformed the unimodal architectures. The unimodal SVM architectures achieved error rates of 7.0% and 4.1% for the PLP and MOD feature sets, respectively. The multimodal SVM architecture yielded an error rate of 3.7% and, hence, was the architecture with the lowest error rate using the pre-segmented dataset.

The right-hand side of the table shows the results of all classifier architectures in the setting of the unsegmented dataset. Regarding the ESN architecture, the multimodal setting

(MOD, MOV) rendered the lowest error rate of 9.1%. The ESN architecture using only the MOD feature achieved an error rate of 13.4%, whereas the architecture using only the MOV feature set had the highest error rate of 17.8% of all ESN architectures. Again, the HMM architecture was evaluated for all available features sets. So far, a large number of modalities was a good indicator for low error rates. However, the results on the unsegmented dataset were more complex. The HMM architectures making use of the MOV feature set achieved high error rates of 26.8% (MOV), 7.4% (MOD, MOV), 7.3% (PLP, MOV) and 7.1% (MOD, MOV, PLP). The remaining unimodal and multimodal HMM architectures performed clearly better and achieved error rates of 6.8% (PLP), 6.7% (MOD) and 6.5% (MOD, PLP). These error rates were among the lowest for the unsegmented dataset. The SVM architecture on the unsegmented dataset had a lower performance than on the pre-segmented dataset. The unimodal SVM architectures yielded error rates of 8.1% (PLP) and 7.6% (MOD). These error rates were higher than the error rate 7.2% of the multimodal SVM architecture (MOD, PLP). The highest precision was achieved by the multimodal SVM architecture with 95%. However, the approach had at the same time the lowest recall of 24% and, hence, had the lowest F_1-measure of 38%. It can generally be noted that the SVM architectures achieved low F_1-measures. The architecture with the highest performance on the unsegmented dataset was the HMM architecture using the MOD and PLP feature sets. It achieved the highest F_1-measure of 72% with a balanced precision and recall. The highest recall of 85% was achieved by the HMM architecture with the overall highest error rate using only the MOV feature. A detailed description of all methods and training procedures can be found in the joint work of Scherer et al. (2012b).

5.3.2. CHMM-MCS and HMM-MCS experiment

The second experiment on the Freetalk Dataset compares the CHMM-MCS and HMM-MCS architectures which were presented in Section 3.2.3 (Glodek et al., 2011c). The findings of the previous experiment are used as a baseline. In general, model parameters were chosen with regard to comparability between the HMM-MCS and CHMM-MCS. Therefore, all HMM and CHMM were trained using GMM with Gaussians having full co-variance matrices and a prior based on the number of the class occurrences was utilized.

For each modality, one HMM was trained using eight states with full transition matrix. In a first step, the ratio of each class per feature set was evaluated to perform the multimodal late fusion. The ratios were then multiplied in order to obtain a combined overall decision as described in Section 3.2.3. The calculations were performed in log-space. The error rates and the corresponding standard deviations are shown in Table 5.11a and were derived

Table 5.10. Error rates and $\mathbf{F}_1$-measures of the HMM-MCS, the restricted CHMM-MCS and the unrestricted CHMM-MCS on the Freetalk Dataset. Statistically significantly higher results with respect to the HMM-MCS with the lowest error rate are marked by $\star$. All values averaged with standard deviation. Arrows indicate how to rate the measures.

(a) HMM-MCS. Models set to 8 hidden states, full transition matrix and 6 mixture components.

Feature set	↓**Error rate**	↑$\mathbf{F}_1$(Laughter)	↑$\mathbf{F}_1$(Utterance)
MOD	8.44±1.88 %	55.8±11 %	95.3±1 %
PLP	8.87±1.99 %	58.5±9 %	95.0±1 %
MOV	29.26±5.78 %	23.4±5 %	81.8±4 %
MOD*MOV	8.95±1.82 %	53.0±11 %	95.0±1 %
MOD*PLP	7.45±1.83 %	61.5±10 %	95.9±1 %
PLP*MOV	8.83±2.27 %	58.3±10 %	95.1±1 %
MOD*PLP*MOV	**7.29±1.84 %**	**62.1±10 %**	**96.0±1 %**

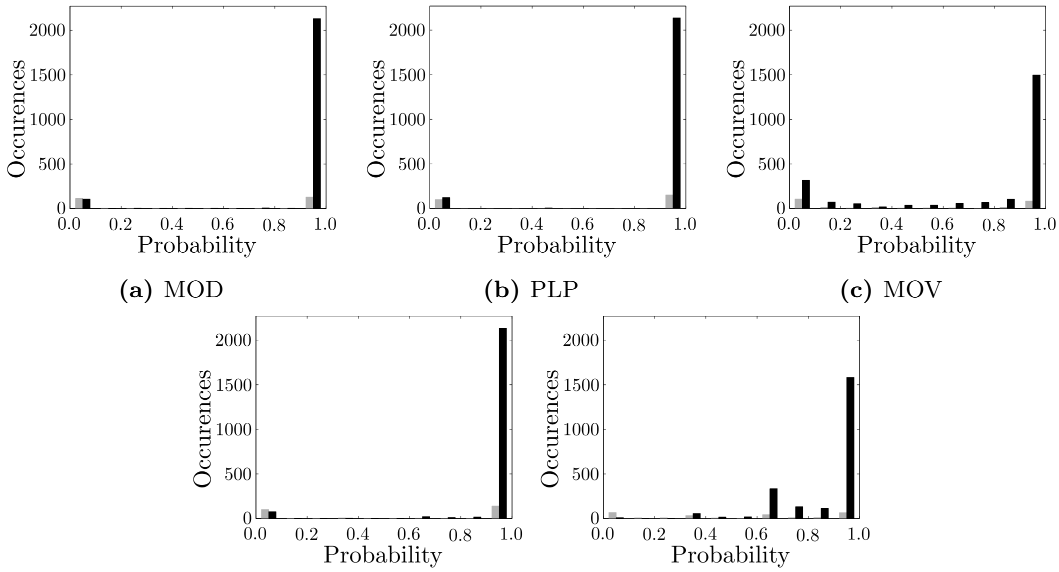
(b) Restricted CHMM-MCS. Models set to 16 hidden states, restricted transition and 6 mixture components.

Feature set	↓**Error rate**	↑$\mathbf{F}_1$(Laughter)	↑$\mathbf{F}_1$(Utterance)
MOD	9.62 ±1.08 %	51.9±7 %	94.7±1 %
PLP	8.94 ±1.46 %	57.5±8 %	95.0±1 %
MOV	25.43±5.11 %	26.5±5 %	84.5±4 %
MOD*MOV	9.10±1.20 %	53.3±7 %	95.0±1 %
MOD*PLP	7.44±1.73 %	61.8±10 %	95.9±1 %
PLP*MOV	8.87±1.62 %	57.3±9 %	95.1±1 %
MOD*PLP*MOV	7.37±1.80 %	61.9±10 %	95.9±1 %
MOD+MOV	9.05±1.24 %	53.6±7 %	95.0±1 %
MOD+PLP	**5.77±1.09 %***	**67.7±8 %***	**96.8±1 %***
PLP+MOV	8.76±1.60 %	57.7±9 %	95.1±1 %
MOD+PLP+MOV	8.24±1.52 %	56.5±8 %	95.4±1 %

(c) unrestricted CHMM-MCS.Models set to 16 hidden states, unrestricted transition matrix and 6 mixture components.

Feature set	↓**Error rate**	↑$\mathbf{F}_1$(Laughter)	↑$\mathbf{F}_1$(Utterance)
MOD	10.10±1.84 %	52.4±7 %	94.3±1 %
PLP	8.67±1.21 %	38.8±14 %	95.3±1 %
MOV	21.94±5.12 %	24.9±7 %	87.0±4 %
MOD*MOV	9.98±1.47 %	51.1±7 %	94.4±1 %
MOD*PLP	5.70±1.04 %**	**62.6±9 %**	96.9±1 %**
PLP*MOV	8.43±0.80 %	39.9±10 %	95.5±0 %
MOD*PLP*MOV	5.74±0.99 %**	62.4±9 %	96.9±1 %**
MOD+MOV	9.96±1.53 %	51.2±7 %	94.4±1 %
MOD+PLP	**5.62±1.07 %****	62.5±9 %	**97.0±1 %****
PLP+MOV	8.30±0.70 %	40.3±10 %	95.5±0 %
MOD+PLP+MOV	7.84±1.36 %	51.3±9 %	95.7±1 %

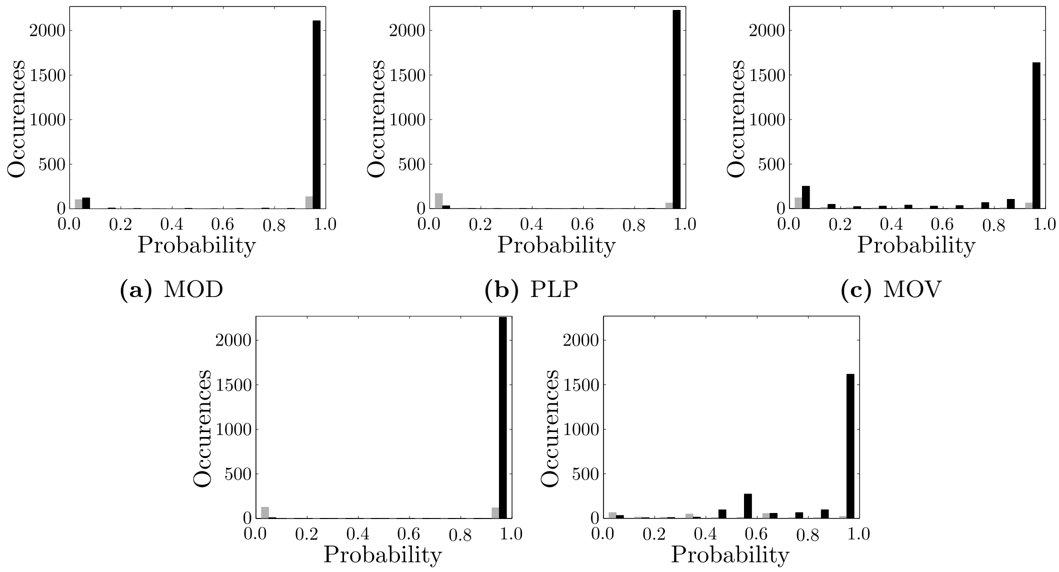
using 10 × 10 cross-validation. The unimodal classifier performances based on a single feature set are provided in the first three rows, and achieved error rates of 8.44%, 8.87% and 29.26% for the MOD, PLP and MOV feature sets, respectively. The fusion results based on combinations of features sets can be found in the consecutive rows. The lowest error rate of 7.3% was achieved by combining all features set, i.e. MOD, PLP, MOV. The corresponding F_1-measure of 96.0% for the class “Utterance” turned out to be very high, while the F_1-measure of the class “Laughter” achieved only 62.1%. Hence, the class “Utterance” was



(a) MOD (b) PLP (c) MOV

(d) MOD*PLP*MOV (e) MOD+PLP+MOV

Figure 5.8. Histogram of the class probability distributions provided by restricted CHMM on the Freetalk Dataset. Probabilities of the class “Utterance” given by black bars, whereas the probabilities of the class “Laughter” given by gray bars. Model set to 16 hidden states, restricted transition matrix and 6 mixture components.



(a) MOD (b) PLP (c) MOV

(d) MOD*PLP*MOV (e) MOD+PLP+MOV

Figure 5.9. Histogram of the class probability distributions provided by the unrestricted CHMM on the Freetalk Dataset. Probabilities of the class “Utterance” given by black bars, whereas the probabilities of the class “Laughter” given by gray bars. Model set to 16 hidden states, unrestricted transition matrix and 6 mixture components.

reliably detected but the class “Laughter” was frequently confused with the opposing class. The HMM-MCS using only the MOD and PLP feature sets achieved a similar error rate of 7.45%. The HMM-MCS using the feature sets (MOD*MOV) and (PLP*MOV) achieved error rates of 8.95% and 9.83%, respectively.

In the next step, the restricted CHMM-MCS was evaluated. The CHMM utilized in the MCS was restricted to have the same expressive capabilities as the HMM in the previous MCS. The restricted CHMM was configured with 16 hidden states and a transition matrix which was set to zero except for the elements that were initialized by the classic HMM. One CHMM was trained for each modality using the repeated label as the causal sequence. The distribution of the class random variable y allows the application of two different fusion strategies, i.e. multiplication and averaging which are denoted by * and $+$, respectively. The error rates and standard deviations of the CHMM-MCS are shown in Table 5.11b. For brevity, only the most relevant results will be pointed out in the following. While the unimodal MCS based on the MOD and PLP features achieved slightly higher error rates of 9.62% and 8.94%, respectively, the MCS based on the MOV feature showed a clear improvement of around 4%, i.e. an error rate of 25.43%. The *-fusion strategy had a performance comparable to the fusion using the HMM-MCS, although the multiplication in both approaches were fundamentally different. The results ranged between 7.44% (MOD*PLP) to 9.10% (MOD*MOV). The $+$-fusion strategy achieved as well only mid-range error rates, however, with one exception. The CHMM-MCS combining MOD$+$PLP outperformed clearly all previous results and achieved an error rate of 5.8% with an F_1-measure of 96.8% for the class “Utterance” and 67.7% for the class “Laughter”. The error rate and the two F_1-measures were statistically significantly better with respect to the paired t-test than the reference HMM-MCS (MOD*PLP*MOV) with the lowest error rate of the previous experiment. The p-values below 5% were marked by the symbol $\star$ and below 1% are marked by the symbol $\star\star$.

Figure 5.8 shows the distributions of the restricted CHMM-MCS outputs per class in a histogram. Each histogram covers a range of $[0, 1]$ with ten bins. The black bars represent the class “Utterance” having a large number of samples, whereas the gray bars represent the class “Laughter”. Figure 5.8a, Figure 5.8b and Figure 5.8c show the output distributions of the unimodal restricted CHMM. The decisions of the unimodal restricted CHMM using the audio features MOD and PLP had similar class distributions. Almost all decisions had a high degree of certainty, i.e. they ranged either close to zero or close to one. The restricted CHMM using the MOV feature set had a broader distribution which expresses the uncertainty associated with the video modality. Figure 5.8d and 5.8e illustrate the combined distributions using multiplication and averaging. While the *-fusion strategy enforced the decidedness of a classes recognition, the $+$-fusion strategy regarded the diversity of the classifiers and, therefore, was more easily affected by false decisions due to the MOV features.

The last part of the experiment analyzes the unrestricted CHMM-MCS using CHMM with 16 hidden states and fully-populated transition matrix. The results are shown in

Table 5.11c. Again only the most relevant results will be pointed out in the following. Applying the +-fusion strategy output of the unrestricted CHMM based on the MOD and PLP feature sets resulted in a low error rate of 5.7% with an F$_1$-measure of 97.0% for the class “Utterance” and 62.5% for the class “Laughter”. Furthermore, the *-fusion strategy of MOD and PLP, as well as MOD, MOV and PLP led to error rates of around 5.7%. The highest F$_1$-measure of 62.6% regarding the class “Laughter” was achieved by unrestricted CHMM operating the features MOD and PLP. All three fusion strategies had an error rate and F$_1$-measure for the class “Utterance” which were statistically significantly better than the best performing HMM-MCS (MOD*PLP*MOV).

The distributions of the class random variable of the unrestricted CHMM-MCS are presented in Figure 5.9. The histograms of the unimodal unrestricted CHMM are shown in Figure 5.9a, Figure 5.9b and Figure 5.9c. While the decisions obtained using the MOD or PLP feature sets were always certain (also in case of confusions), the CHMM based on the MOV feature set was considerably less certain about the class decision. Figure 5.9d shows the *-fusion strategy based on the MOD, PLP and MOV feature sets which achieved the second lowest error rate. The combination had reduced the error rate while keeping the decisions certain. The CHMM-MCS using the +-fusion strategy was strongly influenced by the CHMM using the MOV feature set such that no comparable performance could be achieved.

5.3.3. Summary

Two experiments examine the performance of the HMM-MCS and CHMM-MCS: (1) a baseline experiment and (2) an experiment on HMM-MCS and CHMM-MCS in the same setting.

The first experiment provides the baseline results of three representative architectures: (1) HMM architecture, (2) ESN architecture and (3) SVM architecture. The pre-segmented datasets are frequently used to evaluate classifier performances. However, the results cannot necessarily be transferred to the unsegmented setting. The best approach in the pre-segmented setting was the SVM architecture followed by the HMM architecture. But it turned out that the HMM architecture was the best performing approach in the unsegmented dataset followed by the SVM architecture (Table 5.9). The reason for the lowered performance of the SVM architecture can be concluded from additional performance measures. The low recall and the relatedly low F$_1$-measure showed that the SVM architecture provided only an unilateral class distribution, i.e. the classifier almost exclusively assigned decisions to the majority class. In contrast to that, the HMM architecture showed no tendencies for unilateral classifier decisions. Hence, the HMM architecture is clearly the best choice to perform this recognition task in the unsegmented setting.

In second experiment, the HMM-MCS was compared with the CHMM-MCS using the pre-segmented dataset. While one HMM needs to be trained to model each modality and class of the classification problem, the CHMM requires only one model to be trained for each modality. In the given setting, the repeated label was used to form the causal sequence. Two types of CHMM-MCS were evaluated which made use of the restricted CHMM and the unrestricted CHMM. The probability class distribution provided by the CHMM-MCS was furthermore used to study two fusion strategies, namely averaging (+) and multiplication (*). The HMM and restricted CHMM were parameterized to have an almost identical expressiveness. The unrestricted CHMM still had the same number of hidden states as the HMM and restricted CHMM, but had no restriction regarding the transition between the hidden states.

The results showed that the restricted CHMM-MCS significantly outperformed the HMM-MCS when using the +-fusion strategy and feature sets from all modalities (Table 5.11a and Table 5.11b). Other combinations of feature sets and fusion strategies turned out to have a similar or lower performance compared to the best result of the HMM-MCS. The unrestricted CHMM-MCS achieved the best results and outperformed the HMM-MCS in three cases (Table 5.11c). Two out of three of the best unrestricted CHMM-MCS made use of the *-fusion strategy. Furthermore, the restricted and unrestricted CHMM-MCS clearly outperformed the HMM architecture in the first experiment. A more detailed analysis was performed regarding the two fusion strategies. It was shown that the +-fusion strategy is less sensible to wrong classifier decisions (Figure 5.8). However, the final outputs had a broader distribution over classes and, therefore, can be regarded as less certain. The *-fusion strategy was susceptible to wrong unimodal classifier decisions with high certainty (Figure 5.9) because the multiplication tended to favor the most certain ensemble member decisions.

The studies showed that the CHMM-MCS can be successfully applied in the setting of multimodal fusion and the class probability distributions allow the use of multiple fusion strategies. Furthermore, the study provided evidences that results obtained on pre-segmented datasets cannot directly be transferred to unsegmented dataset since they are generally considerably more challenging.

5.4. MFN and Kalman Filter for Affective State Recognition

This section presents the experiments studying the proposed approaches for temporal multimodal classifier fusion using the AVEC 2011 Dataset and the AVEC 2012 Dataset. The first part presents the baseline experiment in which a reference architecture on the

AVEC 2012 Dataset was empirically evaluated. Subsequently, the architectures of the MFN and Kalman filter are specified and analyzed.

5.4.1. AVEC 2011 Participation — MCS Baseline Study

The baseline experiment which was conducted on the AVEC 2011 Dataset was designed to meet the special requirements of an real-world scenario. The goal was to create a temporal multimodal MCS which is less sensitive to noise, e.g. present in annotation, sensor data, extracted features or classification. The architecture addressed these issues by using bagging (Breiman, 1996), robust base classifiers and the preservation of the uncertainty between different components of the architecture. The outputs of the architecture were submitted to the AVEC 2011 (Glodek et al., 2011d). In the following, three MCS are presented, i.e. the audio MCS, the video MCS and the audio-visual fusion MCS. The outputs of the MCS were submitted to the corresponding three sub-challenges.

The MCS which was designed for the audio sub-challenge is depicted in Figure 5.10. The recognition of each class made use of bagging where each bag was composed of five HMM classifiers (Breiman, 1996). The HMM were trained for all three feature sets and for each class and its negation. The number of hidden states and mixture components of the HMM were optimized using an exhaustive parameter search, resulting in the selection of three hidden states and two GMM mixture components with Gaussians having full covariance matrices. In case a feature set turned out to be not informative with respect to a label dimension, the corresponding HMM were pruned. The evaluation showed that all audio feature sets carried information about the class “Arousal”. The classes “Expectancy” and “Power” were supported by the feature set energy, fundamental frequency f_0, LPC and the feature set MFCC. The class “Valence” was only reliably recognized by the HMM making use of the MFCC feature set. The log-likelihoods were accumulated for the time span of a conversational turn according to the assumption that the affective state of a subject changes only slowly and is likely to remain stable within a turn. The classifier fusion was performed by summing up the all available log-likelihoods of one class. The class decisions were obtained by choosing the class with the highest likelihood. The resulting crisp class decisions were submitted to the audio sub-challenge. However, the log-likelihoods were also used for further processing in the audio-visual MCS.

A schematic drawing of the MCS prepared for the video sub-challenge is depicted in Figure 5.11. A number of 300 form and 300 motion features were concatenated to train a ν-SVM having a linear kernel and probabilistic outputs (Schölkopf et al., 2000; Platt, 2000). Due to memory constraints only 10.000 randomly drawn samples from the complete video sequence were used for training. The optimal values for ν were chosen according to the parameter search to be 0.3 for “Arousal” and “Power” and 0.7 for “Expectancy” and

Figure 5.10. Baseline MCS submitted to the AVEC 2011 audio sub-challenge. Each label dimension is recognized by a bag of HMM using a selection of the three feature sets. The likelihoods are normalized for the audio-visual MCS. Crisp decisions are obtained by selecting the classes with the highest likelihoods.

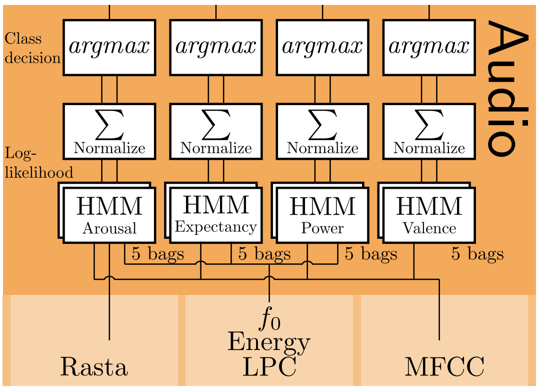
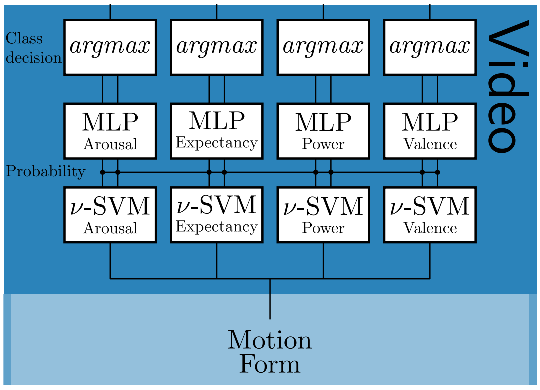


Figure 5.11. Baseline MCS submitted to the AVEC 2011 video sub-challenge. Each class is recognized by a ν-SVM having probabilistic outputs. All outputs are passed to an MLP for each class which refines the class predication. Crisp decisions are obtained by selecting the classes with the highest activations.



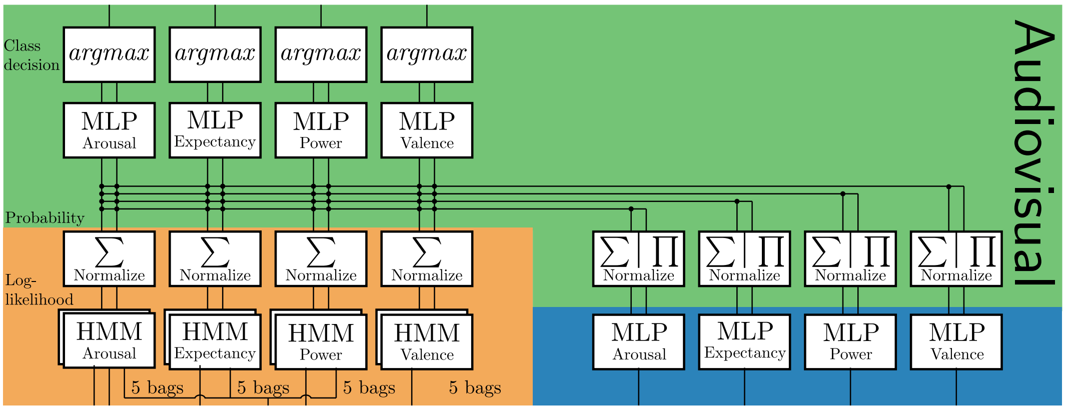


Figure 5.12. Baseline MCS submitted to the AVEC 2011 audio-visual sub-challenge. The outputs of the audio and video MCS are collected and passed an MLP for each class which refines the class predication. Crisp decisions are obtained by selecting the classes with the highest activations.

"Valence". The probability masses of all four ν-SVM were concatenated and used as input to a multi-layer perceptron (MLP) (Haykin, 1999) to obtain a new mapping to the target classes. The class decisions for the video sub-challenge were derived by choosing the class with the highest activation. Analogously to the audio sub-challenge MCS, the activations were used for further processing in the audio-visual MCS.

Table 5.12. Weighted accuracy and unweighted accuracy of the baseline MCS on the AVEC 2011 Dataset. The weighted accuracy (WA) corresponds to the correctly detected samples divided by the total number of samples. The unweighted accuracy (UA) is given by the averaged recall of the two classes of a label dimension. The table provides the results on the development set and the challenge test set.

	Arousal		**Expectancy**		**Power**		**Valence**	
Dataset	↑WA	↑UA	↑WA	↑UA	↑WA	↑UA	↑WA	↑UA
Audio sub-challenge								
Devel. set	66.9%	67.5%	62.9%	58.5%	63.2%	58.4%	65.7%	63.3%
Test set	**63.5%**	**65.7%**	41.1%	41.4%	43.3%	29.9%	**65.4%**	**65.4%**
Video sub-challenge								
Devel. set	58.2%	53.5%	53.5%	53.2%	53.7%	53.8%	53.2%	49.8%
Test set	56.9%	57.2%	47.5%	47.8%	**47.3%**	**47.2%**	55.5%	55.5%
Audio-visual sub-challenge								
Devel. set	69.3%	70.6%	61.7%	60.1%	61.3%	59.1%	68.8%	66.4%
Test set	54.2%	54.3%	**58.5%**	**57.8%**	42.7%	40.0%	44.8%	35.9%

The MCS submitted to the audio-visual sub-challenge relied on the outputs of the audio and video MCS which were already introduced. The design of the architecture is depicted in Figure 5.12. The log-likelihoods of the MCS of the audio sub-challenge were normalized and transformed to probabilistic outputs ranging from zero to one and summing up to one. The activations of the MLP provided by the MCS of the video sub-challenge were as well transformed to meet the probability axioms. In the next step, the probabilistic outputs were collected on word-level and combined using averaging and multiplication. The resulting values were again normalized to range from zero to one and to sum up to one. All values were concatenated to form a new 12-dimensional feature vector containing the predictions of all four classes. The feature vectors were then utilized to derive a new mapping using an MLP. The final decisions were created on the level of conversational turns by normalizing and collecting the outputs within one turn and combining them using multiplication.

The parameters of the three MCS were determined using all available datasets having class labels, i.e. the development and training sets. The MCS which provided the final submission results was trained using the same datasets. Hence, an overfitting on the provided datasets was intentionally done in order to achieve a generalized result on the test set.

After the submission deadline exposed, the rating of the AVEC 2011 was cut down to the class "Arousal" since no submissions showed an adequate performance regarding the other class labels. The audio-visual challenge was withdrawn because of the low number of two participants. Table 5.12 shows the weighted and unweighted accuracies obtained for every sub-challenge, class and the two official data partitions: development set and test set. The proposed MCS of the audio sub-challenge scored second of nine contributions, whereas the MCS submitted to the video sub-challenge achieved the third place among four contributions. The weighted accuracy (WA) corresponds to the ratio of correctly detected samples and the total number of samples. The unweighted accuracy (UA) is given by the averaged recall of the actual class and its negation (Schuller et al., 2010).

The accuracies of the sub-challenge MCS ranged generally close to chance levels. As expected due to the overfitting, the results achieved on the development set regarding all sub-challenges and classes were commonly higher than the results on the test set. The audio sub-challenge MCS achieved good test accuracies of 63.5% and 65.4% for the classes "Arousal" and "Valence". The accuracies of the classes "Expectancy" and "Power" were below chance level, i.e. 41.1% and 43.2%, respectively. Similarly, the video sub-challenge MCS achieved good test accuracies of 56.9% and 55.5% for the classes "Arousal" and "Valence", whereas the classes "Expectancy", "Power" obtained only accuracies below chance level, i.e. 47.5% and 47.6%, respectively. However, the results of the video sub-challenge MCS were consistently closer to 50% than the results of the audio sub-challenge MCS. According to the weighted accuracies, the outputs of the audio-visual sub-challenge MCS appeared to be even less informative. The classes "Arousal", "Expectancy", "Power" and "Valence" had test accuracies of 54.2%, 58.5%, 42.7% and 44.8%, respectively. Obviously, the performance of the multimodal MCS was below the performance of the unimodal MCS.

The audio classification architecture showed first good results, whereas the video architecture did not provided any convincing results. This can most probably be related to the utilized video features which were too generic. The low performance of the audio and the video classification architectures in turn led to the low performance of the audio-visual classification architecture. The multimodal and temporal fusion was performed heterogeneously, i.e. the combination took place on different stages of the architecture and required several normalization steps. In addition to that the architectures combined first the outputs of the modalities and then the aggregated the decisions of multiple time steps. However, it is not clearly in which order the combination should ideally be performed or whether it should even be performed simultaneously.

The participation in the AVEC 2011 was the beginning of the development and extensive research of temporal multimodal MCS fusion techniques which are empirically evaluated in the follow sections.

5.4.2. AVEC 2012 Participation — Markov Fusion Network

The MFN was developed for the submission to the AVEC 2012. All classifiers were extended to not only provide a probabilistic output, but also an additional confidence measure. Classifier robustness was again achieved by bagging (Breiman, 1996) and by classifier functions of low complexity which had proven to be well-suited for the recognition of the task at hand. The final decisions were obtained by averaging the individual outputs of a classifier ensemble (Breiman, 1996). The calculation of the confidences was based on the standard deviation σ of the classifier ensemble outputs within a window of two seconds shifted over

Table 5.13. Absolute correlation coefficient (CC) and root-mean squared error (RMS) of the MFN MCS on the AVEC 2012 Dataset. The absolute correlation coefficient (CC) is obtained by taking the absolute value of the averaged correlation coefficients of each recording. The root-mean squared measure (RMS) error is obtained by averaging the root-mean squared difference of the predictions and ground truth over all sessions. The table provides the results on the development set and challenge test set.

	Arousal		**Expectancy**		**Power**		**Valence**		Mean	
Dataset	↑CC	↓RMS	↑CC	↓RMS	↑CC	↓RMS	↑CC	↓RMS	↑CC	↓RMS
Audio sub-challenge										
WLSC Test	0.080	3.8	0.130	189.9	0.048	4.1	0.038	4.3	0.074	50.5
WLSC Devel.	0.053	4.3	0.180	194.2	0.081	3.8	0.093	5.6	0.102	52.0
Video sub-challenge										
FCSC Test	0.069	24.7	0.140	1133.4	0.022	27.1	0.180	23.3	0.103	302.1
WLSC Test	0.075	4.3	0.182	200.2	0.012	4.3	0.185	4.6	0.113	53.3
FCSC Devel.	0.119	22.2	0.198	1073.2	0.182	22.9	0.137	27.8	0.159	286.5
WLSC Devel.	0.127	4.4	0.195	190.7	0.128	3.9	0.105	5.7	0.139	51.2
Audio-visual sub-challenge										
FCSC Test	0.294	32.2	0.268	1178.9	0.268	30.8	0.131	21.7	0.201	315.9
WLSC Test	0.141	4.6	**0.223**	197.0	0.183	197.0	0.162	4.4	0.177	52.6
FCSC Devel.	0.202	31.4	0.204	1301.4	0.393	27.0	0.148	26.8	0.237	346.6
WLSC Devel.	0.088	5.5	0.211	191.1	0.245	4.5	0.150	5.5	0.174	51.6

the sequence. The measure itself was derived by evaluating $1 - \sigma$ multiplied by a ratio of how many decisions in the window were available. In case the window contained no classifier outputs, the corresponding confidence was set to zero. Hence, confidence values were higher, if the consensus of the classifier was strong and many decisions were available. Large parts of the architecture presented in the previous section were revised. Among others changes, the video sub-challenge MCS was replaced by a new architecture which made use of the high-level features extracted from CERT. The video sub-challenge MCS provided its decisions based on a bag of five NBC which was derived for each class. The audio sub-challenge MCS performed the classification on word-level using a fixed-length feature representation obtained with the help of HMM according to Bicego et al. (2003). The fixed-length features were classified using five bags of random forests (Breiman, 2001). Outputs of the audio and video MCS were systematically rejected in case of a low confidence measure. The rejection was performed using a rejection rate which defined the ratio of decisions to be discarded. The MFN was used to combine the outputs of the audio, video and audio-visual MCS. Since only a two-class classification problem had to be solved, the distributional potential of the MFN was omitted.

The smoothness and the data potential parameters of the MFN, as well as the rejection rates for audio and video outputs, were optimized on the development set for each sub-challenge and class. During conversational turns, the smoothness potential parameters were lifted to higher values for stronger smoothing. According to cross-validation, the classes "Arousal" and "Valence" benefit from a strong similarity over time for all sub-challenges. Therefore, 192 was chosen for the smoothness potential parameter. At the beginning of each conversational turn, the smoothness parameter was decreased to 8 in the corresponding frame. The data parameters were set to 0.5 for the outputs of the video MCS and to 0.3

for the outputs of the audio MCS. A detailed listing of the parameter setting is provided in Table A.1 in the Appendix.

The influence of the parameters and of the outputs provided by the audio and video MCS to the MFN cannot be directly conceived due to the complexity of the performed calculations. One aspect is that there are far more outputs provided by the video MCS than outputs of the audio MCS. Furthermore, the ratio of rejected samples has a strong impact, e.g. 90% of the video outputs but only 50% of the audio outputs were rejected in the audio-visual sub-challenge of the class "Arousal".

Table 5.13 shows the absolute correlation coefficients (CC) and the root-mean-square (RMS) errors achieved with the decisions provided by the submitted audio, video and audio-visual sub-challenge MCS. The final rating of the sub-challenges were performed by taking the mean of the absolute correlation coefficients of all four classes. In most cases, the achieved class results clearly outperformed the baseline results of Table 4.5. In particular the audio-visual sub-challenge MCS regarding the classes "Expectancy" and "Valence" obtained good performances. However, comparing the CC to the performance of the logarithmic function and the challenge winner, the achieved results are not competitive. Only the WLSC regarding the class "Expectancy" in the audio-visual sub-challenge MCS achieved a higher CC than the challenge winner.

The average performance of the submitted architecture can be related to the ambitious endeavor of us to recognize the emotional state without further context knowledge, such as the time or the identity of the emotionally colored interlocutor. As already pointed out, the annotation of emotional states is very demanding. Errors can originate from the annotation process which depends on the interpretation and subjective perception of the labels, the annotators mood and the use of the annotation range and scale, but also on the interface of the annotation tool. Especially, the temporal dependencies of the classes in the AVEC 2012 Dataset which were identified in Section 4.1.3.3 urged us to refrain from the continuously labeled dimensions and to return to the crisp class assignments.

5.4.3. AVEC 2011 — Markov Fusion Network

The follow-up experiment analyses the MFN on the AVEC 2011 Dataset in the context of three classifier architectures which model early, mid-level and late fusion of decisions. The so-called FRT, RFT and RTF architectures were introduced in Section 3.5.2. Each architecture realizes three steps, namely the F-step (fusion of decisions), the R-step (rejection of uncertain decisions) and the T-step (temporal integration of the decisions). The three architectures are created by varying the F-step. The empirical evaluation focuses on:

1. Early, mid-level and late fusion of decisions
2. A real-world scenario with sensor failures

3. Confidence-based rejection option
4. Temporal integration of context information (conversational turns)
5. Additional weighting of predictions using classifier confidences
6. Online and offline processing modes.

The temporal integration of context information and the additional weighting of predictions were performed in the same manner as in the previous experiment on the AVEC 2012 Dataset. However in this experiment, the confidences were not only used to implement the reject option, but also to weight the data potential by multiplying the data potential parameter **k** with the confidence measures over time. The decisions having the lowest confidences were rejected first. A rejection rate defined the ratio of decisions to be discarded. All architectures were evaluated utilizing the offline and online processing modes as described in Section 3.5. The window in the online processing mode was set to five seconds and the estimate of the forth second in the window was stored as the final output. The MFN utilized the outputs of the last estimation process as initial inputs to speed-up convergence. The initial inputs were shifted according to the elapsed time of the last estimation process.

5.4.3.1. Unimodal Results

Table 5.14 shows the performance of the raw unimodal audio and video classifier outputs. Accuracies and F_1-measures of the classifiers without rejections are provided in Table 5.15a. The accuracies of all classes ranged close to 58%. An exception constituted the audio classifier of the class "Arousal" for which a marginally higher accuracy of 61.8% was observed. The audio classifiers recognizing the classes "Expectancy", "Power" and "Valence" rendered slightly unbalanced class distribution as indicated by the F_1-measures. The video classifier provided generally lower accuracies, but showed less strong tendencies for unbalanced class distributions compared to the audio classifier. Table 5.15b and Table 5.15c list the classifier performances when rejecting 10% and 90% of the decisions. Generally, the audio and video classifiers improved their recognition rates. In Table 5.14, the highest accuracies of 62.7%, 59.1% and 59.1% with respect to the classes "Arousal", "Expectancy" and "Power" were achieved when using the audio classifier and rejecting 90% of the samples. The highest accuracy with regard to the class "Valence" of 67.8% was obtained when using the video classifier and rejecting 90% of the samples. The F_1-measures remained rather stable which indicates that the class distributions had not changed significantly although rejection was performed. However, the rejection of decisions resulted in a sparse stream of classifier predications which were reconstructed using the MFN.

In the next step, the performance of the MFN applied to single modalities for online and offline processing modes is analyzed. Parameters and rejection rates were determined using the training sets of the cross-validation. The MFN parameters utilized are provided in Table A.3a in the Appendix. The results are presented in Table 5.16. The calculated performance measures based on the reconstructed decisions, i.e. decisions of all frames were evaluated. The recognition rates were similar or higher compared to the results of the raw unimodal audio and video classifier outputs. For example, the accuracies of the class “Arousal” increased to 71.1% and 63.4% when applying the MFN in online processing mode to the audio outputs and video outputs, respectively. The accuracies of the classes “Expectancy”, “Power” and “Valence” remained almost unchanged. However, it has to be remarked that, in contrast to the raw unimodal outputs, the decisions provided by the MFN were available for every frame. The F_1-measures of the MFN audio outputs concerning “Expectancy”, “Power” and “Valence” indicate that the classifier outputs became increasingly unbalanced which can be related to the sparse and uneven input class distribution. The class distributions provided by the video MFN were clearly more stable with respect to

Table 5.14. Accuracies and F_1-measures of the unimodal word-wise and frame-wise recognition of the audio and video classifiers for different rejection rates on the AVEC 2011 Dataset. All values averaged with standard deviation. Arrows indicate how to rate the measures.

(a) No rejection

Audio	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	61.8±3.6 %	58.9±6.3 %	57.5±9.4 %	57.5±7.9 %
↑F_1	65.8±3.8 %	16.4±7.1 %	69.6±9.3 %	70.1±6.8 %
↑$\overline{F}_1$	56.7±3.4 %	72.6±5.2 %	24.7±6.6 %	24.9±8.4 %

Video	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	57.0±4.3 %	54.7±4.0 %	55.7±2.8 %	59.9±7.4 %
↑F_1	60.9±5.1 %	49.6±9.4 %	57.4±11.3 %	67.1±11.5 %
↑$\overline{F}_1$	51.3±9.3 %	56.6±10.7 %	48.7±12.2 %	43.5±7.1 %

(b) 10% rejection rate

Audio	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	62.0±3.5 %	58.7±6.3 %	57.5±9.4 %	57.5±7.6 %
↑F_1	66.1±3.8 %	15.8±7.3 %	69.6±9.3 %	70.1±6.6 %
↑$\overline{F}_1$	56.9±3.3 %	72.5±5.2 %	24.5±6.2 %	24.4±8.8 %

Video	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	57.7±4.3 %	55.0±3.9 %	55.7±2.8 %	60.4±7.8 %
↑F_1	61.5±5.2 %	50.3±9.5 %	57.2±11.3 %	67.6±11.9 %
↑$\overline{F}_1$	51.8±9.3 %	56.6±10.6 %	49.1±12.2 %	43.5±7.5 %

(c) 90% rejection rate

Audio	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	**62.7±3.1 %**	**59.1±6.6 %**	**59.1±9.9 %**	57.5±7.6 %
↑F_1	67.3±2.8 %	15.9±8.8 %	71.1±9.4 %	70.1±6.6 %
↑$\overline{F}_1$	56.4±4.3 %	72.7±5.6 %	24.2±5.3 %	24.4±8.8 %

Video	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	59.5±4.3 %	56.9±1.6 %	54.4±4.1 %	**67.8±10.4 %**
↑F_1	65.0±5.1 %	56.0±13.5 %	54.9±10.3 %	75.6±11.8 %
↑$\overline{F}_1$	50.6±9.3 %	52.5±14.0 %	50.7±10.0 %	41.0±12.7 %

Table 5.16. Accuracies and F$_1$-measures of the reconstructed unimodal streams using the MFN on the AVEC 2011 Dataset. All values averaged with standard deviation. Arrows indicate how to rate the measures.

(a) Audio

Online	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	**71.7±6.4 %**	56.2±6.7 %	58.3±9.8 %	60.2±10.6 %
↑F$_1$	74.7±6.2 %	5.6±3.5 %	69.6±9.4 %	72.9±8.3 %
↑$\overline{\text{F}}_1$	67.3±7.6 %	71.3±5.5 %	30.7±7.0 %	21.6±13.3 %
A. rej.	10%	50%	90%	50%

Offline	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	68.7±6.3 %	56.4±6.3 %	55.5±10.4 %	59.0±10.5 %
↑F$_1$	72.6±5.6 %	9.5±3.6 %	69.4±9.7 %	72.8±8.1 %
↑$\overline{\text{F}}_1$	63.1±8.6 %	71.1±5.5 %	10.9±12.4 %	11.9±13.8 %
A. rej.	0%	90%	0%	50%

(b) Video

Online	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	63.4±1.9 %	**58.7±3.9 %**	**59.0±4.9 %**	**65.8±9.5 %**
↑F$_1$	66.2±5.3 %	55.2±11.6 %	60.7±13.9 %	73.2±11.7 %
↑$\overline{\text{F}}_1$	57.0±11.8 %	57.9±14.0 %	48.4±16.7 %	45.9±7.0 %
V. rej.	50%	90%	0%	90%

Offline	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	62.5±4.0 %	57.9±4.8 %	58.5±5.0 %	64.4±9.7 %
↑F$_1$	65.0±4.8 %	51.5±12.7 %	59.9±14.2 %	72.0±11.3 %
↑$\overline{\text{F}}_1$	56.6±14.2 %	58.3±14.7 %	48.1±16.8 %	45.7±9.2 %
V. rej.	50%	10%	0%	90%

the F$_1$-measures when compared with the audio MFN. Furthermore, the offline processing mode apparently outperformed the online processing which can be attributed to the small window size. The smoothing in offline processing mode integrated past and the future class decisions, whereas the window in the online processing was limited to a small segment of five seconds. Hence, no information of future events was propagated to the position of the window which provided the final output. This procedure resembles the annotation process and, hence, is likely to be the reason for the good performance.

5.4.3.2. Multimodal Results

Table 5.19a, Table 5.19b and Table 5.19c provide the performance measures of the multimodal FRT, RFT and RTF architectures. All three architecture provided continuously decisions even if one modality got temporally lost. The Table 5.19a shows the performance of the FRT classifier architecture. Both, the online and offline processing modes rendered similar accuracies and F$_1$-measures. The results showed only a marginal improvement compared with the results of the reconstructed unimodal MFN decisions. The accuracies of the classes “Arousal” and “Valence” in the online processing mode achieved 66.8% and 62.9%, respectively. They were lower compared with the accuracies of the unimodal MFN outputs

Table 5.18. Accuracies and F_1-measures of the reconstructed multimodal streams using three different MFN architectures on the AVEC 2011 Dataset. All values averaged with standard deviation. Arrows indicate how to rate the measures.

(a) FRT architecture (Figure 3.7a)

Online	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	66.8±4.4 %	60.3±3.9 %	58.4±3.7 %	62.9±8.7 %
↑F_1	70.5±3.5 %	47.9±10.6 %	64.8±8.4 %	73.3±8.5 %
↑$\overline{F}_1$	61.5±8.0 %	66.7±7.1 %	44.5±11.8 %	33.3±12.5 %
Offline	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	66.8±4.4 %	60.0±3.9 %	58.6±8.0 %	62.2±9.2 %
↑F_1	70.7±3.7 %	48.5±11.3 %	67.7±10.1 %	72.7±8.8 %
↑$\overline{F}_1$	61.1±8.2 %	65.8±7.5 %	38.3±4.4 %	32.3±12.4 %
A. rej.	10%	90%	90%	90%
V. rej.	90%	90%	90%	90%

(b) RFT architecture (Figure 3.7b)

Online	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	68.8±5.2 %	**62.1±2.9 %**	**61.0±6.0 %**	**64.3±9.0 %**
↑F_1	72.5±4.6 %	46.5±12.7 %	67.2±8.7 %	72.9±10.5 %
↑$\overline{F}_1$	63.4±8.6 %	69.6±5.2 %	45.7±14.9 %	40.3±10.3 %
Offline	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	**68.2±4.6 %**	**60.9±4.1 %**	**59.9±6.2 %**	62.6±9.2 %
↑F_1	72.2±4.3 %	42.0±11.9 %	64.4±10.6 %	72.3±9.6 %
↑$\overline{F}_1$	62.4±7.5 %	69.6±5.7 %	46.4±16.7 %	35.2±12.3 %
A. rej.	10%	50%	90%	90%
V. rej.	50%	90%	0%	90%

(c) RTF architecture (Figure 3.7c)

Online	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	**68.9±8.2 %**	59.2±5.2 %	54.6±3.3 %	64.1±9.1 %
↑F_1	66.7±10.0 %	41.2±8.1 %	57.2±1.8 %	68.1±14.7 %
↑$\overline{F}_1$	70.3±7.9 %	68.7±4.3 %	50.7±9.4 %	54.0±7.3 %
Offline	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	65.8±5.6 %	56.8±8.7 %	51.4±6.7 %	**63.4±10.2 %**
↑F_1	68.7±2.5 %	28.9±17.4 %	63.4±4.0 %	69.3±14.9 %
↑$\overline{F}_1$	61.0±11.8 %	67.8±8.7 %	25.9±18.2 %	46.9±11.2 %
A. rej.	10%	50%	90%	50%
V. rej.	50%	90%	0%	90%

in Table 5.16. However, the classes “Expectancy” and “Power” had slightly higher accuracies of 60.3% and 58.4% compared with the corresponding accuracies in Table 5.16.

Table 5.19b shows the results of the RFT architecture which realizes the temporal and multimodal fusion simultaneously. Compared to the previous architecture the accuracies had clearly improved. Furthermore, the RFT architecture outperformed the accuracies of the unimodal MFN outputs in case of the classes “Expectancy” and “Power” utilizing the online and offline processing modes. The accuracies of the classes “Expectancy” and “Power” using the online processing mode were 62.1% and 61.0%, respectively. The accuracies of the classes “Expectancy” and “Power” using the offline processing mode were 60.9% and 59.9%, respectively. The accuracies achieved by the RFT architecture for the classes “Arousal” and

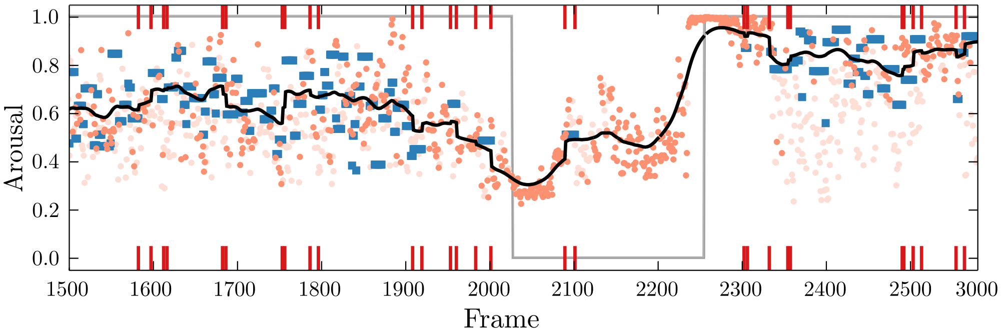


Figure 5.13. Predications for the class "Arousal" derived from multiple modalities combined by the MFN RFT architecture using data from the AVEC 2011 Dataset. The orange dots and the blue squared-shaped markers correspond to the video and audio predictions. Markers in lighter color have been rejected and do not contribute to the MFN. The red ticks indicate turns or pauses within the conversation. The thick black curve corresponds to the fusion result. The light gray curve displays the ground-truth. The parameters used are $w_{normal} = 128, w_{turn} = 4, k_{\text{video}} = .5$ and $k_{\text{audio}} = 5$.

"Valence" were lower than the corresponding accuracies using the unimodal MFN. The class "Arousal" achieved an accuracy of 68.8% in online processing mode and 68.2% in offline processing mode. The accuracies of the class "Valence" were 64.3% and 62.6% utilizing the online and offline processing modes, respectively.

The performance measures of the RTF architecture are shown in Table 5.19c. The RTF architecture achieved a performance similar to the RFT architecture regarding the classes "Arousal" and "Valence", i.e. accuracies of 68.9% and 64.1% using the online processing mode. The offline processing mode achieved accuracies of 65.8% and 63.4% for the classes "Arousal" and "Valence", respectively. The accuracies of the classes "Expectancy" and "Power" were lower than in the RFT architecture. The online processing mode achieved 59.2% and 54.6% with respect to the classes "Expectancy" and "Power", respectively. The offline processing mode achieved 56.8% and 51.4% with respect to the classes "Expectancy" and "Power", respectively. The F_1-measures indicate that the outputs were generally more balanced then the outputs of the RFT architecture and the unimodal fusion. The lowered performance compared with the RFT architecture can be related to the fact that, due to the early temporal fusion, less information was available in each channel during temporal fusion.

Figure 5.13 provides an example of the output of the MFN on the AVEC 2011 Dataset. The classifier outputs for the class "Arousal" based on the audio (blue squares) and video (orange dots) channel are plotted along the time axis. Only the positive probability mass of the two-class classification problem per channel is shown. Decisions with low confidences which were rejected are plotted in a lighter color. The solid black curve represents the estimate of the MFN. The red ticks on the top and bottom of the figure are the beginnings of conversational turns in which the temporal smoothness was suppressed based on the assumption that an affective state remains stable within a turn. As to be seen in the figure, the procedure led to the formation of stable regions and spots of changes whenever a turn occurred.

5.4.4. AVEC 2011 — Kalman Filter for Classifier Fusion

So far, a baseline experiment and an experiment using the MFN were evaluated on the AVEC 2011 Dataset. In this experiment, the Kalman filter for classifier fusion is examined. The raw unimodal audio and video classifier outputs provided in the MFN experiment were also used as inputs to the Kalman filter for classifier fusion which makes the comparison of both methods more intuitive.

Figure 5.14 shows the Kalman filter for classifier fusion in the same setting as the MFN in Figure 5.13. Again, the same classifier outputs for the class “Arousal” based on the audio (blue squares) and video (orange dots) channels are plotted along the time axis. Decisions with low confidences which were rejected are plotted in a lighter color. The solid black curve represents the estimate of the Kalman filter, whereas the gray line depicts the ground truth. The area surrounding the Kalman filter estimate corresponds to the associated variance. It was scaled by the factor of ten for illustration purposes.

5.4.4.1. Unimodal Results

Table 5.21a and Table 5.21b show the performance of the unimodal Kalman filter fusion. Each table shows the results in the upper part and the utilized parameters in the lower part. Analogously to the MFN, the parameters were optimized for each class independently using the training sets of the cross-validation. The measurement prior $\tilde{z}_{mt}$ was set to 0.5 and the corresponding observation noise $\tilde{r}_m$ was set a magnitude higher than the assigned r. Hence, estimates converged slowly to the prior in case measurements were continuously missing. The accuracies of the audio Kalman filter fusion regarding the classes “Arousal” and “Expectancy” improved when compared with the accuracies of the audio-based MFN shown in Table 5.17a. The accuracy of 74.3% regarding the class “Arousal” will later on turn out to be the highest accuracy in the complete study. The rejection rate used to perform the audio Kalman filter fusion for the class “Arousal” was selected to be 10%. In contrast, the audio Kalman filter fusion for the class “Expectancy” and “Power” utilized rejection rates of 90% which indicates that the raw audio classifier decisions were less reliable. The accuracies of the class “Expectancy” and “Power” were 57.5% and 56.5%, respectively. The audio Kalman filter fusion performed for the class “Valence” achieved an accuracy of 59.7%. In case of the classes “Expectancy”, “Power” and “Valence”, the F_1-measures of the classes “Expectancy”, “Power” and “Valence” show that the outputs of the Kalman filter tended to be unilateral. The process noises were generally set to be low and the observation noises ranged from 0.1 to 0.75. The performance measures of the video Kalman filter fusion are shown in Table 5.21b. Analogously to the audio Kalman filter fusion, the accuracies of 64.5%

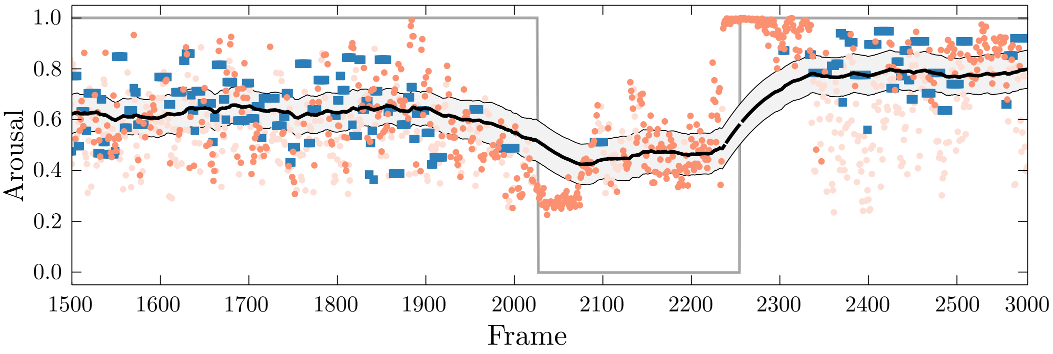


Figure 5.14. Predications for the class "Arousal" derived from multiple modalities combined by the Kalman filter for classifier fusion using data from the AVEC 2011 Dataset. The orange dots and the blue squared-shaped markers correspond to the video and audio predications. Markers in lighter color have been rejected and do not contribute to the Kalman filter. The thick black curve corresponds to the fusion result, while the area around the curve corresponds to the variance determined by the Kalman filter (scaled by 10 for illustration purposes). The light gray curve displays the ground-truth. The parameters used are $q_{\text{audio}} = 10^{-6}$, $q_{\text{video}} = 10^{-5}$ and $r = 0.75$.

and 59.1% regarding the classes "Arousal" and "Expectancy" were higher than the accuracies of the corresponding video-based MFN which are shown in Table 5.17b. The accuracies of the classes "Expectancy" and "Power" achieved 58.9% and 65.4%. and, therefore, performed lower than the corresponding accuracies of the video-based MFN. In general, the accuracies of the video Kalman filter fusion were slightly lower than accuracies of the audio Kalman filter fusion. However, the F_1-measures of the video Kalman filter fusion tended to be more balanced than the F_1-measures of the audio Kalman filter fusion. The selected parameters for the process noises and the observation noises were similar to the parameters of the audio Kalman filter fusion. This finding can be related to the identical classifier output range of $[0, 1]$.

5.4.4.2. Multimodal Results

The performance measures of the multimodal Kalman filter fusion are shown in Table 5.22. The accuracies of the classes "Expectancy" and "Power" achieved 62.5% and 61.8% which was higher than the accuracies of the corresponding unimodal Kalman filter fusion. The multimodal Kalman filter fusion regarding the class "Arousal" achieved an accuracy of 68.5% and, hence, was outperformed by the audio Kalman filter fusion which achieved an accuracy of 74.3%. Analogously, the multimodal Kalman filter fusion yielded an accuracy of 64.2% and, therefore, was outperformed by video Kalman filter rendering an accuracy of 65.4%. The mid-field performance of the multimodal Kalman filter fusion regarding class "Arousal" can most probably be related to the combination of two sources of decisions with different quality. Although, the unimodal Kalman filter fusion outperformed the multimodal Kalman filter fusion in two cases, it has to be pointed out that the class distribution of the

Table 5.20. Averaged accuracies and F$_1$-measures of the reconstructed unimodal streams using the Kalman filter for classifier fusion on the AVEC 2011 Dataset. The lower part of the tables list the parameter assignments. All values averaged with standard deviation. Arrows indicate how to rate the measures.

(a) Audio

Audio	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	**74.3±6.6 %**	57.5±6.2 %	56.5±11.2 %	59.7±11.0 %
↑F$_1$	77.4±6.5	12.5±6.2	69.1±10.2	72.9±8.5
↑$\overline{\mathrm{F}}_1$	69.4±8.2	71.8±5.3	22.7±8.9	16.2±14.1
Reject	10%	90%	90%	50%
q_{audio}	10^{-7}	10^{-7}	10^{-5}	10^{-4}
r	0.1	0.1	0.1	0.75

(b) Video

Video	**Arousal**	**Expectancy**	**Power**	**Valence**
↑Acc.	64.5±2.9 %	**59.1±4.7 %**	**58.9±5.0 %**	**65.4±9.8 %**
↑F$_1$	67.9±4.2	56.7±11.1	60.6±13.8	73.3±11.4
↑$\overline{\mathrm{F}}_1$	57.3±12.7	56.4±17.5	48.2±17.1	44.2±8.5
Reject	50%	90%	0%	90%
q_{video}	10^{-7}	7.5^{-6}	5^{-5}	10^{-4}
r	0.1	0.1	0.1	0.75

multimodal Kalman filter fusion were generally more balanced. Furthermore, the estimates of the multimodal Kalman filter fusion were less susceptible against sensor failures.

5.4.4.3. Comparison to AVEC 2011 Submission Results

Table 5.23 summarizes the performances of the MFN and the Kalman filter for classifier fusion and compares them to the official challenge baseline and winner of the AVEC 2011. The accuracies provided in the table were obtained by averaging the accuracies of all four affective classes. The averaged baseline accuracies of the audio and video sub-challenges achieved 45.05% and 47.50%, respectively. Meng and Bianchi-Berthouze (2011) from the University College London (UCL) were the winner of the audio sub-challenge. They achieved an average accuracy of 53.28%. The video sub-challenge was won by Ramirez et al. (2011) from the University of Southern California (USC) with an average accuracy of 61.03%, The averaged accuracies of the raw audio classifier already outperformed the winner of the audio sub-challenge. However, the average accuracy of the raw video classifier was always lower than accuracy of the winner of the video sub-challenge. The raw classifier results using high rejection rates performed best and achieved 59.60% using the audio modality and 59.65% using the video modality. However, it is important to emphasize that these and the following results based on a repartitioned (subject-independent) dataset composed of the development and training sets for which labels were available, as explained in Section 4.1.3.2.

The unimodal offline MFN achieved averaged accuracies of 59.90% and 60.83% using audio and video channels, respectively. In two out of three cases, the multimodal MFN

Table 5.22. Accuracies and $\mathbf{F_1}$-measures of the reconstructed multimodal streams using the Kalman filter for classifier fusion on the AVEC 2011 Dataset. Lower part of table lists the parameter assignments. All values averaged with standard deviation. Arrows indicate how to rate the measures.

Audio-visual	**Arousal**	**Expectancy**	**Power**	Valence
↑Acc.	68.5±5.7 %	62.5±4.9 %	61.8±6.6 %	64.2±9.3 %
↑F_1	72.6±4.2	42.2±15.7	69.1±7.6	72.6±10.9
↑$\overline{\mathrm{F}}_1$	59.7±15.1	71.1±5.8	43.5±18.0	43.7±3.0
A. rej.	0%	0%	0%	90%
V. rej.	50%	50%	0%	10%
q_{audio}	10^{-6}	5^{-6}	10^{-7}	5^{-5}
q_{video}	10^{-5}	10^{-5}	10^{-5}	10^{-4}
r	0.75	0.1	0.1	0.75

outperformed the unimodal offline MFN. The best multimodal offline MFN average accuracy of 62.09% was achieved by the RFT architecture. The unimodal online MFN achieved averaged accuracies of 61.60% and 61.73% using the audio and video modalities, respectively. The best multimodal online MFN average accuracy was again achieved by RFT architecture which yielded 64.05%.

All previous results were outperformed by Kalman filter for classifier fusion. The unimodal Kalman filter fusion achieved 62.00% using the raw audio classifier outputs and 61.98% using the raw video classifier outputs. The multimodal Kalman filter fusion achieved the highest averaged accuracy of 64.25% in the study.

5.4.5. AVEC 2013 Challenge — Kalman Filter for Classifier Fusion

The Kalman filter for classifier fusion was further applied in the submission to the AVEC 2013 challenge (Kächele et al., 2014a). Unfortunately, the results and the paper were not ready in time such that a participation to the challenge was not possible. However, the challenge organizers kindly agreed to evaluate the submitted predictions on the test set such that comparative results can be provided. The submitted approach featured the learning of two Kalman filter parameters, i.e. noise variance q and the error variance r_m. The labels provided in the challenge were normalized to the range from zero to one and the measurement priors were set to 0.5. The absolute correlation coefficients (CC) of the Kalman filter and three competing approaches are listed in Table 5.24, i.e. the baseline results, the winner of the AVEC 2013 and the approach of Sánchez-Lozano et al. (2013). The audio baseline achieved the best CC on the development set for all classes. However, the same approach achieved also the lowest performance on the test set which indicates that the baseline audio architecture was clearly overfitted. The accuracies of the unimodal Kalman filter fusion on the development set were comparably low. However, the results on the test set achieved a clearly better performance, e.g. the video Kalman filter fusion achieved an averaged CC of 0.130 on the test set which was the best result of all unimodal approaches in the table.

Table 5.23. Accuracies of the AVEC 2011 baseline, the AVEC 2011 winner, the unimodal results, offline MFN, online MFN and Kalman filter for classifier fusion on the AVEC 2011 Dataset. The accuracies are given by the unweighted average of all four classes. Researchers from the University College London (UCL) won the audio sub-challenge. Researchers from the University of Southern California (USC) won the video sub-challenge. Arrows indicate how to rate the measures.

Approach	↑Acc. Audio	↑Acc. Video	↑Acc. Audio-visual
Baseline	45.05%	47.50%	57.93%
Winner	53.28% [UCL]	61.03% [USC]	n/a
Unimodal raw	58.93% [Rej. 0%] 58.93% [Rej. 10%] 59.60% [Rej. 90%]	56.83% [Rej. 0%] 57.20% [Rej. 10%] 59.65% [Rej. 90%]	n/a
Offline MFN	59.90%	60.83%	61.09% [FRT] 62.09% [RFT] 59.35% [RTF]
Online MFN	61.60%	61.73%	62.10% [FRT] 64.05% [RFT] 61.70% [RTF]
Kalman filter	62.00%	61.98%	64.25%

The audio-visual approaches achieved generally higher CC. Sánchez-Lozano et al. (2013) achieved an averaged CC of 0.134, whereas the winner of the AVEC 2013 challenge achieved an averaged CC of 0.141. The multimodal Kalman filter fusion achieved an averaged CC of 0.160 on the test set and, therefore, outperformed all previous results.

5.4.6. Summary

Five experiments were conducted to examine temporal multimodal fusion architectures for the recognition of affective states. A special focus was set on the analysis of the MFN and Kalman filter for classifier fusion which were proposed in this work.

The first experiment utilized the AVEC 2011 Dataset and examines the performance of a classifier architecture which was based on conventional classifier and fusion approaches. The outputs of the architecture were submitted to the AVEC 2011 and achieved the second place out of nine participants in the audio sub-challenge and the third place out of four participants in the video sub-challenge (Table 5.12). The audio-visual challenge was withdrawn because of the low number of two participants.

The second experiment examines the MFN based on AVEC 2012 Dataset (Table 5.13). The AVEC 2012 Dataset extended the former AVEC 2011 Dataset by featuring a continuous annotation. The proposed classifier architecture was the first one which made use of the MFN. However, the submitted results of the MFN classifier architecture achieved only a mid-field rank. As already noted in the paper accompanying our challenge submission Glodek et al. (2012c) and Section 4.1.3.3, the continuous annotations showed strong temporal characteristics. These characteristics together with the performance measures of the challenge, i.e. absolute correlation coefficient (CC), can be exploited to achieve high recognition results

Table 5.24. Absolute correlation coefficient of AVEC 2013 baseline, the Kalman filter for classifier fusion, the AVEC 2013 winner, Sánchez-Lozano et al. (2013) on the AVEC 2013 Dataset. Arrows indicate how to rate the measures.

	Development set			Test set		
	Arousal	Valence	Average	Arousal	Valence	Average
Approach	↑CC	↑CC	↑CC	↑CC	↑CC	↑CC
	Audio sub-challenge					
Baseline	**0.338**	**0.257**	**0.298**	0.089	0.090	0.089
Kalman Filter	0.094	0.103	0.099	0.107	0.114	0.111
	Video sub-challenge					
Baseline	0.337	0.157	0.247	0.076	0.134	0.105
Kalman Filter	0.153	0.098	0.126	0.118	0.142	0.130
	Audio-visual sub-challenge					
Sánchez-Lozano et al. (2013)	0.167	0.192	0.180	0.135	0.132	0.134
AVEC 2013 Winner[a]	n/a	n/a	n/a	n/a	n/a	0.141
Kalman Filter	0.134	0.156	0.145	**0.150**	**0.170**	**0.160**

[a] Visit `http://sspnet.eu/avec2013/` for details. The paper of Meng et al. (2013) contains no additional results regarding the affect sub-challenges (last visited 10/06/2015).

(Table 4.5). As a consequence, the consecutive experiments regarding the MFN and Kalman filter for classifier fusion were performed on the AVEC 2011 Dataset.

The third experiment examines two unimodal MFN architectures and three multimodal MFN architectures. All approaches were evaluated using the online and offline processing mode. The three MFN architectures realize early fusion (FRT), mid-level fusion (RFT) and late fusion (RTF) of decision. The unimodal MFN architecture were able to improve the raw unimodal classifier outputs only marginally (Table 5.14 and Table 5.16). However, the unimodal MFN provide class decisions for every frame of the recording, in contrast to the raw classifier decisions. The three multimodal MFN architectures further improved the recognition performance of the unimodal MFN architecture (Table 5.18). The RFT architecture had the highest accuracies among the multimodal MFN architectures and clearly outperformed the unimodal MFN and baseline accuracies of the first experiment.

The fourth experiment analyzes the Kalman filter for classifier fusion in the setting of the previous experiment. The unimodal Kalman filter fusion architecture had improved accuracies compared with the results of the unimodal MFN architectures (Table 5.16 and Table 5.20). Furthermore, the multimodal Kalman filter fusion architecture outperformed all three multimodal MFN architectures and, hence, turned out to be the best performing architecture of all experiments conducted on the AVEC 2011 dataset (Table 5.22).

The last experiment examines the Kalman filter for classifier fusion on the AVEC 2013 Dataset. Although a participation to the challenge was not possible, the multimodal classifier architecture was evaluated on the official test set. The experiment featured that the parameters of the Kalman filter were learned from data. The outputs of the multimodal Kalman filter fusion clearly outperformed the challenge winner and the results provided by Sánchez-Lozano et al. (2013) (Table 5.24).

The experiments showed that the MFN and Kalman filter for classifier fusion are both promising methods for multimodal temporal fusion which are further capable to reconstruct regions of missing classifier decisions. The Kalman filter for classifier fusion which was applied to both modalities showed the best performance among all examined fusion approaches.

5.5. Unidirectional Layered Architectures

The last study of this chapter addresses the ULA for the recognition of complex patterns as proposed in Section 3.7.1. The BLA is currently analyzed and is going to be published in an alternative form.

The ULA provides a generic mechanism to recognize complex entities. However, the setting in which the architecture is applied has a strong impact on further design decisions details and, as a result, on the final performance. The architecture was studied using the OLAD which was presented in Section 4.1.4.1. The OLAD provides three layers and, furthermore, features an object classification problem. The outputs of the object recognition were used as additional inputs to support the predictions of the classes in the layers. All parameters and test results were derived using 10 × 10 cross-validation. In the following, design details and results of the object recognition and the three layers are presented.

5.5.1. Object Recognition Using Markov Fusion Network

The object recognition was realized using a multimodal MFN MCS. The schematic drawing of the architecture is depicted in Figure 5.15. Three different feature sets, i.e. MFCC, HOC and HOG, were extracted from the audio and video channels and passed to three individual classifiers which operated on frame level. The video-based object recognitions made use of a one-vs-one classifier system utilizing L2 soft-margin SVM with probabilistic outputs Platt (2000). The audio-based object recognition was restricted to the manipulation of the spoon, since the other objects made no perceptible sound when manipulated. The stirring of the spoon in the cup was recognized using an NBC. A probability of one was assigned in case the class “Spoon” was recognized. The classifier returned no probability distribution at all in cases of silence or the class “Spoon” was not recognized. The three streams of probability distributions were combined using the MFN operating in offline processing mode.

Figure 5.16 provides an example of the inputs and outputs of the MFN MCS. The examined video recording shows the subject performing a series of actions involving multiple objects. The actions are interrupted by pauses in which the empty hand rests on the table. The upper three plots show the output of the video and audio classifiers, whereas the fourth

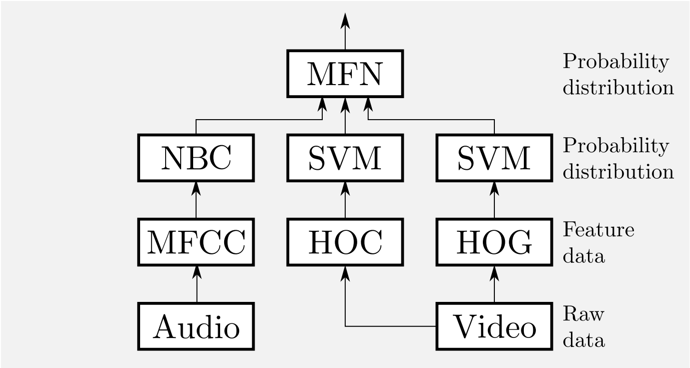


Figure 5.15. Multimodal multi-class object recognition MCS using the MFN on the Office Layered Architecture Dataset. MFCC, HOC and HOG feature vectors were extracted from the raw audio and video data. These features were utilized for classification using an NBC and two SVM. The derived probability distributions over objects were combined by an MFN.

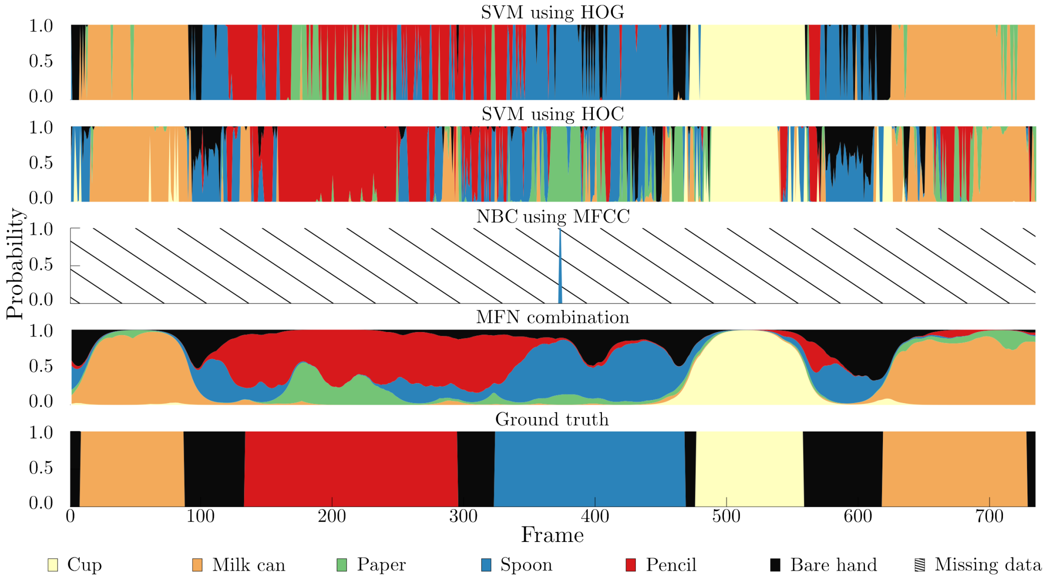


Figure 5.16. Probability distributions in different stages of the MFN MCS performing the object recognition in the Office Layered Architecture Dataset. Each plot shows the change of the probability distributions over time of the three individual classifiers, i.e. the SVM using HOG, the SVM using HOC and the NBC using MFCC, and the MFN combination. The plot at the bottom shows the annotated ground truth. The probability distributions sum up to one for each frame. The color encodes the share assigned to a class. For a detailed description please refer to the text.

plot shows the output of the MFN combination. The last plot shows the annotated ground truth. The plots depict the frame-wise share of probabilities distributed allocated to the objects. Each color represents a class. The color/object assignment can be inferred from the legend of the figure. For instance, the MFN output distribution in the fourth plot at frame 200 allocated the highest probability mass of approximately 80% to the class “Pencil”. The second highest probability of about 20% was assigned to the class “Paper”. All other objects had only marginal shares of probabilities close to zero. Frames in which no estimates were available are indicated by areas filled with diagonal lines.

The two uppermost plots show the estimates of the SVM utilizing HOG and HOC features. Both classifiers provided noisy probability distributions which, however, were often close to the ground truth. Furthermore, both classifiers occasionally misclassified small contiguous time periods, e.g. the class "Paper" in the proximity of frame 200. The third plot shows the predictions of the NBC using the audio features. Since in most frames the energy of the audio channel was below the noise threshold, no estimates were available except in the proximity of frame 385 in which the subject picks up the spoon and produces a sound by stirring the cup. The outputs of the MFN combination is depicted in the fourth plot. The algorithm combined the classifier decisions and simultaneously smoothed the estimates over time such that a large number of erroneous decisions were resolved. Apparently, the outputs provided by the MFN were closer to the annotated ground truth which is shown in the last plot.

The empirical evaluation is presented in the Table 5.26a and shows the accuracies and the F_1-measures of the three individual classifiers, four alternative fusion approaches and the MFN. The unimodal raw classier outputs achieved accuracies of 82.5%, 65.7% and 53.5% using the HOG, HOC and MFCC features, respectively. The F_1-measures confirm that the SVM classifiers using the HOG and HOC features provided balanced recognitions of all classes with the exception of the class "Hand". In the recordings, the subject rests his empty hand between two actions such that, due to the viewpoint of the camera, other objects may become visible in the extracted sub-images. These other objects were then wrongly recognized by the imaged-based classifiers. The accuracy of the NBC using the MFCC was completely based on the performance of the class "Spoon". Hence, only one F_1-measure related to the class "Spoon" was available which achieved 69.2%.

The performance of the MFN is shown in the last column. The MFN achieved an accuracy of 92.8% and, hence, outperformed the single classifiers. A similar finding can be made by examining the F_1-measures which all ranged above 90% except the class "Hand" having only an F_1-measure of 80.7%. For comparison, alternative fusion approaches were evaluated: the sum and product of all modalities for every time step, i.e. point-wise fusion ($\perp$), and the sum and product of all modalities using a sliding window with a size of an eighth of the complete sequence, i.e. windowed fusion (⎍). As a result, the utilized windows had an average length of 10.7 frames (10 frames median). The fusion was conducted by taking the sum, or respectively the product, of all available values. Subsequently, the quantities were normalized using the number of available values to obtain a probability distribution over classes. The accuracies are listed in Table 5.26a and outperformed the results of the single modalities. However, the MFN combination still outperformed the alternative fusion approaches. The point-wise sum fusion and the point-wise product fusion had accuracies of 88.2% and 92.4%, whereas the windowed sum fusion and the windowed product fusion

Table 5.25. Accuracies and $\mathbf{F_1}$-measures of the object recognition obtained by the classifiers, the point wise ($\perp$) and windowed ($\sqcap$) product $\prod$ and average $\sum$ fusion and the MFN on the Office Layered Architecture Dataset. (a) Performance of crisp classifier decisions. (b) Average rank accuracy. The rank denotes the ordered positions achieved in the probability distribution. All values in percent averaged with standard deviation. Arrows indicate how to rate the measures.

(a)

	HOG (video)	HOC (video)	MFCC (audio)	$\sum_{\perp}$ (fusion)	$\sum_{\sqcap}$ (fusion)	$\prod_{\perp}$ (fusion)	$\prod_{\sqcap}$ (fusion)	MFN (fusion)
↑Acc.	82.5±2.9	65.7±2.7	53.5±10.1	82.2±2.5	92.4±1.5	84.2±3.0	84.0±4.7	**92.8±2.5**
↑F_1(Cup)	92.2±3.0	69.5±2.5	n/a	87.9±2.9	95.9±2.4	92.9±2.9	66.9±6.9	95.8±2.9
↑F_1(Can)	83.2±5.3	60.7±4.4	n/a	81.5±4.8	92.4±3.7	85.2±5.1	85.6±5.1	94.7±2.7
↑F_1(Paper)	84.6±4.5	57.2±3.7	n/a	83.5±4.4	91.7±1.9	85.3±4.3	88.5±2.8	93.4±3.8
↑F_1(Spoon)	83.7±2.5	65.3±4.0	69.2±8.5	82.7±2.4	93.9±1.7	85.6±3.1	90.1±4.0	94.2±2.5
↑F_1(Pencil)	82.4±5.1	76.7±4.3	n/a	85.0±4.4	95.7±2.3	86.3±5.1	87.2±9.1	93.7±4.5
↑F_1(Hand)	64.6±5.5	47.1±2.9	n/a	64.6±4.7	73.1±4.9	63.6±5.6	70.0±5.6	80.7±6.2

(b)

	HOG (video)	HOC (video)	MFCC (audio)	$\sum_{\perp}$ (fusion)	$\sum_{\sqcap}$ (fusion)	$\prod_{\perp}$ (fusion)	$\prod_{\sqcap}$ (fusion)	MFN (fusion)
↑Acc.(rank 1)	82.5±2.9	65.7±2.7	53.5±10.1	82.1±2.5	92.4±1.5	84.2±3.0	92.4±1.5	92.8±2.5
↑Acc.(rank 2)	10.7±1.1	17.2±1.3	46.5±10.1	9.9±1.1	4.1±0.8	9.0±1.2	4.1±0.8	5.5±1.1
↑Acc.(rank 3)	3.7±1.0	7.2±0.9	n/a	4.0±0.6	2.1±0.5	3.8±0.7	2.1±0.5	1.4±1.3
↓Acc.(rank 4)	1.6±0.8	4.3±0.5	n/a	2.0±0.5	0.8±0.4	1.5±0.8	0.8±0.4	0.3±0.3
↓Acc.(rank 5)	0.8±0.5	3.3±0.7	n/a	1.2±0.4	0.3±0.3	0.7±0.4	0.3±0.3	0.0±0.1
↓Acc.(rank 6)	0.8±1.1	2.3±0.7	n/a	0.8±0.4	0.2±0.3	0.9±1.2	0.2±0.3	0.0±0.1

achieved accuracies of 84.2% and 84.0%. Hence, the windowed sum fusion was the only approach with an accuracy which came close to the accuracy to the MFN.

So far, the evaluation was based on a binarized rating of probability distributions. To take account of the probability distributions provided by the algorithms, rankings of the object probabilities were performed. The results of these rankings are shown in Table 5.26b. The accuracy concerning the first rank corresponds to the conventional accuracy. The second rank denotes the share of frames in which the actual object was rated to be the second most likely object and so on. The rankings of the SVM classifiers operating on HOG and HOC features show that a large share of objects to be recognized obtained only the second highest probability. The first two ranks of both SVM sum up to a share of 93.2% and 82.9%. Again the decisions based on the MFCC are difficult to rate, since they are only present in a small portion of frames and assign their probabilities to only one class. The added first two ranks of the classic fusion approaches achieved 92.0%, 96.5%, 93.2% and 96.4%, i.e. point-wise sum, windowed sum, point-wise product and windowed product fusion. The MFN provided the correct class among the first two highest probabilities with a chance of 98.3% which was clearly the highest share.

The outputs of the MFN MCS object recognition were used as additional inputs to the HMM and CHMM in the first two layers.

Table 5.27. Accuracies and F$_1$-measures of the Markov models trained using the raw features and the classes of the first layer on the OLAD. The Number preceding the stars coding the significance related to (1) HMM, (2) CHMM or (3) FCHMM. All values averaged with standard deviation. Arrows indicate how to rate the measures. For a detailed description please refer to the text.

Layer 1: Actions — Input: Raw features

Method		HMM	CHMM	FCHMM
	↑Acc.	58.8±3.4 %	62.1±2.1 %^{1**}	**64.3**±**2.6** %^{1*}
		↑F$_1$	↑F$_1$	↑F$_1$
Pick up object	(PUP$_1$)	50.0±7.4	58.3±7.3 ^{1*}	64.2±3.3 $^{1**}_{2*}$
Move object towards head	(MTH$_1$)	84.0±7.7	92.5±4.2 ^{1**}	89.9±5.5 ^{1*}
Move object from head	(MFH$_1$)	49.8±6.0	46.1±6.4	47.6±8.8
Move object towards table	(MTT$_1$)	43.3±5.8	45.1±5.8	45.8±7.0
Move object from table	(MFT$_1$)	81.5±5.9	91.0±4.2 ^{1**}	90.3±4.5 ^{1**}
Lay back object	(LBA$_1$)	51.7±5.8	55.5±6.3	64.0±6.9 $^{1**}_{2*}$
Manipulate object in hand	(HAM$_1$)	97.6±3.2	97.7±3.1	96.2±4.4
Manipulate object at table	(TAM$_1$)	44.3±6.8	43.0±6.0	41.5±9.3
Manipulate object close to head	(HEM$_1$)	48.6±6.6	65.0±7.9 ^{1**}	62.3±15.6 ^{1*}
Empty hand	(EH$_1$)	69.4±8.4	69.6±7.1	73.4±5.8

5.5.2. Action Recognition

The lowermost layer has to recognize the action classes. Three sequential classifiers were evaluated, namely the classic HMM and the two HMM extensions presented in this thesis, i.e. CHMM and the FCHMM. The CHMM and FCHMM made use of the outputs provided by object recognition which were provided to the Markov models using the causal sequences. Cross-validation was used to determine the optimal configurations of the models. All three Markov models were set to have five hidden states and GMM observations with full covariances. The HMM utilized a left-to-right transition matrix and a GMM with five mixture components, whereas the CHMM and FCHMM made use of a full transition matrix with a GMM having only three mixture components. As already mentioned, the FCHMM is sensitive to the degree of fuzziness in the causal sequences. Therefore, the additional parameter p which controls the fuzziness of the object recognition input was optimized and set to $p = 30$.

The accuracies and F$_1$-measures of all three models in the first layer are shown in Table 5.27. The upper part of the table shows the accuracies of the HMM, CHMM and FCHMM, whereas the lower part of the table lists the F$_1$-measures of the classes. The HMM achieved an accuracy of 58.8% and, therefore, was outperformed by the CHMM and the FCHMM which achieved accuracies of 62.1% 64.3%. Although the FCHMM had the highest accuracy, the CHMM had a higher level of significance ($*\ p < 0.05$ and $**\ p < 0.01$; paired t-test) with respect to the HMM. The F$_1$-measure of the CHMM and FCHMM significantly outperformed the HMM in several classes such as PUP$_1$, MTH$_1$, MFT$_1$ and HEM$_1$.

The second layer was trained using the outputs of the first layer. Hence, the outputs of each test set fold was collected and joint such that a new complete dataset was compiled.

Table 5.28. Accuracies and F_1-measures of the Markov models trained using the output of the first layer and the classes of the second layer on the Office Layered Architecture Dataset. The Number preceding the stars coding the significance related to (1) HMM, (2) CHMM or (3) FCHMM. All values averaged with standard deviation. Arrows indicate how to rate the measures. For a detailed description please refer to the text.

(a)

Layer 2: Activities		Input: layer 1 HMM		
Method		HMM	CHMM	FCHMM
	↑Acc.	71.1±5.7 %	**80.8±5.3** %^{1**}	77.4±6.5 %^{1**}
		↑F_1^{HMM}	↑F_1^{CHMM}	↑F_1^{FCHMM}
Drink from cup	(DC_2)	85.6±8.2	98.2±3.8 ^{1**}	96.0±6.7 ^{1*}
Fill milk into a cup	(MC_2)	61.5±17.6	87.9±8.3 ^{1**}	81.1±14.5 ^{1*}
Read note	(RN_2)	80.3±9.9	88.4±6.9	89.3±9.5
Stir cup	(SC_2)	39.1±19.8	56.3±12.7	45.9±21.1
Add sugar and stir	(SSC_2)	64.3±17.5	65.3±19.2	64.5±15.9
Write note	(WN_2)	56.5±18.1	81.7±11.3 ^{1**}	74.9±12.2 ^{1**}
No activity	(HEM_2)	81.9±7.0	84.1±5.0	81.3±7.8

(b)

Layer 2: Activities		Input: layer 1 CHMM		
Method		HMM	CHMM	FCHMM
	↑Acc.	78.0±3.5 %	**82.3±5.4** %^{1*}	81.8±4.7 %^{1*}
		↑F_1^{HMM}	↑F_1^{CHMM}	↑F_1^{FCHMM}
Drink from cup	(DC_2)	97.4±4.2	96.3±6.2	98.3±3.6
Fill milk into a cup	(MC_2)	89.0±7.6	93.4±7.7	95.4±7.8
Read note	(RN_2)	80.5±11.5	82.9±9.8	91.3±7.3 $^{1**}_{2*}$
Stir cup	(SC_2)	48.3±23.1	47.0±19.1	55.7±22.6
Add sugar and stir	(SSC_2)	59.7±18.9	67.7±19.4	66.0±17.4
Write note	(WN_2)	79.2±12.8	87.2±9.4 ^{3**}	77.6±7.1
No activity	(HEM_2)	82.5±7.2	87.4±6.9	85.5±4.1

(c)

Layer 2: Activities		Input: layer 1 FCHMM		
Method		HMM	CHMM	FCHMM
	↑Acc.	78.2±6.3	81.2±6.4	**81.8±5.3** %
		↑F_1^{HMM}	↑F_1^{CHMM}	↑F_1^{FCHMM}
Drink from cup	(DC_2)	94.8±7.1	97.1±4.7	92.3±9.2
Fill milk into a cup	(MC_2)	70.5±19.4	94.0±5.3 ^{1**}	93.4±8.4 ^{1**}
Read note	(RN_2)	82.7±14.9	83.4±13.8	88.0±9.3
Stir cup	(SC_2)	49.0±21.7	46.9±19.0	57.0±13.7
Add sugar and stir	(SSC_2)	74.7±8.3 ^{2*}	62.8±11.8	73.4±9.5 ^{2*}
Write note	(WN_2)	68.9±16.3	79.8±9.1 ^{1*}	80.3±12.3 ^{1*}
No activity	(HEM_2)	85.4±4.3	87.8±6.8	84.3±5.8

Table 5.30. Accuracies and F_1-measures of the Markov models trained using the raw features and the classes of the second layer on the Office Layered Architecture Dataset. The Number preceding the stars coding the significance related to (1) HMM, (2) CHMM or (3) FCHMM. All values averaged with standard deviation. Arrows indicate how to rate the measures. For a detailed description please refer to the text.

Layer 2: Activities		Input: Raw features		
Method		HMM	CHMM	FCHMM
	↑Acc.	85.9±5.1 %	88.2±3.6 %	**89.4±4.4** %^{1**}
		↑F_1^{HMM}	↑F_1^{CHMM}	↑F_1^{FCHMM}
Drink from cup	(DC_2)	93.7±6.4	99.2±2.4 ^{1*}	99.1±2.9 ^{1*}
Fill milk into a cup	(MC_2)	85.4±13.0	90.1±8.7	95.5±6.2
Read note	(RN_2)	85.9±10.7	85.9±10.8	90.6±5.2
Stir cup	(SC_2)	71.9±13.3	77.5±11.8	77.4±16.0
Add sugar and stir	(SSC_2)	83.1±9.0	84.7±7.1	85.9±7.3
Write note	(WN_2)	81.3±11.3	87.6±11.9	84.2±15.2
No activity	(HEM_2)	89.6±3.9	89.7±3.1	90.8±4.7

5.5.3. Activity Recognition

The second layer has to recognize the activity classes based on the observed sequences of actions of the previous layer. The activity recognition was studied using the HMM, CHMM and FCHMM. The accuracies and F_1-measures achieved by the classifiers operating in the second layer are shown Table 5.28. Table 5.29c, Table 5.29b and 5.29a provide the performance measures given that the HMM, CHMM and FCHMM was used to provide the inputs from the first layer, respectively. The utilized parameters of the models on the second layer are listed in Table A.5a in the Appendix.

Table 5.29c shows the results in case the HMM outputs of the first layer were used to recognize the classes of the second layer. In this setting, the CHMM achieved the highest accuracy of 80.8%, followed by the FCHMM and HMM with 77.4% and 71.1%, respectively. The Table 5.29b shows the results in case the inputs to the second layer were provided by the CHMM in the first layer. This configuration led to an HMM accuracy of 78.0% followed by CHMM and FCHMM accuracies of 82.3% and 81.8%, respectively. The results clearly outperformed the results presented in the previous Table 5.29c. F_1-measures show that the FCHMM was able to recognize the classes "Stir cup" and "Add sugar and stir" more robustly. The results of the last configuration in which the inputs were provided by the FCHMM operating in the first layer are provided in Table 5.29a. The accuracies in this setting are 78.2% for the HMM, 81.2% for the CHMM and 81.8% for the FCHMM. Hence, the FCHMM was the approach with the highest performance in this setting. However, the accuracy of the CHMM in the first and the second layer which is presented in Table 5.29b was still higher.

The experiments on the second layer showed that the HMM was outperformed by the CHMM and the FCHMM. Furthermore, the F_1-measures revealed that the classes "Stir cup" and "Add sugar and stir" were often confused. This is of importance, since these classes are crucial in order to perform a successful recognition of user preferences.

At this point, it is legitimate to inquire the performance of an alternative approach. Therefore, the next experiment addressed models which were trained directly on the raw features. Table 5.30 shows the results when applying all three models directly to the raw features. The utilized parameters are listed in Table A.5b in the Appendix. The performance measures were clearly higher compared with the layered approach: The HMM achieved an accuracy of 85.9%, the CHMM of 88.2% and the FCHMM achieved even 89.4%. As already observed in the previous Section 5.5.2 and also discussed in Scherer et al. (2012b), the recognition performance of pre-segmented and unsegmented datasets can differ significantly. Up to this point, the directly trained models promised to be the optimal choice and to be preferred over the ULA approach. However, there is still one more layer to be analyzed. The final models of the second layer were trained on the complete segmented dataset, since the test results in the next section based on a separate dataset.

Table 5.31. Accuracies and F_1-measures of the classes of the second layer at the third layer using the directly trained Markov models on the Office Layered Architecture Dataset. The Number preceding the stars coding the significance related to (1) HMM, (2) CHMM or (3) FCHMM. All values averaged with standard deviation. Arrows indicate how to rate the measures. For a detailed description please refer to the text.

Layer 3: Activities Input: Raw features

Method		HMM	CHMM	FCHMM
Layer 3: Activities	↑Acc.	$49.6\pm15.1\,\%$	$\mathbf{65.4\pm13.9}\,\%^{1**}_{3**}$	$57.6\pm15.4\,\%^{1**}$
		$\uparrow\mathrm{F}_1^{\mathrm{HMM}}$	$\uparrow\mathrm{F}_1^{\mathrm{CHMM}}$	$\uparrow\mathrm{F}_1^{\mathrm{FCHMM}}$
Drink from cup	(DC_2)	75.5 ± 25.4	76.3 ± 11.9	73.5 ± 21.0
Fill milk into a cup	(MC_2)	21.5 ± 28.0	$35.7\pm36.6^{1**}$	30.2 ± 31.6
Read note	(RN_2)	40.0 ± 33.5	$75.9\pm27.5^{1**}$	47.4 ± 29.0
Stir cup	(SC_2)	9.2 ± 10.2	12.0 ± 15.0	20.6 ± 15.7
Add sugar and stir	(SSC_2)	24.3 ± 25.2	$28.8\pm30.2^{1**}$	25.4 ± 26.3
Write note	(WN_2)	53.6 ± 30.6	$62.9\pm30.4^{1**}$	58.8 ± 31.3
No activity	(HEM_2)	2.6 ± 5.3	$4.3\pm4.9^{1*}$	4.0 ± 5.3

Table 5.32. Accuracies and F_1-measures of the classes of the second layer at the third layer using the layered architecture on the Office Layered Architecture Dataset. The Number preceding the stars coding the significance related to (1) HMM, (2) CHMM or (3) FCHMM. All values averaged with standard deviation. Arrows indicate how to rate the measures. For a detailed description please refer to the text.

(a)

Layer 3: Activities Input: layer 1 HMM

Method		HMM	CHMM	FCHMM
Layer 3: Activities	↑Acc.	$40.9\pm10.3\,\%$	$\mathbf{65.4\pm14.4}\,\%^{1**}_{3**}$	$57.1\pm11.2\,\%^{1**}$
		$\uparrow\mathrm{F}_1^{\mathrm{HMM}}$	$\uparrow\mathrm{F}_1^{\mathrm{CHMM}}$	$\uparrow\mathrm{F}_1^{\mathrm{FCHMM}}$
Drink from cup	(DC_2)	69.9 ± 15.6	$83.0\pm11.9^{1**}$	$86.6\pm7.4^{1**}_{2**}$
Fill milk into a cup	(MC_2)	19.5 ± 22.5	$39.6\pm40.6^{1**}_{3**}$	$33.9\pm37.2^{1**}$
Read note	(RN_2)	45.8 ± 26.2	$63.3\pm28.0^{1**}$	$63.5\pm28.3^{1**}$
Stir cup	(SC_2)	16.5 ± 14.0	$36.7\pm23.9^{1**}_{3**}$	$28.4\pm20.8^{1**}$
Add sugar and stir	(SSC_2)	15.6 ± 25.0	14.4 ± 14.8	$30.4\pm31.9^{1**}_{2**}$
Write note	(WN_2)	33.0 ± 23.8	$62.6\pm32.3^{1**}_{3**}$	$47.0\pm27.4^{1**}$
No activity	(HEM_2)	5.7 ± 11.9	5.1 ± 7.3	5.1 ± 5.7

(b)

Layer 3: Activities Input: layer 1 CHMM

Method		HMM	CHMM	FCHMM
Layer 3: Activities	↑Acc.	$53.6\pm11.4\,\%$	$62.3\pm11.3\,\%^{1**}$	$\mathbf{64.8\pm14.5}\,\%^{1**}_{2**}$
		$\uparrow\mathrm{F}_1^{\mathrm{HMM}}$	$\uparrow\mathrm{F}_1^{\mathrm{CHMM}}$	$\uparrow\mathrm{F}_1^{\mathrm{FCHMM}}$
Drink from cup	(DC_2)	$85.5\pm11.0^{2**}$	78.8 ± 9.9	$90.1\pm7.7^{1**}_{2**}$
Fill milk into a cup	(MC_2)	30.4 ± 33.5	$40.0\pm41.6^{1**}$	$41.7\pm43.8^{1**}$
Read note	(RN_2)	67.1 ± 27.9	66.1 ± 25.4	$77.0\pm27.5^{1**}_{2**}$
Stir cup	(SC_2)	$29.3\pm19.0^{3**}$	$34.0\pm23.1^{1**}_{3**}$	24.5 ± 20.7
Add sugar and stir	(SSC_2)	$11.3\pm11.7^{3**}$	$18.7\pm19.9^{1**}_{3**}$	7.3 ± 8.2
Write note	(WN_2)	40.7 ± 24.6	$58.5\pm29.0^{1**}$	$57.9\pm30.3^{1**}$
No activity	(HEM_2)	5.7 ± 9.7	$8.0\pm8.2^{3**}$	3.5 ± 7.4

(c)

Layer 3: Activities Input: layer 1 FCHMM

Method		HMM	CHMM	FCHMM
Layer 3: Activities	↑Acc.	$53.3\pm15.1\,\%$	$67.1\pm14.1\,\%^{1**}$	$\mathbf{72.1\pm14.8}\,\%^{1**}_{2**}$
		$\uparrow\mathrm{F}_1^{\mathrm{HMM}}$	$\uparrow\mathrm{F}_1^{\mathrm{CHMM}}$	$\uparrow\mathrm{F}_1^{\mathrm{FCHMM}}$
Drink from cup	(DC_2)	75.9 ± 15.8	$88.4\pm11.0^{1**}$	$92.4\pm7.3^{1**}$
Fill milk into a cup	(MC_2)	28.5 ± 30.9	$38.2\pm40.1^{1**}$	$42.6\pm43.5^{1**}$
Read note	(RN_2)	67.1 ± 27.8	$74.7\pm25.9^{1**}$	$77.3\pm27.9^{1**}$
Stir cup	(SC_2)	14.2 ± 13.4	$23.7\pm19.8^{1**}$	$35.0\pm23.3^{1**}$
Add sugar and stir	(SSC_2)	10.9 ± 18.7	$19.3\pm19.8^{1**}$	$13.7\pm15.0^{1*}$
Write note	(WN_2)	42.7 ± 28.3	$61.6\pm30.7^{1**}$	$64.2\pm31.9^{1**}$
No activity	(HEM_2)	2.6 ± 5.8	$6.0\pm6.3^{1**}$	$5.3\pm8.2^{1*}$

5.5.4. User Preference Recognition

The third layer has to recognize the preferences of the user. Within this study, this was accomplished by using a DMLN which operated on a sequence of activities. Two experiments are provided in this section: (1) the models of the second layer and the directly applied models performing unsegmented activity recognition and (2) the performance of the DMLN.

The performance of the activity recognition using the unsegmented dataset of the third layer is studied in the first experiment. The accuracies and F_1-measures of the directly applied models are provided in Table 5.31. Table 5.33a, Table 5.33b and Table 5.33c show the accuracies and F_1-measures of the ULA using the HMM, CHMM and FCHMM in the first layer, respectively. Given the directly applied models, the CHMM achieved the highest accuracy of 65.4%, followed by the FCHMM and HMM which achieved accuracies of 57.6% and 49.5%, respectively. However, the results of the ULA clearly outperformed the direct application of the models. Utilizing the HMM in the first layer achieved the lowest accuracies in the evaluation with 40.9%, 65.4% and 57.1% for the HMM, CHMM and FCHMM in the second layer, respectively. Using the CHMM in the first layer achieved higher accuracies of 53.6%, 62.3% and 64.8% for the HMM, CHMM and FCHMM on the second layer. Table 5.33c provides the performance measures using the FCHMM in the lowermost layer. The accuracies of the HMM, CHMM and FCHMM on the second layer yielded accuracies of 53.3%, 67.1% and 72.1%. Using the FCHMM in the first and the second layer provided significantly higher accuracies and F_1-measures than using the FCHMM in the first layer and the HMM in the second layer. According to the F_1-measures, the class “Stir cup” performed slightly better using the ULA. However, the class “Add sugar and stir” which is crucial for the recognition of the user preferences “White coffee” and “Sweet white coffee” had a critically low performance. Generally speaking, the F_1-measures of the class “No activity” were consistently low in Table 5.31 and Table 5.32. This is mostly because it was cross-talked by activities before and after the target class. Fortunately, this class carries no information for the third layer and is merely responsible to separate the classes. Furthermore, it is worth noting that the standard deviations of the accuracies and F_1-measures were comparably high. Nonetheless, the performances of the models were very good considering that an unsegmented sequential classification problem with seven classes was solved. In the following, the FCHMM was used in the first and the second layer for further processing.

The second experiment examines the performance of the user preference recognition in the third layer which was realized using a DMLN. The model of the DMLN utilizes one static and four temporal random variables. The static (time-independent) random variable represents the coffee preference of the user (`class`). The time-dependent variables are the currently performed activity (`perf`), the activity that ended during the last time-step or null (`act`), the state of the coffee inside the cup (`cof`), and whether the cup is currently stirred (`stir`). Information from the second layer was integrated as soft observations on the `perf` variables. The DMLN model ran at a lower temporal resolution of 2 Hz compared to the underlying layers. Hence, only every 10th output of the second layer was included.

Table 5.34. Probabilistic logical rules of the DMLN model used in the Office Layered Architecture Dataset. Signature of predicates `perf(Time,Activity!)`, `class(Class!)`, `act(Time,Activity!)`, `cof(Time,Class!)`, and `stir(Time)` where "!" denotes a functional argument. Every formula is prefixed by a weight, where ∞ means a formula must hold with certainty for every assignment to its free variables.

Weight	First-order logic	ID
w	$\texttt{perf}(t,a) \wedge \texttt{perf}(t+1,a)$	F_1
∞	$(\texttt{perf}(t,a_1) \wedge \texttt{perf}(t+1,a_2) \Rightarrow \texttt{act}(t,a_1)) \wedge a_1 \neq a_2$	F_2
∞	$(\exists a : \texttt{perf}(t,a) \wedge \texttt{perf}(t+1,a)) \Rightarrow \texttt{act}(t,\mathrm{NULL})$	F_3
∞	$\texttt{perf}(t,\mathrm{DC}_2) \Rightarrow \exists c : \texttt{cof}(t,c) \wedge \texttt{class}(c)$	F_4
∞	$\texttt{cof}(0,\mathrm{BC}_3)$	F_5
∞	$\texttt{cof}(t,c) \wedge \neg(\texttt{act}(t,\mathrm{MC}_2) \vee \texttt{act}(t,\mathrm{SSC}_2)) \Rightarrow \texttt{cof}(t+1.c)$	F_6
∞	$\texttt{act}(t,\mathrm{MC}_2) \Leftrightarrow (\texttt{cof}(t,\mathrm{BC}_3) \wedge \texttt{cof}(t+1,\mathrm{MC}_3)) \vee (\texttt{cof}(t,\mathrm{SC}_3) \wedge \texttt{cof}(t+1,\mathrm{MCS}_3))$	F_7
∞	$\texttt{act}(t,\mathrm{SSC}_2) \Leftrightarrow (\texttt{cof}(t,\mathrm{BC}_3) \wedge \texttt{cof}(t+1,\mathrm{SC}_3)) \vee (\texttt{cof}(t,\mathrm{MC}_3) \wedge \texttt{cof}(t+1,\mathrm{MCS}_3))$	F_8
∞	$\texttt{cof}(t,\mathrm{BC}_3) \Rightarrow \texttt{stir}(t)$	F_9
∞	$\texttt{stir}(t) \Rightarrow \neg\texttt{act}(t,\mathrm{SC}_2)$	F_{10}
∞	$\texttt{act}(t,\mathrm{MC}_2) \vee \texttt{act}(t,\mathrm{SSC}_2) \Rightarrow \neg\texttt{stir}(t+1)$	F_{11}
∞	$\texttt{act}(t,\mathrm{SC}_2) \Rightarrow \texttt{stir}(t+1)$	F_{12}
∞	$\neg(\texttt{act}(t,\mathrm{MC}_2) \vee \texttt{act}(t,\mathrm{SSC}_2) \vee \texttt{act}(t,\mathrm{CS}_2)) \Rightarrow (\texttt{stir}(t) \Leftrightarrow \texttt{stir}(t+1))$	F_{13}
∞	$\texttt{perf}(t,\mathrm{DC}_2) \Rightarrow \texttt{stir}(t)$	F_{14}

Table 5.35. Accuracies and F_1-measures of the DMLN trained using the output of the second layer and the classes of the third layer on the Office Layered Architecture Dataset. The results are based on the outputs of the second layer. The outputs of the first and the second layer were obtained using an FCHMM. All values averaged with standard deviation. Arrows indicate how to rate the measures. For a detailed description please refer to the text.

Layer 3: User preferences — Input: layer 1/2 FCHMM/FCHMM

Method		DMLN
	↑Acc.	**90.0±9.9** %
		↑F_1^{DMLN}
Black coffee	(BC_3)	98.0±6.3
White coffee	(WC_3)	81.3±16.6
Sweet coffee	(SC_3)	100.0±0.0
Sweet white coffee	(SWC_3)	90.8±13.3

The probabilistic logical rules of the DMLN are specified in Table 5.34. All formulas in the DMLN model except Formula F_1 are deterministic. Deterministic formulas must hold with certainty and, therefore, are weighted with an infinite value. The model has been designed according to our understanding of the process in the dataset. Some rules encode objective mechanics of the domain. For example pouring milk into black coffee produces "White coffee" (Formula F_7), or pouring something into the coffee "Unstirs" it (Formula F_{11}) while "Stirring" makes the coffee stirred again (Formula F_{12}). Other rules represent assumptions, e.g. someone will not stir the coffee if it is already stirred (no explicit formula). Others are technical details, such as the rule asserting that the coffee is black at the beginning of the recording (Formula F_5). The Formula F_1 encodes the probability that a performed activity will be also performed in the next time step. The weight of this rule describes how likely an activity changes. Frequent activity changes are less probable with higher weights. The weight of the Formula F_1 was learned using maximum likelihood estimation (MLE) with the ground truth user preference as evidence. Averaging the weight over all folds of the validation set yielded in the value of 3.99±0.08. The fitted models provided the same predictions on all instances, simply because most of the information is encoded by the deterministic rules indicating the the exact value of the weighting parameter is of minor relevance.

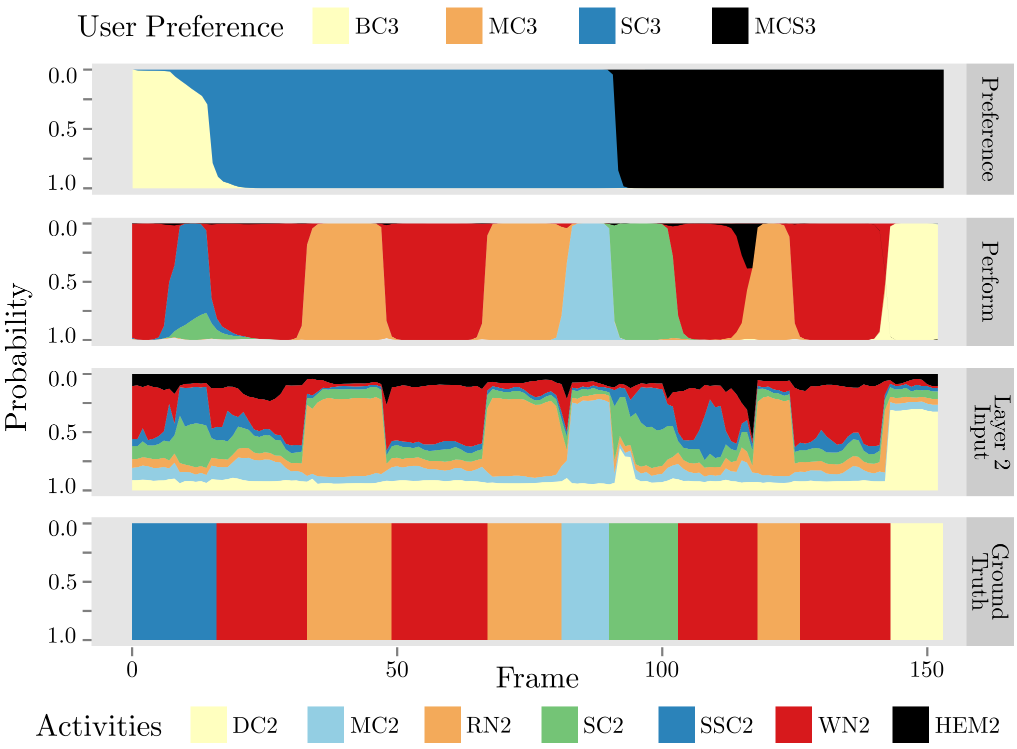


Figure 5.17. Probability distributions in different parts of the DMLN model applied in third layer of the Office Layered Architecture Dataset. From top to bottom: The marginal probabilities of the state of the coffee cup estimated in the third layer, the currently performed activity estimated in the third layer, the currently performed activity estimated in the second layer (input to the third layer), and at the bottom the true current activity. One time step corresponds to 500 ms.

The accuracies and F_1-measures of the DMLN are shown in Table 5.35. The DMLN achieved an accuracy of 90.0% with 9.9% standard deviation. Considering that the DMLN had integrated information of about one minute using the inputs provided by the second layer with an accuracy of 72.1%, the final performance is more than satisfactory. A large number of class confusions occurring in the previous layer was resolved by the probabilistic rules.

Figure 5.17 shows the belief over time for a selection of latent variables in the DMLN model. The figure was generated using a human-assigned weighting of $w = 3$ which led to an accuracy of 96.3%. This accuracy was even superior to the performance of the weighting obtained using MLE. Given the rather noisy and uncertain observations from the second layer that served as input, it is notable how well the belief about the currently performed activity obtained from the DMLN model matches the true activity in the figure. Furthermore, it is interesting to observe how the confusion between adding sugar (SSC2) and stirring (SC2) was resolved in the interval of frame 10 and 20. This incidence is related to the rule that stirring will not happen if the coffee is already stirred (and black coffee is always stirred). Further information regarding the utilized DMLN can found in Glodek et al. (2014a).

5.5.5. Summary

A large study using the OLAD was performed in order to examine the ULA. The OLAD is a dataset which is composed of three layers of classes building upon each other. The first layer comprises action classes which compose the activity classes of the second layer. The classes of the second layer in turn compose the user preference classes of the third layer. An additional object recognition can be utilized to support the classifiers of the first two layers. The HMM, CHMM and FCHMM was evaluated in the first two layers and the DMLN in the third layer. The CHMM and FCHMM made use of the crisp and fuzzy object recognition.

The object recognition was performed as a part of the study on the ULA. It was used to additionally evaluate the performance of the MFN. Three individual classifiers were trained based on features extracted from the audio and video channels. The classifier operating on the audio channel provided only evidence for one class, due to the properties of the objects to be recognized. In case the single target class was not recognized, no information was propagated to the MFN. Two video classifiers were trained using HOG and HOC features as inputs. The decisions of the classifiers were used to evaluate different fusion approaches, i.e. the MFN and four temporally extended multimodal fusion methods. The results showed that the MFN clearly outperformed the conventional fusion methods (Table 5.25). Analyzing the crisp combined outputs showed that only the windowed product fusion achieved an almost competitive performance. Further analysis on the fuzzy class memberships revealed that the MFN had higher chance to predict the correct class under the first two most likely classes than the windowed product fusion. According to this experiment, the class distributions obtained from the MFN is to be favored compared to the other fusion approaches.

The following experiments focused on the layers of the ULA. The results achieved on the first layer showed that the CHMM and FCHMM significantly outperformed the HMM (Table 5.27). The F_1-measures show that certain classes were recognized with lower reliability. However, the second layer which operates on the outputs of the first layer integrates information on a larger time scale and, hence, should be able to resolve these inconsistencies.

The performance of the second layer was evaluated for all possible Markov models in the first layer and all three models in the second layer, i.e. the HMM, CHMM and FCHMM (Table 5.28). The best out of these nine configurations was the CHMM in the first and the second layer. The second best configuration utilized the FCHMM in the second layer and the CHMM or FCHMM in the first layer. The performance of the ULA was validated by models being directly applied to the classes of the second layer. The results of this experiment clearly outperformed the layered approach (Table 5.30 and Table 5.28).

The predictions of the second layer are required to perform the user preference recognition. However, the sequence of activities corresponds to an unsegmented setting. Therefore, the directly trained models and the ULA were additionally evaluated on the third layer using the classes of the second layer. The experiment showed that, in this setting, the ULA clearly outperformed the direct trained models. The best result was achieved using the FCHMM in the first and the second layer (Table 5.32 and Table 5.31).

The evaluation of the third layer was performed using the FCHMM in the first and the second layer and the DMLN in the third layer. The DMLN implemented a set of probabilistic logical rules which modeled the transition between activities, the state of the coffee cup and the implication to the user preference class. The DMLN achieved a recognition rate of about 90% (Table 5.35). The experiment showed that the probabilistic logical rules can suppress wrong predictions from the preceding layers.

The study of the ULA confirmed that the layered architecture can recognize complex entities based on observations made from sensors using a unified framework. The experiment in which the Markov models were additionally applied directly to the classes of the second layer revealed that the benefits of the ULA becomes first perceptible and evident in the unsegmented setting of the next layer. Furthermore, it was shown that errors in the preceding layer can be compensated with the help of the logical rules.

Chapter 6

Discussion

The chapter discusses the results of the empirical evaluation presented in the preceding Chapter 5. It is thematically organized in the same manner as Chapter 3 which presents the methodical advances. First the proposed ensemble approach for density estimation, the EGMM, is discussed. Subsequently, the results achieved using the HMM-GPD and the CHMM-GPD are discussed. The next section reviews achievements of the novel MCS approaches, including the CHMM for late classifier fusion and the MFN and the Kalman filter for multimodal temporal classifier fusion. Finally, the ULA for complex pattern recognition is discussed which was empirically evaluated using the MFN, CHMM and FCHMM.

Each discussion first summarizes the goals of the study and then analyzes the results in order to identify advantages and problems of the approach. At the end of each discussion a preliminary conclusion is drawn. A complete list of abbreviations can be found in Section A.1 in the Appendix. Details on the utilized datasets are provided in the corresponding sections of Chapter 4

6.1. Ensemble GMM

The EGMM is an approach for robust density estimation which uses an ensemble of GMM. The study of the EGMM bases on two experiments. The first experiment provides a comparison to state-of-the-art approaches to assess the performance of the presented approach. The second experiment provides a deeper insight into the method's robustness by systematically evaluating the EGMM using three different beta distributions with the bias-variance decomposition.

The first experiment compares the proposed algorithm with the approaches studied by Ormoneit and Tresp (1998) who focused on density estimation algorithms using either superpositions of GMM or penalized GMM. The EGMM was evaluated using the Ormoneit Dataset I (ODI) and the Ormoneit Dataset II (ODII) which were proposed by Ormoneit and Tresp (1998). ODI is based on an sufficient number of two-dimensional samples, whereas ODII contains only a low quantity of ten-dimensional samples. The proposed algorithm clearly outperformed the alternative approaches on the ODI (Table 5.1, page 125). The EGMM based on the complete set of evaluated GMM achieved on average an accuracy of 84.55%, while the computationally less expensive pruned EGMM which used only the 25 GMM with the highest likelihoods for testing achieved an accuracy of 84.77%. In case of the ODII, the highest accuracy of 81.7% was achieved by the penalized likelihood approach (Table 5.1, page 125). However, the EGMM and the pruned EGMM reached the second and third place with accuracies of 78.8% and 77.4%, respectively. The penalized likelihood approach regularizes the GMM parameters by adding prior knowledge to the EM algorithm. Hence, it is difficult to assess the comparability of this approach. However, the EGMM which does not use additional knowledge yielded good results. The experiment showed that the EGMM is a good choice for accurate density estimation while having a reasonable computational effort. The selection of the 25 GMM to set up the pruned EGMM does not guarantee to obtain an improved accuracy. However, limiting the number of ensemble members can constrain the model's complexity and, therefore, can be used to adjust the computational effort when testing new data samples.

The second experiment examines the Beta Distribution Dataset (BDD) which comprise three different beta distributions. The bias-variance decomposition was utilized to compare the EGMM to conventional GMM. For most of the evaluated beta distributions, a GMM with a small number of mixture components will provide the best approximation. However, non-Gaussian distributions can only be represented by a GMM having an infinite number of mixture components. Hence, the experiment simulates the search in a large parameter space in which the target distribution cannot be described by the actual model. Therefore, this setting is well suited to examine the robustness of the GMM and EGMM. The evaluation of the classic GMM showed that the risk of choosing a non-optimal model increases the less the distribution resembles a Gaussian. In contrast to that, the EGMM can provide stable estimates (Figure 5.2). Hence, the risk of choosing an unqualified model is considerably lower with the EGMM than with the classic GMM.

The experiments show that the EGMM provides more accurate estimates to non-Gaussian densities compared with classic GMM or GMM bagging approaches. Furthermore, the EGMM provides more robust densities than the GMM with regard to non-Gaussian distributions.

6.2. HMM and CHMM Using Graph Probability Densities

The HMM-GPD approach integrates the GPD proposed by Trentin and Di Iorio (2008) into the probabilistic HMM such that not only sequences of variable length, but also variable-sized feature vectors provided as graphs, can be modeled.

The study of the HMM-GPD examines the performance and flexibility of the approach using two action recognition datasets, namely the Occlusion Action Dataset (OAD) and the UUlm Multi-perspective Action Dataset (UUlmMAD). The first experiment which uses OAD provides an intuition of the HMM-GPD to handle sequences of features with variable length. The second experiment compares the HMM-GPD and CHMM-GPD with a state-of-the-art approach for action recognition and utilizes the UUlmMAD.

The experiment on the OAD shows that the recognition of actions can be performed despite feature vectors of variable-length which originate from simulated occlusions. The experiment is composed of two parts. In the first part, a set of HMM-GPD was trained using exclusively complete graph data. The testing of the derived models were then performed on sets with different degrees of occlusion configurations, i.e. graph data with different degrees of incompleteness. Testing on the complete graph data resulted in the best performance. The performance measures showed only a slight decrease of the accuracies and F_1-measures for test sets with increasing occlusions. In case of the heaviest occlusion, the accuracy dropped around 10% (upper part of Table 5.2, page 129). It is important to note that it is not self-evident that testing can be performed on data samples which are composed only of a fraction of what was used to carry out the training of the classifier. In the second part of the experiment, a set of HMM-GPD was trained using complete and incomplete graph data. Results were again derived by testing on sets with different degrees of occlusion configurations. The highest accuracy was obtained when testing on the complete graph dataset. Once more the performance decreased only slightly in case the testing was performed on incomplete graph data (lower part of Table 5.2, page 129). The results showed that the approach is able to perform training and testing on incomplete data. This property is of a particular advantage for the creation of new datasets in real-world scenarios.

The second experiment examines the HMM-GPD and the CHMM-GPD in a more complex scenario using the UUlmMAD. The dataset features 14 actions performed by 31 subjects and was recorded using three video cameras and a motion capturing system. The dataset is composed of sequences of graphs in which the number of edges can change every time step. Graph sequences were generated with the help of occlusion-free skeleton models recorded by the motion capturing system. A post-processing step then generated the self-occlusions information by simulating the optical characteristics of retail depth cameras, e.g. the Kinect™ camera. Finally, the self-occlusion information was used to remove the affected joints from skeleton model. For each action, three self-occlusion sequences were computed matching the three perspectives of the video cameras. Hence, the HMM-GPD, the CHMM-GPD and a state-of-the-art architecture was evaluated on four datasets: three datasets containing data of only one perspective and a dataset containing data of all three perspectives. Testing was always done using data of each perspective separately.

The baseline experiment examines the state-of-the-art architecture proposed by Laptev et al. (2008). The architecture performs the classification in three steps. In the first step, interest points are assigned to spatial and temporal locations in the video stream. Then, HOG and HOF features are extracted and further processed to a fixed-length feature vector using the bag-of-words approach. In the last step, the fixed-length feature vector is classified using a one-vs-one SVM classifier system. The empirical evaluation yielded that the architecture is not view-invariant. A successful recognition using data of one perspective can only be achieved, if data of the target perspective is also present during training. Hence, all three perspectives were successfully recognized only in case data of all three camera perspectives were used for training (Table 5.3, page 132).

The second experiment examines the performance of the HMM-GPD approach. The results show that the HMM-GPD approach is partially view-invariant. The training using data of a single perspective led always to a reliable recognition on test data of the same perspective. In two cases, the training on data of a single perspective was sufficient to allow a reliable recognition using test data of another perspective. Performing the training using data of all three perspectives again provided a successful view-invariant recognition regarding each of the three perspectives. The performance measures clearly improved compared to the state-of-the-art architecture. However, view-invariance regarding the three perspectives was still only achieved in case the training was performed on data of all perspectives (Table 5.5, page 135). This finding can be related to the fact that some edges in distinct temporal segments (represented by hidden states) are insufficiently modeled by the HMM-GPD due to the missing coverage of the dataset.

The third experiment examines the CHMM-GPD which performs the classification using only a single trained model. The CHMM-GPD shares hidden states between action classes such that the chance for a sufficient modeling of edges for all hidden states increases. The results show a clearly improved view-invariant performance. In nine out of twelve cases, the approach achieved accuracies of over 94%, whereas two training and test configurations still achieved accuracies of over 73%. The remaining configurations which still have a lower performances can again be related to insufficiently modeled edges (Table 5.7, page 137). However, in case a sufficient data coverage is given, a robust recognition can be achieved on data generated from any of the three camera perspectives.

The experiments on the OAD and the UUlmMAD show that the HMM-GPD and CHMM-

GPD can be used to train and to test sequences and features of variable length. In the given application of action recognition, the data were only partially available due to occlusions. However, the approaches achieved still good view-invariance performance. The best approach turned out to be the CHMM-GPD which, in contrast to the HMM-GPD, requires only a single model to be trained due to the sharing of hidden states between classes.

6.3. CHMM-MCS for Spotting Laughter

The next study is composed of two experiments which make use of the pre-segmented and unsegmented sequences of the Freetalk Dataset which addresses laughter recognition in unconstrained dialogs. The first experiment examines three temporal multimodal classifier fusion architectures, whereas the second experiment analyzes the HMM-MCS and CHMM-MCS.

The first experiment which examines the three multimodal temporal classifier fusion architectures combines decisions derived from up to three sources, i.e. two audio feature sets (MOD), (PLP) and a video feature set (MOV). Each of the three architectures uses a different base classifier, namely the ESN, SVM or HMM. The empirical evaluation of the pre-segmented dataset showed that the use of additional modalities is in general advantageous (Table 5.9, page 145). The best performing approach was the SVM architecture using the MOD and PLP features which achieved an error rate of 3.7%. The second best performance was achieved by the HMM architecture using all three feature sets and resulted in an error rate of 6.3%. The ESN architecture was not evaluated because the sequences in the pre-segmented dataset were too short to perform the required adaptation phase.

The evaluation using the unsegmented dataset further approved the advantage of combining decisions from multiple modalities. However, a performance similar to the pre-segmented dataset was not achieved on the unsegmented dataset. Two findings were observed: (1) architectures making use of the MOV feature set had a lower performance and (2) the performances of the SVM architecture felt behind the one of the HMM architecture. The SVM architecture combining MOD and PLP features which achieved the best error rate in the pre-segmented setting yielded only a mid-field performance on the unsegmented dataset. Nonetheless, the SVM still had an acceptable error rate of 7.2% and the best precision in the unsegmented setting. However, taking a closer look at the other performance measures revealed that the outstanding precision came at the cost of the recall which had the lowest value of 24% in the complete study. The best recall of 85% was achieved by the HMM architecture using exclusively the MOV feature set. This HMM architecture yielded the highest error rate of 26.8% and a mid-field ranked F$_1$-measure of 56% due to a low precision of 42%. The lowest error rate of 6.5% on the unsegmented dataset was achieved by the HMM architecture using the MOD and PLP feature sets. The ratio of the precision (64%)

and recall (80%) resulted in the best F_1-measure of 72%. In summary, the experiment confirmed the benefits of combining multiple sources of decisions, the importance of examining unsegmented datasets and the need for a wise choice of performance measures.

The second experiment on the Freetalk Dataset examines the fusion performance of the HMM-MCS and the CHMM-MCS. In HMM-MCS, likelihoods need to be provided for each class and modality which are then combined using the probability class ratio. In contrast, the CHMM-MCS combines only the probability distributions over labels of the CHMM trained for each modality. Hence, the CHMM-MCS comes with the advantage that only one model needs to be trained for each modality, and that probability distributions over class labels are given natively which is beneficial for further processing. The experiment examines three approaches: (1) HMM-MCS, (2) restricted CHMM-MCS and (3) unrestricted CHMM-MCS.

The lowest error rate of 7.29% among all evaluated HMM-MCS was achieved by combining all available feature sets, i.e. MOD, PLP and MOV. A similar error rate of 7.45% was achieved using only the MOD and PLP feature sets. The combination of either the PLP or the MOD feature set with the MOV feature set led to an increase of the error rate which indicates the weakness of the video MOV feature set (Table 5.11a, page 147).

The restricted CHMM-MCS was designed to have the same expressiveness as the HMM-MCS by using a comparable transition matrix configuration. Two kinds of fusion strategies were evaluated together with the CHMM-MCS, i.e. averaging and multiplication. The achieved error rates were generally close to the error rates of the HMM-MCS. However, the combination of the MOD and PLP feature sets using the averaging fusion strategy achieved an error rate of 5.77% and, therefore, significantly outperformed the best HMM-MCS (Table 5.11b, page 147).

The last experiment studies the unrestricted CHMM-MCS which makes use of a full transition matrix. The approach outperformed the HMM-MCS in three settings: (1) multiplication fusion strategy using the MOD, PLP and MOV feature sets with an error rate of 5.74%, (2) multiplication fusion strategy using the MOD and PLP feature sets with an error rate of 5.70% and (3) averaging fusion strategy using the MOD and PLP feature sets with an error rate of 5.62% (Table 5.11c, page 147).

It is important to emphasize that due to the restricted setup of the study the capabilities of the CHMM and their combinations had by far not been driven to their limits. For instance, it is possible to combine class probability distributions of models with different configurations, e.g. models having different numbers of hidden states and mixture components or different kinds of transition matrices. Furthermore, a weighing of individual CHMM can be performed according to their classification performances.

Four findings can be drawn from the study: (1) CHMM-MCS are to be favored over the

HMM-MCS, (2) multimodal fusion can bring a considerable benefit, (3) results based on a pre-segmented dataset cannot be transfered to an unsegmented setting and (4) additional appropriate performance measures are of great value to properly assess classifier performances.

6.4. MFN and Kalman Filter for Classifier Fusion for Affective State Recognition

The MFN and the Kalman filter for classifier fusion is examined in a study using the application of affective state recognition. The introduction to this thesis identified different MCS requirements which were met during the development and empirical evaluation of the algorithms, e.g. processing of temporal and multimodal data, robustness to noisy features and the necessity to handle missing classifier decisions. The MFN and the Kalman filter for classifier fusion integrate multimodal classifier decisions over time in order to increase the accuracy and robustness of a classifier system. Furthermore, these approaches can partially compensate missing classifier decisions occurring for instance due to sensor failures or due to rejections of decisions because of low classifier confidences.

6.4.1. AVEC — Baseline Experiment

The participation at the Audio/Visual Emotion Challenge (AVEC) in 2011 was the starting point for a large study on temporal fusion MCS. The performance of the submission serves as a baseline for later experiments. The MCS was composed of three components, i.e. the audio, video and audio-visual components. The audio and video components utilized bagging with HMM and SVM, respectively. Predictions from the audio and video components were combined in the audio-visual component using an MLP. The AVEC 2011 provided four target classes, i.e. “Arousal”, “Valence”, “Expectancy” and “Power”. Nonetheless, the official winners of the AVEC 2011 were determined using only the class “Arousal”.

In general, the highest accuracies were achieved for the class “Arousal” closely followed by the class “Valence”. The classes “Power” and “Expectancy” had generally the lowest recognition rates regarding the three components. The proposed audio sub-challenge component achieved the second place among nine contributions with an accuracy of 63.5%, whereas the video sub-challenge component achieved the third place among four contributions with an accuracy of 56.9%. The contribution to audio-visual sub-challenge achieved an accuracy of 54.2% and, therefore, performed lower than the audio or video component (Table 5.12, page 154). The audio-visual challenge was officially withdrawn because of the low number of two participants. There is evidence that the predictions resulting from the audio and video modalities conflicted each other such that no stable mapping to the target class could be learned.

The baseline experiment shows that affective state recognition is a challenging task. Simple classifier functions and the bagging approach turned out to be the most reliable approaches. Furthermore, the fusion of classifier decisions on word and conversational turn level confirm the advantage of temporal fusion. However, the audio-visual component provided no satisfying performance due to inconsistent observations made in the individual modalities.

6.4.2. AVEC — Markov Fusion Network

The second experiment studies the MFN. The output of the MFN classifier architecture was submitted to the AVEC 2012 which extended the AVEC 2011 by providing continuous labels. The AVEC 2012 was rated using the absolute value of the averaged correlation coefficients between predictions and annotations derived for each recording. As described in Section 4.1.3.3, the absolute value of averaged correlation coefficients is a misleading measure since it can rate negative outcomes positively. Furthermore, we showed that the provided annotations are temporally biased. An unfitted logarithmic function achieved an outstanding rating which was able to compete with the winners of challenge and even beat them in one sub-challenge. A comprehensive discussion on this topic can be found in (Kächele et al., 2014b; Glodek et al., 2012c).

The proposed MFN classifier architecture processes class probabilities and classifier confidences which were obtained individually from the audio and video components. Class probabilities with low confidences were rejected such that only the remaining confident probabilities were combined using the MFN. The smoothness potential parameter was decreased for one frame at the beginning of each conversational turn based on the assumption that the subject's affective state stays stable within a turn. Our challenge results could neither compete with the challenge winner nor with the aforementioned logarithmic function (Table 5.13, page 156). We believe that the continuous annotations and the utilized performance measure of the AVEC 2012 may have distorted the competition. Therefore, the following experiments of this thesis were performed using the binarized annotations of the AVEC 2011.

The third experiment examines the MFN on the AVEC 2011 Dataset in three parts. The first part analyzes the performances of the raw unimodal base classifiers. The second part studies the performances of the unimodal fusion architectures using the MFN. Finally, the last part examines the performances of three MFN architectures which were presented in Section 3.5.2.

The results of the unimodal base classifiers provides the baseline for later evaluations of the MFN. Similar to the first baseline experiment on AVEC 2011 Dataset, the video modality turned out to be less suitable for predicting the classes “Arousal”, “Expectancy” and “Power”. Only the label dimension “Valence” constituted an exception. Higher rejection rates generally improved the accuracies but also amplified the trend of unilateral class distributions (Table 5.14, page 159).

In the second part, the unimodal decisions were temporally combined using the MFN. The performance measures were clearly improved compared with the unimodal baseline results. In case of the class “Arousal”, the audio MFN rendered an accuracy which improved approximately 10%. Other unimodal MFN achieved performances similar to the unimodal baseline with high rejection rates (Table 5.16, page 160). However, it has to be emphasized that the MFN reconstructed missing decisions such that the evaluation in this part of the experiment was performed considering all frames of the recordings.

The last part of the experiment examines the three MFN architectures for multimodal temporal fusion which follow the principle of the early (FRT), mid-level (RFT) and late (RTF) fusion defined in Section 3.5.2. The best performance was achieved by the mid-level fusion architecture (RFT). which can be related to the concurrent combination of input decisions when compared to the FRT and RTF architectures. Furthermore, the online processing mode of the MFN generally outperformed the offline processing mode (Table 5.18, page 161). This can be explained by the fact that no future information is propagated back in the online processing mode which matches the annotation process of the dataset.

The experiments show that the MFN can be applied in a large variety of possible architectures. Using the AVEC 2011 Dataset, the concurrent multimodal and temporal MFN architecture, i.e. the RFT architecture, using the online processing mode turned out to be the approach with the best performance. Although the MFN parameters and rejected decisions are easy to be interpreted, it is difficult to predict the implications of the interplay between the potential functions. Nonetheless, the MFN stands out by its exceptional characteristics: It reconstructs missing classifier decisions which may result from sensor failures or the use of a reject option. Furthermore, the provided probability distributions are well-suited for further processing. It is even possible to deploy the MFN in large surveillance areas. In this setting, classifier decisions from sensors can dynamically be switched on and off depending on the location of the subject to be monitored.

The results achieved with the MFN clearly outperformed the submission results of the original challenge. Moreover, the original dataset of the AVEC 2011 was repartitioned using the parts for which labels were available. As a result, the actual classification problem became more difficult. The folds were arranged to be subject-independent and, due to only a share of the original dataset was used, the folds themselves were smaller. Comparing the performances of the raw unimodal classifiers, the unimodal MFN and the multimodal MFN, it becomes clear that a major portion of the performance improvements can be contributed to the MFN. The next experiments examine the Kalman filter for classifier fusion in the same setting as the MFN

6.4.3. AVEC — Kalman Filter for Classifier Fusion

The last two experiments address the Kalman filter for classifier fusion using the AVEC 2011 Dataset and the AVEC 2013 Dataset. Analogously to the MFN, the Kalman filter for classifier fusion provides outputs for every frame, even if a frame is not supported by any decision.

The experiment which uses the AVEC 2011 Dataset reproduced the last two parts of the MFN experiment using the Kalman filter for classifier fusion. Therefore, two empirical evaluations were performed: (1) unimodal Kalman filter fusion and (2) multimodal Kalman filter fusion. The performance measures of the unimodal Kalman filter fusion improved the predictions of the raw unimodal classifiers. The unimodal Kalman filter fusion had a similar performance as the unimodal MFN. The audio and video Kalman filter fusion approaches achieved higher accuracies for the classes "Arousal" and "Expectancy", whereas the unimodal MFN using the online processing mode performed better for the classes "Power" and "Valence" (Table 5.20, page 165).

Analogously, the multimodal Kalman filter fusion and the multimodal MFN performed similarly. The multimodal Kalman filter fusion performed slightly better for the classes "Expectancy" and "Power", whereas the multimodal MFN outperformed the Kalman filter fusion with respect to the classes "Arousal" and "Valence". However, the differences are only marginal. Nonetheless, the improvement of the multimodal Kalman filter with respect to the baseline study is worth to be mentioned: The accuracy of the class "Arousal" increased from 54.2% to 68.5% (Table 5.22, page 166).

The last experiment examines the Kalman filter for classifier fusion using the the AVEC 2013. To do so, the approach was integrated into the classifier architecture which was evaluated on the official AVEC 2013 test set. In contrast to the previous study, the parameters of the Kalman filter were learned using the EM algorithm. The architecture outperformed the other contributions submitted to the AVEC 2013.

The study of the MFN and the Kalman filter for classifier fusion in the area of affective state recognition show that both approaches can compensate missing classifier decisions and, therefore, are well-suited for the application in real-world scenarios, e.g. the recognition of hard-to-classify affective states. In contrast to the MFN, the Kalman filter models the uncertainty explicitly. The MFN on the other hand bears the advantage that the parameters can be interpreted more intuitively. In summary, the MFN and the Kalman filter for classifier fusion proved both to be effective multimodal and temporal fusion techniques.

6.5. Layered Architecture for Complex Pattern Recognition

The recognition of complex entities which are not directly observable and occur in a large variety of appearances is a challenging task. The generic ULA addresses this issue with the help of multiple layers. Basic entities are recognized in the lowest layer based on sensor observations. The corresponding class decisions are collected and passed to the subsequent layer which uses them in order to recognize more abstract classes. This procedure is repeated until the desired complex entities is recognized.

The study of the ULA uses the Office Layered Architecture Dataset (OLAD) which was designed to have three layers of classes. The first layer recognizes elementary action classes based on features directly extracted from sensor data. The second layer recognizes activity classes which are composed of the actions classes of the previous layer. The third layer which uses the predictions of the activity classes as input infers the user preference classes. An object recognition task is included as well to provide an additional source of symbolic knowledge (class information) to support the first two layers.

The study addresses a number of topics: the qualification of different Markov models, the integration of sub-symbolic and symbolic information and the performances of pre-segmented and unsegmented pattern recognition. Furthermore, an additional experiment examines the performance of the MFN in the area of object recognition.

6.5.1. Object recognition Using the Markov Fusion Network

According to the design of the OLAD, the user preferences have to be inferred based on the observations of activities and actions. The activities and actions in turn are recognized based on the movements of the skeleton model. However, the object which is manipulated by the subject can provide additional evidence for actions and activities. Therefore, the OLAD features an additional object recognition task to detect the objects being manipulated in the hand of the subject.

The MFN MCS performing the object recognition uses the outputs of three base classifiers. Two classifiers use video feature vectors as inputs and one classifier operates on audio feature vectors. The multimodal classifier decisions are combined over time using an MFN. The results of the MFN MCS and four alternative fusion techniques, i.e. frame-wise and windowed fusion using the sum or the product, show that the MFN clearly outperformed the alternative approaches. Only the combination using the windowed sum achieved a competing performance (Table 5.26a, page 172). The performances of the approaches were further analyzed using ranked accuracies. The first two ranks of the windowed sum fusion achieved an accuracy of 96.5%, whereas the MFN achieved an accuracy of 98.3% (Table 5.26b, page 172). These results demonstrated the benefit of using the MFN to combine multiple modalities over time. Furthermore, the experiment shows that the MFN also allows modalities to contribute to one single class in a multi-class problem. The output of the object recognition was utilized as additional symbolic knowledge which was provided to the CHMM and FCHMM in the first two layers.

6.5.2. Action and Activity Recognition Using the CHMM and FCHMM

The action and activity recognitions were performed in the first two layers of the ULA. Three different Markov models were analyzed, i.e. the HMM, CHMM and FCHMM. The causal sequences provided additional external symbolic information to the CHMM and FCHMM. The fuzzy causal sequences of the FCHMM were allocated using the object probability distributions, whereas the causal sequences of the CHMM were allocated with the crisp object class assignments.

The evaluation of action recognition in the first layer shows that the FCHMM outperformed the CHMM and HMM (Table 5.27, page 173). The results provided first evidences for the benefit of using the additional external symbolic knowledge. A class probability distribution over actions for each time step was prepared and made available to the second layer.

The activity recognition in the subsequent layer was evaluated using all combinations of HMM, CHMM and FCHMM in the first and second layer. It turned out that the best accuracy of 82.3% was achieved when using the CHMM in the first layer and second layer. Using either the CHMM or the FCHMM in the first and the FCHMM in second layer yielded in the second best accuracy of 81.8%. Approaches using the HMM generally performed lower, e.g. the HMM in the first and second layer achieved an accuracy of only 71.1% (Table 5.28, page 174).

To validate the benefit of the ULA, an additional experiment in which the HMM, CHMM and FCHMM were directly applied to the classes of the second layer was performed (Table 5.30, page 174). The directly trained models yielded a surprisingly good performance. However, the high accuracies can be related to the pre-segmentation of the sequences. That means, the observed sequences of the directly trained models were comparably large, whereas the training and testing on the sequences provided by the first layer had to make use of smaller sequences due to the nested shifted windows. The missing information in the beginning and at the end of the pre-segmented activity test sequences caused that the uppermost layer had less information available. This assumption is confirmed by a follow-up experiment, in which the direct and layered activity recognition approaches were tested on complete user preference sequences. The activity recognition performed on complete user preference sequences can be compared with an unsegmented recognition setting. In case of the unsegmented setting, the performance of the directly trained models dropped severely. The best accuracy among the directly trained models was achieved by the CHMM with 65.4% (Table 5.31, page 176), whereas the layered approach yielded the best accuracy of 72.1% using the FCHMM in the first and the second layer (Table 5.32, page 176). Again we see that the measured performances based on pre-segmented data cannot be transfered to the unsegmented setting. Nonetheless, the recognition performance had decreased compared with the pre-segmented setting, because the unsegmented setting is clearly more challenging. The user preference recognition was realized using the FCHMM in the first and the second layer. The second layer provided a class probability distribution over activities for each time step to the third layer.

The patterns to be recognized in the third layer have a large variety of appearances and, therefore, require rules which relate activities happening at different points in time to each other. Hence, the third layer must make use of alternative approaches than classic pattern recognition approaches.

6.5.3. User Preference Recognition Using the MLN

The last layer recognizes the complex entity of user preference based on the output of the second layer. The dataset was designed such that logical rules can be used to describe the user preferences. However, a logic framework is too rigid because of the uncertainty associated with the predictions of the second layer. Hence, a combination of probability and first-order logic in form of the DMLN was utilized in the third layer. The complex interplay of activities and their relevance for the recognition of the user preference required the DMLN template model to be generated by human. The template model contained only one model parameter which was trained using the EM algorithm. The DMLN achieved an outstanding accuracy of 90.0% which is impressive regarding the fact that information was propagated through three layers (Table 5.35, page 178). Any wrong classifier decision in the former layer could have a fatal impact to the final user preference class assignment. As shown in Figure 5.17, the DMLN was able to correct wrong classifier decisions of the second layer with the help of the first-order logic rules. Hence, the modeled symbolic information in the DMLN was successfully integrated with sub-symbolic sensor data.

The study of the ULA on the OLAD offers many interesting findings. In the first part of the study, the MFN was evaluated and compared with alternative multimodal temporal fusion techniques among which the MFN turned out to be superior. Furthermore, decisions which give evidence for only a single class can be integrated into the MFN. The second part of the study evaluated the performances of the first two layers. The experiments show that the integration of additional external class information to control the selection of hidden states can increase the recognition performance of the layers. Furthermore, it was shown that performance measures obtained on pre-segmented sequences cannot necessarily by transfered to an unsegmented setting. In order to derive the input for the next layer, an unsegmented recognition of the next layer's data steam has to be performed. The errors occurring in this recognition can be compensated to a certain degree by the next layer. Finally, the last part of the study has demonstrated that the DMLN is well-suited to recognize complex entities in the ULA and can even correct wrong inputs originating from the preceding layer.

Chapter 7

Conclusions

In this work, pattern recognition approaches for cognitive technical systems (CTS) with a focus on real-world scenarios and complex environments were developed and evaluated. This chapter reviews the achievements towards the three research questions raised in Section 1.1. In the end, an outlook on interesting future work will be given.

(i) ***How to combine multimodal and temporal probabilistic classifier results?***
This question was addressed by introducing and evaluating two novel classifier fusion approaches, namely the Markov fusion network (MFN) and the Kalman filter for classifier fusion (Glodek et al., 2014c, 2013b).

Both approaches combine multiple streams of classifier decisions derived from different modalities to a single temporal stream of decisions. Furthermore, both approaches are able to handle missing classifier decisions which occur for instance due to sensor failures or due to a rejection of decisions because of a low classifier confidences (Glodek et al., 2012c,b). Both approaches allow individual weighting of base classifiers and a temporal smoothing. The MFN stands out by its intuitive and flexible parameterization, whereas the Kalman filter is a well understood method with a large number of established extensions. The algorithms were successfully applied in the context of affective state recognition in multiple editions of the Audio/Visual Emotion Challenge (AVEC) (Glodek et al., 2014c, 2013b). Further experiments proved their superiority over conventional fusion techniques (Glodek et al., 2014a).

Moreover, this thesis addressed parameter learning algorithms for MFN and Kalman filter for classifier fusion. The development of suitable parameter learning algorithms is challenging due to differences in base classifier performances and potentially missing class decisions. For instance, the absence or even different rates of classifier decisions can influence the fusion performance. In this work, a heuristic learning algorithm for the MFN was presented which addressed these aspects. Regarding the Kalman filter for classifier fusion, the well-known EM algorithm was examined (Kächele et al., 2014a).

(ii) ***How to handle missing information in sequential classifiers?***
The question is addressed by modeling information in the form of graphs which can vary in shape and size. However, it is not apparent how to perform a classification based on graph data, and even less on sequential graph data.

In this work, graph probability densities (GPD) as proposed by (Trentin and Di Iorio, 2008) were integrated into the hidden Markov model (HMM) in order to represent sequences of weighted graphs which vary over time in shape and size (Glodek et al., 2013d). The training of an HMM using GPD (HMM-GPD) requires that all possible edges in a graph are sampled at a rate which allows sufficient statistics to be learned. The HMM-GPD can be used for classification by fitting one model to each class. An alternative representation is provided by the conditioned hidden Markov model (CHMM) using GPD (CHMM-GPD). The CHMM-GPD makes it possible to share hidden states over classes such that only one model is needed to solve a classification problem.

The HMM-GPD and the CHMM-GPD were examined in the setting of activity recognition in which self-occlusion complicates the training and testing of image-based recognizers. The novel approaches use an application-specific representation of the subject given by a skeleton model. Such a skeleton model can be provided by retail depth cameras which, however, can capture only joints of the skeleton model being not afflicted by self-occlusions. In order to perform a comprehensive study, a complete skeleton model was recorded and self-occlusions was added later on with the help of a post-processing step. The partially available skeleton model is regarded as an incomplete feature vector. Two experiments were performed to study the new approaches: (1) a feasibility experiment using a retail depth camera and (2) a comparative experiment using a state-of-the-art motion capturing system and three video cameras. The first experiment shows that the HMM-GPD provides an outstanding classification performance even under heavy self-occlusions. Moreover, the experiment demonstrates that the algorithm allows the training based on incomplete feature vectors which facilitates the collection of data in real-world scenarios (Glodek et al., 2013d). The second experiment examines classifier performances with a special focus on view-invariance and compares the state-of-the-art computer vision architecture proposed by Laptev et al. (2008) with the HMM-GPD and the CHMM-GPD. The state-of-the-art approach recognized actions only from the perspectives which were already provided during training (Glodek et al., 2015b). The evaluations using the HMM-GPD showed a partial view-invariant action recognition. Whenever the training set comprised a sufficient coverage of graphs the HMM-GPD was able to achieve view-invariance. However, the chance of a good training set coverage regarding edges turned out to be still to low. The CHMM-GPD improves this coverage by unifying the representations into a single model. The evaluations showed that the recognition based on the CHMM-GPD was able to significantly improve view-invariance.

It has to be emphasized that the proposed algorithms do not only provide good classification performances, but also stand out by their unique properties. Training and testing was performed using sequences of graphs with varying size. The CHMM-GPD can group similar graphs of different classes and represent them jointly in a single hidden state. An article presenting the findings made in the study performed on the UUlmMAD is going to be published in the near future.

(iii) ***How to create high performance classifiers for complex classes?***
Two classifier architectures in which the class complexity is gradually increased and symbolic knowledge is used for the recognition of complex entities were introduced to address this question. The two proposed layered architectures for the recognition of complex entities are: (1) the unidirectional layered architecture (ULA) and (2) the bidirectional layered architecture (BLA).

Both architecture are composed of layers which focuses on a set of classes of different complexity. The lowest layer performs the recognition of basic classes which are then propagated to the next layer in which more complex classes with an increased temporal granularity are recognized. The ULA propagates information only upwards, whereas the BLA simultaneously propagates information back to lower layers.

In case of the ULA, any type of classifier can be applied to perform the recognition within a layer (Glodek et al., 2014a). However, due to the temporal characteristics of most patterns in real-world applications, the consideration of using Markov models is often inevitable. Furthermore, it is important to propagate probability distributions over classes to the next layer rather than crisp class assignments in order to preserve as much information as possible (Glodek et al., 2012d,d,a). The dynamic Markov logic network (DMLN) Geier and Biundo (2011) combines probability and first-order logic in a dynamical model and, hence, is a good candidate to realize the recognition of complex entities (Glodek et al., 2013a).

The BLA extends the ULA by allowing an exchange of information between the layers. To do so, the architecture is restricted to a conditioned hidden Markov model (CHMM) in the lower layer and to the DMLN in the upper layer. The training of the architecture can be performed similarly to the ULA. However, a set of random variables in both layers need to encode the same semantics. The random variables are unified during inference such that the states of the CHMM and DMLN can exchange their information.

The ULA is able to correct inconsistencies originating from wrong recognitions of lower layers. In addition to that, the BLA can propagate the corrected inconsistencies back to the lower layer such that a re-adaption can take place.

Two datasets were recorded to examine the proposed architectures. The first dataset examines the ULA and addresses the recognition of user preferences in an office scenario. The subject performs various activities which are composed of short actions. The order of activities is constrained by logical rules which help to specify the user preference to be recognized in the last layer. The second dataset was recorded to examine the BLA and is situated in a workout scenario. The subject performs various activities such as: putting objects on a table, putting objects back into a bag and using the objects. Again, the sequence of activities follows a set of rules. However, in this setting the information given by the rules can help to improve the recognition of the lower layer classes. The findings of this experiment will be published in the near future in cooperation with Thomas Geier from the Institute of Artificial Intelligence (University Ulm) and Georg Layher from the Vision and Perception Science Lab (University Ulm).

The study on the first dataset examines multiple aspects of the proposed ULA (Glodek et al., 2014a). The implementation makes use of sequential classifiers and realizes a consistent handling of uncertainty. It was shown that fuzzy conditioned hidden Markov models (FCHMM) are well-suited to be deployed in the lower layers, whereas the DMLN is qualified to recognize the complex entities. In summary, the study demonstrates the successful integration of symbolic DMLN knowledge with the sub-symbolic knowledge of classifiers using multiple layers of intermediate class representations.

In addition to the already presented algorithms further related approaches were developed and evaluated. One approach is the EGMM which extends the well-known GMM by the ensemble method to enhance the robustness of density estimation compared with classic approaches which was confirmed by two experiments (Glodek et al., 2013c). Furthermore, it was shown that an exhaustive parameter optimization which is usually required for model selection is less vital using the EGMM in order to obtain an accurate density estimation (Glodek et al., 2010). The second approach introduced in this work is the inequality constraint multi-class fuzzy-in and fuzzy-out SVM (IC-MC-F^2-SVM) which is a multi-class SVM operating on fuzzy class memberships (Glodek et al., 2014b). In contrast to the multi-class fuzzy-in and fuzzy-out SVM (MC-F^2-SVM) proposed by Schwenker et al. (2014) the fuzzy labels are modeled with the help of inequality constraints. Further work was published in collaboration with other scientist, e.g. (Schels et al., 2014; Scherer et al., 2012a,b). A detailed overview of all publications can be found on page 249 after the reference list.

7.1. Outlook

Although a large number of studies has been performed during this thesis, there are still many interesting developments and evaluations possible which can provide a deeper understanding of the proposed methods.

An exciting issue is the development of extensions to the MFN and Kalman filter for classifier fusion. Furthermore, a comparative study of both parameter learning algorithms can be conducted in order to further identify the characteristics and differences between the MFN and Kalman filter for classifier fusion. The classifier fusion approaches can for example also be deployed in large surveillance areas where the base classifiers can be dynamically switched on and off depending on the location of the monitored object.

The HMM-GPD can be studied in the field of object related activity recognition. Whenever objects are recognized close to the manipulation range of a subject, a corresponding edge can be dynamically added to the graph. This technique would allow the dynamic switching of context during classification. Furthermore, it would be beneficial to additionally model the orientation and location of the subject with respect to the world coordinate system in order to further reduce confusion between classes with similar movements, e.g. squat jump, toe touch and sit down. A more detailed examination of the CHMM-GPD regarding the transition matrix and hidden states can provide a deeper understanding of the model and the relations of classes.

The CHMM promises to have the ability to recognize class changes in windowed sequential data by using test label sequences which are composed of one classes in the first half and the other class in the second half of the causal sequence.

Up to this point, the recognition of complex classes requires human-generated models. The layered architecture could be extended in order to identify new activities and rules. At least the learning of rules will most probably still require human assistance, e.g. active learning. However, an adaptive layered architecture could be of great value for CTS.

In this work, pattern recognition approaches have been combined with approaches from symbolic AI. This constitutes a first step for the integration of the building blocks presented in the introduction. In the next steps, further components must be added to the CTS. However, it is not crucial to add a component to the CTS which can perform a complex task or do a special task exceptionally good, but to have a vast number of components realizing small functionalities. They contribute valuable information to the inner building block which, therefore, can build up a comprehensive model of the world state on which reasoning can be performed. This in turn leads to interaction patterns which match the situative context and, therefore, are perceived by the interacting subject as being based on profound knowledge. The basic components required for the realization of such a Companion system have already reached an elaborated state. They have to be assembled based on the paradigms examined in this work, e.g. small components and consistent handling of uncertainty (Glodek et al., 2015a). There is still much research to be done in order to realize the smooth transition and the interplay between these components.

Chapter A

Appendix

The current chapter provides supplemental material. Section A.1 provides an exhaustive list of all abbreviation used. The utilized notation is presented in Section A.2. Algorithms which would have exceeded the scope of text, but are still of relevance to this work, are provided in Section A.3. Section A.4 presents details on the utilized parameters which would have as well exceeded the scope of the text. A publication list with additional information is featured in Section A.5.

A.1. Abbreviations

This section provides a list of all abbreviations used in this thesis. Additional information about the corresponding term can be found in the index.

Name	Abbreviation
Aggregated expectation-maximization	Ag-EM
Audio/Visual Emotion Challenge	AVEC
Absolute correlation coefficient	CC
Beta Distribution Dataset	BDD
Bidirectional layered architecture	BLA
Computer Expression Recognition Toolbox	CERT
Conditioned hidden Markov model	CHMM
Cyclic pattern kernel	CPK
Cognitive technical system	CTS
Conditional random field	CRF
Dynamic Markov logic network	DMLN
Ensemble Gaussian mixture model	EGMM

Name	Abbreviation
Expectation-maximization	EM
Echo state network	ESN
Fuzzy conditioned hidden Markov model	FCHMM
Fully-continuous sub-challenge (AVEC 2012)	FCSC
Fuzzy-in fuzzy-out support vector machine	F^2-SVM
Fusion/rejection of uncertain decision/temporal integration	FRT
Graph probability density	GPD
Gaussian mixture model	GMM
Human computer interaction	HCI
Hidden Markov model	HMM
Histogram of colors	HOC
Histogram of optical flow	HOF
Histogram of orientation gradients	HOG
Hue-saturation-value color model	HSV
Inequality constraint multiple classes fuzzy-in fuzzy-out SVM	IC-MC-F^2-SVM
Independent and identically distributed	i.i.d.
Input-output hidden Markov model	IOHMM
Latent-dynamic conditional random field	LDCRF
Layered hidden Markov model	LHMM
Linear Predictive Coding	LPC
Multiple classes fuzzy-in fuzzy-out SVM	MC-F^2-SVM
Multiple classifier system	MCS
Mel frequency cepstral coefficients	MFCC
Markov fusion network	MFN
Markov logic network	MLN
Multi layer perceptron	MLP
Modulation spectrum feature (Freetalk datast)	MOD
Movement feature (Freetalk datast)	MOV
Markov random field	MRF
Naïve Bayes classifier	NBC
Occlusion action dataset	OAD
Ormoneit Dataset I	ODI
Ormoneit Dataset II	ODII
Occlusion action dataset	OAD
Office layered architecture dataset	OLAD
Perceptual linear prediction feature	PLP
Relative Spectral Transform	RASTA

Name	Abbreviation
Red-green-blue color model	RGB
Rejection of uncertain decision/fusion/temporal integration	RFT
Root mean square	RMS
Rejection of uncertain decision/temporal integration/fusion	RTF
Support vector machine	SVM
Unweighted accuracy	UA
Universal background model	UBM
Unidirectional layered architecture	ULA
UUlm Generic Layered Dataset	UUlmGLAD
UUlm Multi-perspective Dataset	UUlmMAD
Weighted accuracy	WA
Walk kernel	WK
Word-level sub-challenge (AVEC 2012)	WLSC

Abbreviations in the OAD

OAD class “Drink”	DC
OAD class “Eat”	EA
OAD class “Read”	RN

Abbreviations in uulmGLAD

Layer 1	
Pick up object	PUP_1
Move object towards head	MTH_1
Move object from head	MFH_1
Move object from table	MFT_1
Lay back object	TAM_1
Manipulate object in hand	HAM_1
Manipulate object at table	TAM_1
Manipulate object close to head	HEM_1
Empty hand	EH_1

Layer 2	
Drink from cup	DC_2
Fill milk into a cup	MC_2
Read note	RN_2
Stir cup	SC_2
Add sugar and stir	SSC_2
Write note	WN_2
No activity	HEM_2
Layer 3	
Black coffee	BC_3
White coffee	WC_3
Sweet coffee	SC_3
Sweet white coffee	SWC_3

Abbreviations in UUlmMAD

The abbreviations of classes used in the UUlmMAD are introduced in Figure 4.6 on page 99.

A.2. Notation

This section provides a list of notations in note form used in this thesis.

Description	Examples
Scalar values are in general denoted by lowercase, italic, non-bold letters with serifs. If the scalar value denotes a number of a set, it is usually written using an uppercase letter.	$x,\ y,\ i,\ N,\ M,\ \lambda$
Vectors are written using lowercase, non-italic, bold letters with serifs. Elements of vectors are indexed by the corresponding non-bold letter with an additional subscript, e.g. x_i.	$\mathbf{x},\ \boldsymbol{\mu}$
Matrices are written using uppercase non-italic, bold letters with serifs. Elements of the matrix are indexed by the corresponding non-bold letter with an additional subscript, e.g. x_{ij}, $x_{n-1,j}$. The identity matrix is denoted by $I \in \mathbf{R}^D$	$\mathbf{X},\ \boldsymbol{\Sigma}$
Sets are written using uppercase calligraphic letters. The cardinality of set are denoted by $\|\cdot\|$.	$\mathcal{T}, \mathcal{C}$
Sets of number, e.g. the set of real numbers, are written using blackboard bold letters. The first example shows the set of all real numbers, the second symbol shows the set of non-negative real numbers and the third symbol shows the set of natural numbers. Natural number are defined as positive integers without zero.	$\mathbb{R}, \mathbb{R}_{\geq 0}, \mathbb{Z}$
Functions are written using lowercase italic non-bold letters with serifs with a list of parameters encapsulated by round brackets. In case the function returns a vector it is written using a bold letter.	$f(\cdot),\ \mathbf{f}(\cdot)$
Distinct functions are denoted with fixed operator names: The expected value ($\mathbb{E}$), the variance (var), the covariance (cov), the determinant (det) and the normal distribution ($\mathcal{N}$).	$\mathbb{E}$, var, cov, det, $\mathcal{N}, \ldots$
Distinct variables are denoted with fixed operator names: The mean value ($\boldsymbol{\mu}$), the variance (σ), the covarianz ($\boldsymbol{\Sigma}$).	$\boldsymbol{\mu},\ \sigma,\ \boldsymbol{\Sigma}, \ldots$

Description	Examples
Tupels pool an arbitrary but finite number of ordered values using round brackets. They contain multiple instances of the same kind of elements.	(x, z), $(\mathbf{x}, y)$
Random variables are denoted using sans-serif letters. Vectors or matrices of random variables can be denoted using bold and uppercase letters as already described.	x, $\boldsymbol{\mathsf{x}}$, $\boldsymbol{\mathsf{X}}$
Functions of probabilities are denoted by $p(\cdot)$, or alternatively $q(\cdot)$. The assignment of random variables are given by $\mathsf{x} = x$. However, the notation is often simplified by only writing the assignment x. Conditioned probabilities are denoted by a vertical line, e.g. $p(\mathsf{x}\|\mathsf{y})$. The term $p(\mathsf{x}\|\mathsf{y})$ is read "probability of x given y". To allow the reader a fast mapping from a probability function, e.g. $p(\mathsf{w}_t = w_t\|\mathsf{w}_{t-1} = w_{t-1})$, to its representing parameter variable, e.g. $\mathbf{A}$, the parameter will often be written as being part of the probability function, e.g. $p(\mathsf{w}_t = w_t\|\mathsf{w}_{t-1} = w_{t-1} \shortmid \mathbf{A})$. Certain probability density functions have their own symbols, e.g. the Gaussian distribution $\mathcal{N}$ and the Gaussian mixture model $\mathcal{G}$.	$p(\cdot)$, $q(\cdot), p(\mathsf{x} = x) = p(x)$, $p(\boldsymbol{\mathsf{x}} = \mathbf{x}) = p(\mathbf{x})$, $p(\mathsf{x}\|\mathsf{y})$, $p(\boldsymbol{\mathsf{x}}_t\|\mathsf{w} = w_t \shortmid \mathbf{A})$
The quantity $\delta_{x=y}$ takes the value one in case the equation $x = y$ holds true and zero elsewhere.	$\delta_{x=y}$
Ranges are defined by the two quantities a and b placed within brackets. Depending on the kind of brackets, endpoints are either within the interval or not, e.g. $[a, b] := \{x \in \mathbb{R}\|a \leq x \leq b\}$ or $[a, b) := \{x \in \mathbb{R}\|a \leq x < b\}$.	$[a, b]$, (a, b)
Sets can alternatively be defined using an index with a lower and upper limit. For instance, the expression $\{\theta_l : l = 1, \ldots, L\}$ is equivalent to $\{\theta_l\}_{l=1}^{L}$.	$\{\theta_l\}_{l=1}^{L}$

A.3. Algorithms

The current section provides details on the algorithms which have been used in this work. However, a comprehensive presentation of these algorithms in the text would have exceeded the scope of the manuscript.

A.3.1. Cross-validation

The cross-validation was applied multiple times in the empirical evaluations. This section introduces the procedure and the terms used in this context.

The cross-validation is commonly used in a nested fashion, as illustrated by the 10×10 cross-validation in Figure A.1. The outer cross-validation takes the complete dataset and divides it into folds of development sets and corresponding test sets. The partitioning of the data is depicted in the upper part of the figure with ten folds. Each vertical division corresponds to a fold based on the complete dataset. Within a vertical division, the gray block corresponds to the test set and the concatenated white blocks to the development set. The test set is used to obtain the final statistics on the dataset and utilized models. To avoid overfitting during the parameter optimization on the development set, the development set is further divided into folds of training sets and validation sets, as depicted in the lower part of the figure. Each vertical division corresponds to a fold based on the development dataset. Within a vertical division, the gray block corresponds to the validation set and the concatenated white blocks to the training set. The parameters are determined with the help of all configurations of training and validation sets using parameter search. The $M \times N$ cross-validation would stand for dividing the complete dataset into M folds and the development set into N folds.

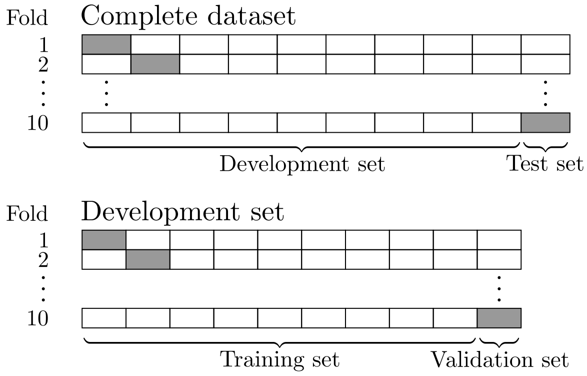


Figure A.1. Schematic drawing of a 10×10 cross-validation. The upper part of the figure shows the first partitioning of the complete dataset into 10 folds of development and test sets. The lower part of the figure shows how the development set is further partitioning into 10 folds of training and validation sets.

A.3.2. Ormoneit I & II

The following algorithm was first described by Ormoneit and Tresp (1996) and defines the generation of the distributions contained in Ormoneit I Dataset and Ormoneit II Dataset to study density classification performances.

Algorithm A.1: Algorithm proposed by Ormoneit and Tresp (1996) to create an artificial dataset for density estimation. Explained in detail in Section 4.1.1.

```
Data: Dimensionality of samples $D$, Even number of samples $N$, offset $\Delta$ of the two
      centers from the origin and standard deviation of the centers $\sigma$
Result: Training dataset $\mathcal{T} = \{(\mathbf{x}^{(n)}, y^{(n)})|\mathbf{x}^{(n)} \in \mathbb{R}^D$and class label $y^{(n)}\}_{n=1}^{N}$
for n=1 toN do
    $\mathbf{y}^{(n)} = \begin{cases} 0 & \text{if } n \leq N/2 \\ 1 & \text{otherwise} \end{cases}$ ;        /* Determine class label */
    $\mathbf{x} \sim \mathcal{N}(\cdot|\mathbf{0}, \mathbf{I}^D)$ ;        /* Draw from multivariate distribution */
    $\mathbf{x} = \mathbf{x}/||\mathbf{x}||_2$ ;        /* Project onto the unit circle */
    $\epsilon \sim \mathcal{N}(\cdot|0, \sigma^2)$ ;        /* Create dispertion around unit circle */
    $\mathbf{x} = (1 + \epsilon)\mathbf{x}$ ;
    $\mathbf{x}^{(n)} = \begin{cases} \mathbf{x} + \Delta & \text{if } n \leq N/2 \\ \mathbf{x} & \text{otherwise} \end{cases}$ ;        /* Offset according to class label */
    $\mathcal{T} = \mathcal{T} \cup \{(\mathbf{x}^{(n)}, y^{(n)})\}$
```

A.3.3. Markov fusion network

The program describes the gradient decent algorithm to derive the estimate of the MFN given a set of predictions.

Algorithm A.2: Markov fusion network gradient descent algorithm.

```
**Data**: Current estimate $\mathbf{Y} \in \mathbb{R}^{I \times T}$, predictions $\mathbf{X} \in \mathbb{R}^{M \times I \times T}$ and
model parameters $\mathbf{w} \in \mathbb{R}^{T-1}$, $\mathbf{k} \in \mathbb{R}^{M}$ and $u$
**Result**: Energy $E$ and the gradient $\frac{\partial \mathbf{Y}}{\partial y_{it}}$
$E_{\text{data}} = 0$; $E_{\text{smooth}} = 0$; $E_{\text{prob}} = 0$;
**for** $t$=1 **to**$T$ **do**
    $\sigma = 0$ ;                                                /* Initialization */
    **for** $i$=1 **to**$I$ **do**
        $\frac{\partial \mathbf{Y}}{\partial y_{it}} = 0$;
        $\sigma = \sigma + y_{it}$;
    $E_{\text{prob}} = E_{\text{prob}} + u \cdot \Big(1 - \sigma\Big)^2$;
    **for** $m$=1 **to**$M$ **do**
        **for** $i$=1 **to**$I$ **do**
            **if** isavailable($x_{imt}$) **then**                 /* $E_{\text{data}}$ */
                $E_{\text{data}} = E_{\text{data}} + k_{mt} \cdot \Big(y_{it} - x_{imt}\Big)^2$;
                $\frac{\partial \mathbf{Y}}{\partial y_{it}} = \frac{\partial \mathbf{Y}}{\partial y_{it}} + 2 \cdot k_{mt} \cdot \Big(y_{it} - x_{imt}\Big)$
            **if** $t$ == 0 **then**                             /* $E_{\text{smooth}}$ */
                $E_{\text{smooth}} = E_{\text{smooth}} + w_0 \cdot \Big(y_{i0} - y_{i1}\Big)^2$;
                $\frac{\partial \mathbf{Y}}{\partial y_{it}} = \frac{\partial \mathbf{Y}}{\partial y_{it}} + 2 \cdot w_0 \cdot \Big(y_{i0} - y_{i1}\Big)$
            **else if** $t < T$ - 1 **then**
                $E_{\text{smooth}} = E_{\text{smooth}} + w_{t-1} \cdot \Big(y_{it} - y_{it-1}\Big)^2 + w_t \cdot \Big(y_{it} - y_{it+1}\Big)^2$;
                $\frac{\partial \mathbf{Y}}{\partial y_{it}} = \frac{\partial \mathbf{Y}}{\partial y_{it}} + 2 \cdot w_{t-1} \cdot \Big(y_{it} - y_{it-1}\Big) + 2 \cdot w_t \cdot \Big(y_{it} - y_{it+1}\Big)$
            **else if** $t$ == $T$ - 1 **then**
                $E_{\text{smooth}} = E_{\text{smooth}} + w_{T-2} \cdot \Big(y_{iT-1} - y_{it-2}\Big)^2$;
                $\frac{\partial \mathbf{Y}}{\partial y_{it}} = \frac{\partial \mathbf{Y}}{\partial y_{it}} + 2 \cdot w_{T-2} \cdot \Big(y_{iT-1} - y_{it-2}\Big)$;
            $E_{\text{prob}} = E_{\text{prob}} + u \cdot \Big(1_{[0>y_{it}]} \cdot y_{it}\Big)^2$;    /* $E_{\text{prob}}$ */
            $\frac{\partial \mathbf{Y}}{\partial y_{it}} = \frac{\partial \mathbf{Y}}{\partial y_{it}} + u \cdot (2 - 2 \cdot \sigma) \cdot (-y_{it}) + u \cdot 2 \cdot 1_{[0>y_{it}]} \cdot y_{it}$;
$E = E_{\text{data}} + E_{\text{smooth}} + E_{\text{prob}}$;
```

A.4. Detailed Parameter Setting of Empirical Evaluations

To keep the manuscript uncluttered, we refrained from presentating detailed lists of parameter configurations during the actual empirical evaluation. The collected lists are to be found in this part of the Appendix.

Table A.1 contains the parameter settings used for the MFN in the AVEC 2012 which were used in the evaluation presented in Section 5.4.2.

Table A.1. Parameter settings of the MFN in the submission to the AVEC 2012.

Arousal	A. rej.	V. rej.	k_{audio}	k_{video}	w	w_{turn}
WLCS Audio	50%	-	0.3	-	192	8
WLCS Video	-	90%	-	0.5	64	8
FCSC Video	-	50%	-	0.5	192	8
WLCS Audiovisual	50%	90%	0.3	0.5	192	8
FCSC Audiovisual	50%	90%	0.3	0.5	192	8
Expectancy						
WLCS Audio	10%	-	0.3	-	192	8
WLCS Video	-	10%	-	0.5	128	8
FCSC Video	-	90%	-	0.5	64	8
WLCS Audiovisual	90%	10%	0.5	0.3	128	8
FCSC Audiovisual	90%	10%	0.5	0.3	128	8
Power						
WLCS Audio	-	-	0.3	-	192	8
WLCS Video	-	90%	-	0.5	64	8
FCSC Video	-	10%	-	0.5	192	8
WLCS Audiovisual	50%	90%	0.3	0.5	192	8
FCSC Audiovisual	50%	90%	0.3	0.5	64	8
Valence						
WLCS Audio	50%	-	0.3	-	192	8
WLCS Video	-	10%	-	0.5	192	8
FCSC Video	-	0%	-	0.5	192	8
WLCS Audiovisual	0%	10%	0.3	0.5	192	8
FCSC Audiovisual	50%	50%	0.3	0.5	192	8

Table A.2 provides the parameters used for the different MFN architectures evaluated on the AVEC 2011 dataset presented in Section 5.4.3.

Table A.2. Parameter settings of the MFN for the study using the AVEC 2011 Dataset.

(a) Unimodal (Table 5.16)

Audio	A.	E.	P.	V.
w_{u}	400	96	400	256
w_{t}	32	96	32	32
k_{audio}	0.1	0.2	0.1	0.2
Video	A.	E.	P.	v.
w_{u}	400	128	400	96
w_{t}	32	4	32	96
k_{video}	0.1	1.0	1.0	1.0

(b) Multimodal (Table 5.18)

FRT	A.	E.	P.	V.
w^{u}	400	400	400	400
w^{t}	32	32	32	32
k	0.5	0.5	0.5	0.5
λ_{audio}	1.0	0.1	0.1	1.0
λ_{video}	0.1	1.0	1.0	1.0
RFT	A.	E.	P.	V.
w^{u}	400	128	400	96
w^{t}	32	4	32	96
k_{audio}	0.1	1.0	1.0	1.0
k_{video}	0.1	1.0	1.0	1.0
RTF	A.	E.	P.	V.
$w^{\mathrm{u}}_{\mathrm{audio}}$	400	96	128	96
$w^{\mathrm{t}}_{\mathrm{audio}}$	32	96	4	4
k_{audio}	0.5	0.3	0.1	0.1
$w^{\mathrm{u}}_{\mathrm{video}}$	400	128	400	96
$w^{\mathrm{t}}_{\mathrm{video}}$	32	4	32	96
k_{video}	0.05	0.3	0.05	0.3

Table A.4 provides the parameters utilized for the HMM, CHMM, FCHMM in the Unidirectional layered architecture on the OLAD presented in Section 5.5.

Table A.4. Parameter settings of the Markov models in the second layer used in the study of the Office Layered Architecture Dataset. (a) Layered architecture parameter setting of the models on the second layer. (b) Parameter setting of the models directly trained to the classes of the second layer.

(a)

Layer 1	Layer 2	Hidden states	Mix. comp.	left-to-right	Covariance	p
FCHMM	FCHMM	5	2	no	diag	50
FCHMM	CHMM	5	3	yes	diag	-
FCHMM	HMM	2	3	no	full	-
CHMM	FCHMM	5	4	no	diag	100
CHMM	CHMM	5	3	yes	diag	-
CHMM	HMM	2	4	yes	full	-
CHMM	FCHMM	5	2	no	diag	50
CHMM	CHMM	5	3	no	diag	-
CHMM	HMM	2	4	no	diag	-

(b)

Layer 2	Hidden states	Mix. comp.	left-to-right	Covariance	p
FCHMM	4	4	no	full	30
CHMM	4	3	yes	full	-
HMM	3	4	yes	full	-

A.5. Abstracts of Main Contributions

This section provides a detailed list of publications in which the author of this thesis was involved. The elements of the list are color coded. The light gray color indicates a workshop contribution, the medium gray color a contribution to a conference and the dark color a contribution to an article. Each element lists the title, the abstract (if available), a set of Keywords, the citation and the name of the document enclosed to the thesis. The contributions are listed in chronological order.

Fusion Paradigms in Cognitive Technical Systems for Human-Computer Interaction

Abstract:

Recent trends in human-computer interaction (HCI) show a development towards cognitive technical systems (CTS) to provide natural and efficient operating principles. To do so, a CTS has to rely on data from multiple sensors which must be processed and combined by fusion algorithms. Furthermore, additional sources of knowledge have to be integrated, to put the observations made into the correct context. Research in this field often focus on optimizing the performance of the individual algorithms, rather than reflecting the requirements of CTS. This article presents the information fusion principles in CTS architectures we developed for Companion Technologies. Combination of information generally goes along with the level of abstractness, time granularity and robustness, such that large CTS architectures must perform fusion gradually on different levels—starting from sensor-based recognitions to highly abstract logical inferences. In our CTS application we sectioned information fusion approaches into three categories: perception-level fusion, knowledge-based fusion and application-level fusion. For each category, we introduce examples of characteristic algorithms. In addition, we provide a detailed protocol on the implementation performed in order to study the interplay of the developed algorithms.

Keywords: Temporal, multimodal, fusion, incomplete sequences of decisions, sequences of incomplete features, MCS, ULA, MFN, Kalman filter for classifier fusion

Citation:

M. Glodek, F. Honold, T. Geier, G. Krell, F. Nothdurft, S. Reuter, F. Schüssel, T. Hörnle, K. Dietmayer, W. Minker, S. Biundo, M. Weber, G. Palm, and F. Schwenker. Fusion paradigms in cognitive technical systems for human-computer interaction. *Neurocomputing*, 161(2015):17–37, 2015a. doi: 10.1016/j.neucom.2015.01.076

Filename: `2015_Glodek_HGKNRSHDMBWPS_NEUROCOMP.pdf`

UulmMAD—A Human Action Recognition Dataset for Ground-Truth Evaluation and Investigation of View Invariances

Abstract:

In recent time, human action recognition has gained increasing attention in pattern recognition. However, many datasets in the literature focus on a limited number of target-oriented properties. Within this work, we present a novel dataset, named UUlmMAD, which has been created to benchmark state-of-the-art action recognition architectures addressing multiple properties, e.g. high-resolutions cameras, perspective changes, realistic cluttered background and noise, overlap of action classes, different execution speeds, variability in subjects and their clothing, and the availability of a pose ground-truth. The UUlmMAD was recorded using three synchronized high-resolution cameras and an inertial motion capturing system. Each subject performed fourteen actions at least three times in front of a green screen. Selected actions in four variants were recorded, i.e. normal, pausing, fast and deceleration. The data has been post-processed in order to separate the subject from the background. Furthermore, the camera and the motion capturing data have been mapped onto each other and 3D-avatars have been generated to further extend the dataset. The avatars have also been used to emulate the self-occlusion in pose recognition when using a time-of-flight camera. In this work, we analyze the UUlmMAD using a state-of-the-art action recognition architecture to provide first baseline results. The results emphasize the unique characteristics of the dataset. The dataset will be made publicity available upon publication of the paper.

Keywords: Sequences of incomplete features, dataset

Citation:

M. Glodek, G. Layher, F. Heilemann, F. Gawrilowicz, G. Palm, F. Schwenker, and H. Neumann. UUlmMAD—a human action recognition dataset for ground-truth evaluation and investigation of view invariances. In F. Schwenker, S. Scherer, and L.-P. Morency, editors, *Proceedings of the International Workshop on Multimodal Pattern Recognition of Social Signals in Human-Computer-Interaction (MPRSS)*, volume 8869 of *Lecture Notes in Computer Science (LNCS)*, pages 77–91. Springer International Publishing, 2015b. ISBN 978-3-319-14898-4. doi: 10.1007/978-3-319-14899-1_8

Filename: `2015_Glodek_LHGPSN_MPRSS.pdf`

On Annotation and Evaluation of Multi-modal Corpora in Affective Human-Computer Interaction

Abstract:

In this paper, we discuss the topic of affective human- computer interaction from a data driven viewpoint. This comprises the collection of respective databases with emotional contents, feasible annotation procedures and software tools that are able to conduct a suitable labeling process. A further issue that is discussed in this paper is the evaluation of the results that are computed using statistical classifiers. Based on this we propose to use fuzzy memberships in order to model affective user state and endorse respective fuzzy performance measures.

Keywords: Temporal, multimodal, fusion, MCS, Kalman filter for classifier fusion

Citation:

M. Kächele, M. Schels, S. Meudt, V. Kessler, M. Glodek, P. Thiam, S. Tschechne, G. Palm, and F. Schwenker. On annotation and evaluation of multi-modal corpora in affective human-computer interaction. In R. Böck, F. Bonin, N. Campbell, and R. Poppe, editors, *Proceedings of Multimodal Analyses enabling Artificial Agents in Human-Machine Interaction*, volume 8757 of *Lecture Notes in Computer Science (LNCS)*, pages 35–44. Springer, 2014b. doi: 10.1007/978-3-319-15557-9_4

Filename: `2015_Kaechele_SKGTTPS_ma3hmi.pdf`

A layered architecture for probabilistic complex pattern recognition to detect user preferences

Abstract:

The recognition of complex patterns is nowadays one of the most challenging tasks in machine learning, and it promises to be of great benefit for many applications, e.g. by allowing advanced human computer interaction to access the user's situative context. This work examines a layered architecture that operates on different temporal granularities to infer complex patterns of user preferences. Classical hidden Markov models (HMM), conditioned HMM (CHMM) and fuzzy CHMM (FCHMM) are compared to find the best configuration in the lower architecture layers. In the uppermost layer, a Markov logic network (MLN) is applied to infer the user preference in a probabilistic rule-based manner. For each layer a comprehensive evaluation is given. We provide empirical evidence showing that the layered architecture using FCHMM and MLN is well-suited to recognize patterns on different layers.

Keywords: Temporal, integration of sub-symbolic and symbolic information, ULA, CHMM, FCHMM

Citation:

M. Glodek, T. Geier, S. Biundo, and G. Palm. A layered architecture for probabilistic complex pattern recognition to detect user preferences. *Journal of Biologically Inspired Cognitive Architectures*, 9:46–56, 2014a

Filename: `2014_Glodek_GBP_BICA90.pdf`

Combination of sequential class distributions from multiple channels using Markov fusion networks

Abstract:

The recognition of patterns in real-time scenarios has become an important trend in the field of multi-modal user interfaces in human computer interaction. Cognitive technical systems aim to improve the human computer interaction by means of recognizing the situative context, e.g. by activity recognition, or by estimating the affective state of the human dialogue partner. Classifier systems developed for such applications must operate on multiple modalities and must integrate the available decisions over large time periods. We address this topic by introducing the Markov fusion network (MFN) which is a novel classifier combination approach, for the integration of multi-class and multi-modal decisions continuously over time. The MFN combines results while meeting real-time requirements, weighting decisions of the modalities dynamically, and dealing with sensor failures. The proposed MFN has been evaluated in two empirical studies: the recognition of objects involved in human activities, and the recognition of emotions where we successfully demonstrate its outstanding performance. Furthermore, we show how the MFN can be applied in a variety of different architectures and the several options to configure the model in order to meet the demands of a distinct problem.

Keywords: Temporal, multimodal, fusion, MCS, MFN

Citation:

M. Glodek, M. Schels, F. Schwenker, and G. Palm. Combination of sequential class distributions from multiple channels using Markov fusion networks. *Journal on Multimodal User Interfaces (JMUI)*, 8(3):257–272, 2014c. doi: 10.1007/s12193-014-0149-0

Filename: `2014_Glodek_SSG_JMUI.pdf`

A New Multi-class Fuzzy Support Vector Machine Algorithm

Abstract:

Abstract. In this paper a novel approach to fuzzy support vector machines (SVM) in multi-class classification problems is presented. The proposed algorithm has the property to benefit from fuzzy labeled data in the training phase and can determine fuzzy memberships for input data. The algorithm can be considered as an extension of the traditional multi-class SVM for crisp labeled data, and it also extents the fuzzy SVM approach for fuzzy labeled training data in the two-class classification setting. Its behavior is demonstrated on three benchmark data sets, the achieved results motivate the inclusion of fuzzy labeled data into the training set for various tasks in pattern recognition and machine learning, such as the design of aggregation rules in multiple classifier systems, or in partially supervised learning.

Keywords: MC-F^2-SVM

Citation:

F. Schwenker, F. Markus, M. Glodek, M. Kächele, S. Meudt, M. Schels, and M. Schmidt. A new multi-class fuzzy support vector machine algorithm. In N. El Gayar, F. Schwenker, and C. Suen, editors, *Proceedings of the International Workshop on Artificial Neural Networks in Pattern Recognition (ANNPR)*, volume 8774 of *Lecture Notes in Computer Science (LNCS)*, pages 153–164. Springer, 2014. doi: 10.1007/978-3-319-11656-3_14

Filename: `2014_Schwenker_FGKMSS_ANNPR.pdf`

Fusion of audio-visual features using hierarchical classifier systems for the recognition of affective states and the state of depression

Abstract:

Reliable prediction of affective states in real world scenarios is very challenging and a significant amount of ongoing research is targeted towards improvement of existing systems. Major problems include the unreliability of labels, variations of the same affective states amongst different persons and in different modalities as well as the presence of sensor noise in the signals. This work presents a framework for adaptive fusion of input modalities incorporating variable degrees of certainty on different levels. Using a strategy that starts with ensembles of weak learners, gradually, level by level, the discriminative power of the system is improved by adaptively weighting favorable decisions, while concurrently dismissing unfavorable ones. For the final decision fusion the proposed system leverages a trained Kalman filter. Besides its ability to deal with missing and uncertain values, in its nature, the Kalman filter is a time series predictor and thus a suitable choice to match input signals to a reference time series in the form of ground truth labels. In the case of affect recognition, the proposed system exhibits superior performance in comparison to competing systems on the analysed dataset.

Keywords: Temporal, multimodal, fusion, MCS, Kalman filter for classifier fusion

Citation:

M. Kächele, M. Glodek, D. Zharkov, S. Meudt, and F. Schwenker. Fusion of audio-visual features using hierarchical classifier systems for the recognition of affective states and the state of depression. In M. De Marsico, A. Tabbone, and A. Fred, editors, *Proceedings of the International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, pages 671–678. SciTePress, 2014a

Filename: `2014_Kaechele_GZMS_ICPRAM_106_CR.pdf`

Using Speaker Group Dependent Modelling to Improve Fusion of Fragmentary Classifier Decisions

Abstract:

Current speech-controlled human computer inter- action is purely based on spoken information. For a successful interaction, additional information such as the individual skills, preferences and actual affective state of the user are often mandatory. The most challenging of these additional inputs is the affective state, since affective cues are in general expressed very sparsely. The problem can be addressed in two ways. On the one hand, the recognition can be enhanced by making use of already available individual information. On the other hand, the recognition is aggravated by the fact that research is often limited to a single modality, which in real-life applications is critical since recognition may fail in case sensors do not perceive a signal. We address the problem by enhancing the acoustic recognition of the affective state by partitioning the user into groups. The assignment of a user to a group is performed at the beginning of the interaction, such that subsequently a specialized classifier model is utilized. Furthermore, we make use of several modalities, acoustics, facial expressions, and gesture information. The combination of decisions not affected by sensor failures from these multiple modalities is achieved by a Markov Fusion Network. The proposed approach is studied empirically using the LAST MINUTE corpus. We could show that compared to previous studies a significant improvement of the recognition rate can be obtained.

Keywords: Temporal, multimodal, fusion, MCS, MFN

Citation:

I. Siegert, M. Glodek, and G. Krell. Using speaker group dependent modelling to improve fusion of fragmentary classifier decisions. In *Proceedings of the International IEEE Conference on Cybernetics (CYBCONF)*, pages 132–137. IEEE, 2013. doi: 10.1109/CYBConf.2013.6617458

Filename: `2013_Siegert_GPKSAW_Siegert_Cybconf.pdf`

Using unlabeled data to improve classification of emotional states in human computer interaction

Abstract:

The individual nature of physiological measurements of human affective states makes it very difficult to transfer statistical classifiers from one subject to another. In this work, we propose an approach to incorporate unlabeled data into a supervised classifier training in order to conduct an emotion classification. The key idea of the method is to conduct a density estimation of all available data (labeled and unlabeled) to create a new encoding of the problem. Based on this a supervised classifier is constructed. Further, numerical evaluations on the EmoRec II corpus are given, examining to what extent additional data can improve classification and which parameters of the density estimation are optimal.

Keywords: Temporal, multimodal, fusion, MCS, partially supervised learning

Citation:

M. Schels, M. Kächele, M. Glodek, D. Hrabal, S. Walter, and F. Schwenker. Using unlabeled data to improve classification of emotional states in human computer interaction. *Journal on Multimodal User Interfaces (JMUI) (Special Issue on From Multimodal Analysis to Real-Time Interactions with Virtual Agents)*, 8(1):5–16, 2014. doi: 10.1007/s12193-013-0133-0

Filename: `2013-Schels-KGHWS_JMUI.pdf`

Multi-Modal Classifier-Fusion for the Recognition of Emotions

Abstract:
<no abstract available>

Keywords: Temporal, multimodal, fusion, MCS, MFN

Citation:
M. Schels, M. Glodek, S. Meudt, S. Scherer, M. Schmidt, G. Layher, S. Tschechne, T. Brosch, D. Hrabal, S. Walter, G. Palm, H. Neumann, H. Traue, and F. Schwenker. Multi-modal classifier-fusion for the recognition of emotions. In *Coverbal synchrony in Human-Machine Interaction*, pages 73–97. CRC Press, 2013a
Filename: `2013_Schels_GMSSLTBHWTGNS_CoverbalSynchrony.pdf`

Multi-View Video Based Tracking and Audio-Visual Identification of Persons in a Human-Computer-Interaction Scenario

Abstract:
User identification and tracking are definitely the basic tasks in any human computer interaction (HCI) scenario. For these tasks we propose a multi-view approach utilizing multi- camera systems and audio processing systems. Face detectors and face recognizers are based on orientation histogram and eigenface techniques, and Mel Frequency Cepstral Coefficients (MFCC) are applied for speaker identification. In order to achieve a robust user identification and localization spatio-temporal classifier fusion methods have been integrated into the overall classifier system, support vector machines (SVM) and k nearest neighbor (kNN) models are used as base classifiers. A general office environment with up to six persons was the test bed for data collection and numerical evaluation.

Keywords: Temporal, multimodal, fusion, MCS

Citation:
S. Meudt, M. Glodek, M. Schels, and F. Schwenker. Multi-view video based tracking and audio-visual identification of persons in a human-computer-interaction scenario. In *Proceedings of the IEEE International Conference on Cybernetics (CYBCONF)*, pages 116–121, June 2013. doi: 10.1109/CYBConf.2013.6617454
Filename: `2013_Meudt_GSS.pdf`

Recognizing User Preferences Based on Layered Activity Recognition and First-Order Logic

Abstract:
Only few cognitive architectures have been proposed that cover the complete range from recognizers working on the direct sensor input, to logical inference mechanisms of classical artificial intelligence (AI). Logical systems operate on abstract predicates, which are often related to an action-like state transition, especially when compared to the classes recognized by pattern recognition approaches. On the other hand, pattern recognition is often limited to static patterns, and temporal and multimodal aspects of a class are often not regarded, e.g. by testing only on pre-segmented data. Recent trends in AI aim at developing applications and methods that are motivated by data-driven real world scenarios, while the field of pattern recognition attempts to push forward the boundary of pattern complexity. We propose a new generic architecture to close the gap between AI and pattern recognition approaches. In order to detect abstract complex patterns, we process sequential data in layers. On each layer, a set of elementary classes is recognized and the outcome of the classification is passed to the successive layer such that the time granularity increases. Layers can combine modalities, additional symbolic information or make use of reasoning algorithms. We evaluated our approach in an on-line scenario of activity recognition using three layers. The obtained results show that the combination of concepts from pattern recognition and high-level symbolic information leads to a prosperous and powerful symbiosis.

Keywords: Temporal, multimodal, fusion, MCS, ULA, MFN

Citation:
M. Glodek, T. Geier, S. Biundo, F. Schwenker, and G. Palm. Recognizing user preferences based on layered activity recognition and first-order logic. In *Proceedings of the International IEEE Conference on Tools with Artificial Intelligence (ICTAI)*, pages 648–653. IEEE, 2013a
Filename: `2013_Glodek_GBSP_ICTAI.pdf`

Hidden Markov Models with Graph Densities for Action Recognition

Abstract:
Human action recognition in video streams is a fast developing field in pattern recognition and machine learning. Local image representations, e.g. space-time interest points, have proven to be the current most reliable choice of feature in sequences in which the region of interest is difficult to determine. However, the question how to deal with more severe occlusions has large been ignored. This work proposes a new approach which directly addresses heavy occlusions by modeling the skeleton-based features using a probability density functions (PDF) defined over graphs. We integrated the proposed density into an hidden Markov model (HMM) to model sequences of graphs of arbitrary sizes, i.e. occlusions setting may change over time. The approach is evaluated using a dataset embracing three action classes, studying six different types of occlusions (involving the removal of subgraphs from the graphical representation of action sequence). The presented study shows clearly that actions from even heavily occluded sequences can be reliably recognized.

Keywords: Temporal, sequences of incomplete features, HMM GPD

Citation:
M. Glodek, S. Reuter, M. Schels, K. Dietmayer, and F. Schwenker. Kalman filter based classifier fusion for affective state recognition. In Z.-H. Zhou, F. Roli, and J. Kittler, editors, *Multiple Classifier Systems (MCS)*, volume 7872 of *Lecture Notes in Computer Science (LNCS)*, pages 85–94. Springer Berlin Heidelberg, 2013b. doi: 10.1007/978-3-642-38067-9_8
Filename: `2013_Glodek_TSP_IJCNN.pdf`

Kalman Filter Based Classifier Fusion for Affective State Recognition

Abstract:
The combination of classifier decisions is a common approach to improve classification performance. However, non-stationary fusion of decisions is still a research topic which draws only marginal attention, although more and more classifier systems are deployed in real-time applications. Within this work, we study Kalman filters as a combiner for temporally ordered classifier decisions. The Kalman filter is a linear dynamical system based on a Markov model. It is capable of combining a variable number of measurements (decisions), and can also deal with sensor failures in a unified framework. The Kalman filter is analyzed in the setting of multi-modal emotion recognition using data from the audio/visual emotional challenge 2011. It is shown that the Kalman filter is well-suited for real-time non-stationary classifier fusion. Combining the available sequential uni- and multimodal decisions does not only result in a consistent continuous stream of decisions, but also leads to significant improvements compared to the input decision performance.

Keywords: Temporal, multimodal, fusion, MCS, Kalman filter for classifier fusion

Citation:
M. Glodek, S. Reuter, M. Schels, K. Dietmayer, and F. Schwenker. Kalman filter based classifier fusion for affective state recognition. In Z.-H. Zhou, F. Roli, and J. Kittler, editors, *Multiple Classifier Systems (MCS)*, volume 7872 of *Lecture Notes in Computer Science (LNCS)*, pages 85–94. Springer Berlin Heidelberg, 2013b. doi: 10.1007/978-3-642-38067-9_8
Filename: `2013_Glodek_RSDS_MCS.pdf`

Fusion of Fragmentary Classifier Decisions for Affective State Recognition

Abstract:
Real human-computer interaction systems based on different modalities face the problem that not all information channels are always available at regular time steps. Nevertheless an estimation of the current user state is required at anytime to enable the system to interact instantaneously based on the available modalities. A novel approach to decision fusion of fragmentary classifications is therefore proposed and empirically evaluated for audio and video signals of a corpus of non-acted user behavior. It is shown that visual and prosodic analysis successfully complement each other leading to an outstanding performance of the fusion architecture.

Keywords: Temporal, multimodal, fusion, MCS, MFN

Citation:
G. Krell, M. Glodek, A. Panning, I. Siegert, B. Michaelis, A. Wendemuth, and F. Schwenker. Fusion of fragmentary classifier decisions for affective state recognition. In F. Schwenker, S. Scherer, and L.-P. Morency, editors, *Proceedings of the Workshop on Multimodal Pattern Recognition of Social Signals in Human-Computer-Interaction (MPRSS)*, volume 7742 of *Lecture Notes in Computer Science (LNCS)*, pages 116–130. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-37080-9. doi: 10.1007/978-3-642-37081-6_13
Filename: `2013_Krell_GPSMWS_MPRSS.pdf`

Multi-Modal Fusion Based on Classifiers Using Reject Options and Markov Fusion Networks

Abstract:

Classifying continuous signals from multiple channels poses several challenges: different sample rates from different types of channels have to be incorporated. Furthermore, when leaping from the laboratory to the real world, it is mandatory to deal with failing sensors and also uncertain or even incorrect classifications. We propose a new Multi Classifier System (MCS) based on the application of classifier making use of an reject option and a Markov Fusion Network (MFN) which is evaluated in an off-line and on-line manner. The architecture is tested using the publicly available AVEC corpus, that collects affectively labeled episodes of human computer interaction. The MCS achieved a significant improvement compared to the results obtained on the single modalities.

Keywords: Temporal, multimodal, fusion, MCS, MFN

Citation:

M. Glodek, M. Schels, G. Palm, and F. Schwenker. Multi-modal fusion based on classifiers using reject options and Markov fusion networks. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, pages 1084–1087. IEEE, 2012b

Filename: `2012_Glodek_SPS_ICPR12_preprint.pdf`

Spotting Laughter in Natural Multiparty Conversations: A Comparison of Automatic Online and Offline Approaches Using Audiovisual Data

Abstract:

It is essential for the advancement of human-centered multimodal interfaces to be able to infer the current user's state or communication state. In order to enable a system to do that, the recognition and interpretation of multimodal social signals (i.e., paralinguistic and nonverbal behavior) in real-time applications is required. Since we believe that laughs are one of the most important and widely understood social nonverbal signals indicating affect and discourse quality, we focus in this work on the detection of laughter in natural multiparty discourses. The conversations are recorded in a natural environment without any specific constraint on the discourses using unobtrusive recording devices. This setup ensures natural and unbiased behavior, which is of this work. one of the main foci To compare results of methods, namely Gaussian Mixture Model (GMM) supervectors as input to a Support Vector Machine (SVM), so-called Echo State Networks (ESN), and a Hidden Markov Model (HMM) approach, are utilized in online and offline detection experiments. The SVM approach proves very accurate in the offline classification task, but is outperformed by the ESN and HMM approach in the online detection (F1 scores: GMM SVM 0.45, ESN 0.63, HMM 0.72). Further, we were able to utilize the proposed HMM approach in a cross-corpus experiment without any retraining with respectable generalization capability (F1 score: 0.49). The results and possible reasons for these outcomes are shown and discussed in the article. The proposed methods may be directly utilized in practical tasks such as the labeling or the online detection of laughter in conversational data and affect-aware applications.

Keywords: Temporal, multimodal, fusion, MCS

Citation:

S. Scherer, M. Glodek, F. Schwenker, N. Campbell, and G. Palm. Spotting laughter in natural multiparty conversations: A comparison of automatic online and offline approaches using audiovisual data. *ACM Transactions on Interactive Intelligent Systems (TiiS) (Special Issue on Affective Interaction in Natural Environments)*, 2(1):4:1–4:31, 2012b. doi: 10.1145/2133366.2133370

Filename: `2012_Scherer_GSCP_ACMTIIS(2,1).pdf`

Recognizing Human Activities Using a Layered Markov Architecture

Abstract:
In the field of human computer interaction (HCI) the detection and classification of human activity patterns has become an important challenge. The problem can be understood as a specific problem of pattern recognition which addresses three topics, namely fusion of multiple modalities, spatio-temporal structures and a vast variety of pattern appearances the more abstract a pattern gets. In order to approach the problem, we propose a layered architecture which decomposes temporal patterns into elementary sub-patterns. Within each layer the patterns are detected using Markov models. The results of a layer are passed to the next successive layer such that on each layer the temporal granularity and the complexity of patterns increases. A dataset containing activities in an office scenario was recorded. The activities are decomposed to basic actions which are detected on the first layer. We evaluated a two-layered architecture using the dataset showing the feasibility of the approach.

Keywords: Temporal, multimodal, fusion, MCS, ULA

Citation:
M. Glodek, G. Layher, F. Schwenker, and G. Palm. Recognizing human activities using a layered Markov architecture. In A. E. Villa, W. Duch, P. Érdi, F. Masulli, and G. Palm, editors, *Proceedings of the International Conference on Artificial Neural Networks and Machine Learning (ICANN)*, volume 7552 of *Lecture Notes in Computer Science (LNCS)*, pages 677–684. Springer Berlin Heidelberg, 2012a. ISBN 978-3-642-33268-5. doi: 10.1007/978-3-642-33269-2_85
Filename: `2012_Glodek_LSP_ICANN.pdf`

A Generic Framework for the Inference of User States in Human Computer Interaction

Abstract:
The analysis of affective or communicational states in human-human and human-computer interaction (HCI) using automatic machine analysis and learning approaches often suffers from the simplicity of the approaches or that very ambitious steps are often tried to be taken at once. In this paper, we propose a generic framework that overcomes many difficulties associated with real world user behavior analysis (i.e. uncertainty about the ground truth of the current state, subject independence, dynamic realtime analysis of multimodal information, and the processing of incomplete or erroneous inputs, e.g. after sensor failure or lack of input). We motivate the approach, that is based on the analysis and spotting of behavioral cues that are regarded as basic building blocks forming user state specific behavior, with the help of related work and the analysis of a large HCI corpus. For this corpus paralinguistic and nonverbal behavior could be significantly associated with user states. Some of our previous work on the detection and classification of behavioral cues is presented and a layered architecture based on hidden Markov models is introduced. We believe that this step by step approach towards the understanding of human behavior underlined by encouraging preliminary results outlines a principled approach towards the development and evaluation of computational mechanisms for the analysis of multimodal social signals.

Keywords: Temporal, multimodal, fusion, MCS, ULA

Citation:
S. Scherer, M. Glodek, G. Layher, M. Schels, M. Schmidt, T. Brosch, S. Tschechne, F. Schwenker, H. Neumann, and G. Palm. A generic framework for the inference of user states in human computer interaction: How patterns of low level behavioral cues support complex user states in HCI. *Journal on Multimodal User Interfaces (JMUI)*, 6(3–4):117–141, 2012a. doi: 10.1007/s12193-012-0093-9
Filename: `2012_Scherer_GLSSTTSNP_JMUI_preprint.pdf`

Revisiting AVEC 2011 – An Information Fusion Architecture

Abstract:

Combining information from multiple sources is a vivid field of research. The problem of emotion recognition is inherently multi-modal. As automatic recognition of the emotional states is performed imperfectly by the single mode classifiers, its combination is crucial. In this work, the AVEC 2011 corpus is used to evaluate several machine learning techniques in the context of information fusion. In particular temporal integration of intermediate results combined with a reject option based on classifier confidences. The results for the modes are combined using a Markov random field that is designed to be able to tackle failures of individual channels.

Keywords: Temporal, multimodal, fusion, MCS, MFN

Citation:

M. Schels, M. Glodek, G. Palm, and F. Schwenker. Revisiting AVEC 2011—an information fusion architecture. In A. Esposito, S. Squartini, G. Palm, B. Apolloni, S. Bassis, A. Esposito, and F. C. Morabito, editors, *Neural Nets and Surroundings*, volume 19 of *Smart Innovation, Systems and Technologies*, pages 385–393. Springer Berlin Heidelberg, 2013b. ISBN 978-3-642-35466-3. doi: 10.1007/978-3-642-35467-0_38

Filename: `2012_Schels_GSP_WIRN12_preprint.pdf`

Multi-Modal Classifier-Fusion for the Classification of Emotional States in WOZ Scenarios

Abstract:

Learning from multiple sources is an important field of research in many applications. Amongst of the benefits of such an approach is that different sources can correct each other or that a failure of a channel can be easier compensated. The emotion of a subject can give helpful cues for a computer in a human machine dialog. The problem of emotion recognition is inherently multimodal. The most intuitive way of inferring a user state is to use facial expression and spoken utterances. However, bio-physiological readings can be helpful in this context. In this study, a novel information fusion architecture for the classification of human emotions in a computer interaction is proposed. We use information from the three modalities above mentioned. It turned out that the combination of different sources can be helpful for the classification. Also, a reject option for the classifiers is evaluated and yields promising results.

Keywords: Temporal, multimodal, fusion, MCS

Citation:

M. Schels, M. Glodek, S. Meudt, M. Schmidt, D. Hrabal, R. Böck, S. Walter, and F. Schwenker. Multi-modal classifier-fusion for the classification of emotional states in WOZ scenarios. In Y. G. Ji, editor, *Advances in Affective and Pleasurable Design*, number 22 in Advances in Human Factors and Ergonomics Series, pages 644–653. CRC Press, 2012a. ISBN 978-1439871188. doi: 10.1201/b12525-78

Filename: `2012_Schels_GMSHBWS_AHFE12.pdf`

Towards Emotion Recognition in Human Computer Interaction

Abstract:
The recognition of human emotions by technical systems is regarded as a problem of pattern recognition. Here methods of machine learning are employed which require substantial amounts of 'emotionally labeled' data, because model based approaches are not available. Problems of emotion recognition are discussed from this point of view, focusing on problems of data gathering and also touching upon modeling of emotions and machine learning aspects.

Keywords: Dataset

Citation:
G. Palm and M. Glodek. Towards emotion recognition in human computer interaction. In A. Esposito, S. Squartini, and G. Palm, editors, *Neural Nets and Surroundings*, volume 19 of *Smart Innovation, Systems and Technologies*, pages 323–336. Springer Berlin Heidelberg, 2013. doi: 10.1007/978-3-642-35467-0_32
Filename: `2012_Palm_G_Wirn12_preprint.pdf`

Ensemble Gaussian Mixture Models for Probability Density Estimation

Abstract:
Estimation of probability density functions (PDF) is a fundamental concept in statistics. This paper proposes an ensemble learning approach for density estimation using Gaussian mixture models (GMM). Ensemble learning is closely related to model averaging: while the standard model selection method determines the most suitable single GMM, the ensemble approach uses a subset of GMM which are combined in order to improve precision and stability of the estimated probability density function. The ensemble GMM is theoretically investigated and also numerical experiments were conducted to demonstrate benefits from the model. The results of these evaluations show promising results for classifications and the approximation of non-Gaussian PDF.

Keywords: EGMM

Citation:
M. Glodek, M. Schels, and F. Schwenker. Ensemble Gaussian mixture models for probability density estimation. *Computational Statistics*, 28(1):127–138, 2013c. doi: 10.1007/s00180-012-0374-5
Filename: `2012_Glodek_SS_Cost10_preprint.pdf`

Multiple Classifier Combination Using Reject Options and Markov Fusion Networks

Abstract:

The audio/visual emotion challenge (AVEC) resembles a benchmarking data collection in order to evaluate and develop techniques for the recognition of affective states. In our work, we present a Markov fusion network (MFN) for the combination of different individual classifiers, that is derived from the well-known Markov random fields (MRF). It is capable to restore missing values from a sequence of decisions and can integrate multiple channels and weights them dynamically using confidences. The approach shows promising challenge results compared to the baseline.

Keywords: Temporal, multimodal, fusion, MCS, MFN

Citation:

M. Glodek, M. Schels, G. Palm, and F. Schwenker. Multiple classifier combination using reject options and Markov fusion networks. In *Proceedings of the International ACM Conference on Multimodal Interaction (ICMI)*, pages 465–472. ACM New York, NY, 2012c. ISBN 978-1-4503-1467-1. doi: 10.1145/2388676.2388778

Filename: `2012_Glodek_SPS_ICMI12_preprint.pdf`

Detecting Actions by Integrating Sequential Symbolic and Sub-symbolic Information in Human Activity Recognition

Abstract:

Detecting human activities is a challenging field for sequential algorithms in machine learning and several approaches have already been proposed. One approach is to make use of the hierarchical structure of the activities to be classified by subdividing them into more elementary actions [12]. Alternatively the fusing of additional context information has been investigated to obtain a more meaningful feature space. Within this work both approaches are pursued by utilizing the layered architecture proposed by Oliver et al. with the conditioned hidden Markov model (CHMM) [8]. The model is evaluated using a dataset containing sequential sub-symbolic information (i.e. the position of body parts) and symbolic information (i.e. the detected object the person interacts with). The results outperform the classical approach making no use of the additional symbolic information.

Keywords: Temporal, multimodal, integration of sub-symbolic and symbolic information, MCS, ULA

Citation:

M. Glodek, F. Schwenker, and G. Palm. Detecting actions by integrating sequential symbolic and sub-symbolic information in human activity recognition. In P. Perner, editor, *Proceedings of the International Conference on Machine Learning and Data Mining (MLDM)*, volume 7376 of *Lecture Notes in Computer Science (LNCS)*, pages 394–404. Springer Berlin Heidelberg, 2012d. ISBN 978-3-642-31536-7. doi: 10.1007/978-3-642-31537-4_31

Filename: `2012_Glodek_SP_MLDM12.pdf`

Multimodal Emotion Classification in Naturalistic User Behavior

Abstract:
The design of intelligent personalized interactive systems, having knowledge about the user's state, his desires, needs and wishes, currently poses a great challenge to computer scientists. In this study we propose an information fusion approach combining acoustic, and bio-physiological data, comprising multiple sensors, to classify emotional states. For this purpose a multimodal corpus has been created, where subjects undergo a controlled emotion eliciting experiment, passing several octants of the valence arousal dominance space. The temporal and decision level fusion of the multiple modalities outperforms the single modality classifiers and shows promising results.

Keywords: Temporal, multimodal, fusion, MCS, dataset

Citation:
S. Walter, S. Scherer, M. Schels, M. Glodek, D. Hrabal, M. Schmidt, R. Böck, K. Limbrecht, H. Traue, and F. Schwenker. Multimodal emotion classification in naturalistic user behavior. In J. Jacko, editor, *Proceedings of the International Conference on Human-Computer Interaction: Towards Mobile and Intelligent Interaction Environments*, volume 6763 of *Lecture Notes in Computer Science (LNCS)*, pages 603–611. Springer, 2011
Filename: `2011_Walter_SSGHSBLTS_HCI11.pdf`

On the discovery of events in EEG data utilizing information fusion

Abstract:
One way to tackle brain computer interfaces is to consider event related potentials in electroencephalography, like the well established P300 phenomenon. In this paper a multiple classifier approach to discover these events in the bioelectrical signal and with them whether or not a subject has recognized a particular pattern, is employed. Dealing with noisy data as well as heavily imbalanced target class distributions are among the difficulties encountered. Our approach utilizes partitions of electrodes to create robust and meaningful individual classifiers, which are then subsequently combined using decision fusion. Furthermore, a classifier selection approach using genetic algorithms is evaluated and used for optimization. The proposed approach utilizing information fusion shows promising results (over 0.8 area under the ROC curve).

Keywords: Temporal, fusion, MCS

Citation:
M. Schels, S. Scherer, M. Glodek, H. Kestler, G. Palm, and F. Schwenker. On the discovery of events in EEG data utilizing information fusion. *Computational Statistics*, 28(1):5–18, 2013c. doi: 10.1007/s00180-011-0292-y
Filename: `2011_Schels_SGKPS_COST.pdf`

Multiple Classifier Systems for the Classification of Audio-Visual Emotional States

Abstract:

Research activities in the field of human-computer interaction increasingly addressed the aspect of integrating some type of emotional intelligence. Human emotions are expressed through different modalities such as speech, facial expressions, hand or body gestures, and therefore the classification of human emotions should be considered as a multimodal pattern recognition problem. The aim of our paper is to investigate multiple classifier systems utilizing audio and visual features to classify human emotional states. For that a variety of features have been derived. From the audio signal the fundamental frequency, LPC and MFCC coefficients, and RASTA-PLP have been used. In addition to that two types of visual features have been computed, namely form and motion features of intermediate complexity. The numerical evaluation has been performed on the four emotional labels Arousal, Expectancy, Power, Valence as defined in the AVEC dataset. As classifier architectures multiple classifier systems are applied, these have been proven to be accurate and robust against missing and noisy data.

Keywords: Temporal, multimodal, fusion, MCS

Citation:

M. Glodek, S. Tschechne, G. Layher, M. Schels, T. Brosch, S. Scherer, M. Kächele, M. Schmidt, H. Neumann, G. Palm, and F. Schwenker. Multiple classifier systems for the classification of audio-visual emotional states. In S. D'Mello, A. Graesser, B. Schuller, and J.-C. Martin, editors, *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, volume 6975 of *Lecture Notes in Computer Science (LNCS)*, pages 359–368. Springer Berlin Heidelberg, 2011d. ISBN 978-3-642-24570-1. doi: 10.1007/978-3-642-24571-8_47

Filename: `2011_Glodek_TLSBSKSNPS_ACII11.pdf`

Conditioned Hidden Markov Model Fusion for Multimodal Classification

Abstract:

Classification using hidden Markov models (HMM) is in general done by comparing the model likelihoods and choosing the class more likely to have generated the data. This work investigates a conditioned HMM which additionally provides a proba- bility for a class label and compares different fusion strategies. The notion is two-fold: on the one hand applications in affective computing might pass their uncertainty of the classification to the next processing unit, on the other hand different streams might be fused to increase the performance. The dataset studied incorporates two modalities and is based on a naturalistic multiparty dialogue. The goal is to discriminate between laughter and utterances. It turned out that the conditioned HMM outperforms classical HMM using different late fusion approaches while additionally providing a certainty about class decision.

Keywords: Temporal, multimodal, fusion, MCS, CHMM

Citation:

M. Glodek, S. Scherer, and F. Schwenker. Conditioned hidden Markov model fusion for multimodal classification. In *Proceedings of the Annual Conference of the International Speech Communication Association (ISCA), Interspeech*, pages 2269–2272. ISCA, ISCA, 2011c

Filename: `2011_Glodek_SS_Interspeech11_preprint.pdf`

Incorporating Uncertainty in a Layered HMM Architecture for Human Activity Recognition

Abstract:
In this study, conditioned HMM (CHMM), which inherit the structure from the latent-dynamic conditional random field (LDCRF) proposed by Morency et al. but is also based on a Bayesian network. Within the model a sequence of class labels is influencing a Markov chain of hidden states which are able to emit observations. The structure allows that several classes make use of the same hidden state. Categories and Subject Descriptors

Keywords: Temporal, multimodal, fusion, MCS, ULA, CHMM

Citation:
M. Glodek, L. Bigalke, M. Schels, and F. Schwenker. Incorporating uncertainty in a layered HMM architecture for human activity recognition. In *Proceedings of the Joint Workshop on Human Gesture and Behavior Understanding (J-HGBU)*, pages 33–34. ACM, 2011b. doi: 10.1145/2072572.2072584
Filename: `2011_Glodek_BSS_ACMMult11.pdf`

Recognizing Human Activities Using a Layered HMM Architecture

Abstract:
The development of so-called computer systems shows a tendency towards being aware of the user and the environment, offering a broad variety of interactions with the user. It is already feasible to detect faces, estimate the pose of the user, recognize emotion from speech, be aware of the environment and augment it with additional information. In this context, Oliver et al. proposed a layered cognitive system to detect human activities based on a multitude of modalities. The architecture detects complex activities based on a stream of crisp class assignments rendered by classifiers on the preceding layer. The current study investigates the possible increase in performance by passing the uncertainty of the class decision instead of crisp class assignments to the next layer. Oliver et al. utilized hidden Markov models (HMM) to detect the class on each layer. In order obtain a distribution over classes an alternative classifier, namely the conditioned HMM (CHMM) has been examined. The CHMM has the same structure as the latent-dynamic conditional random fields (LDCRF). Unlike the LDCRF, which is based on a Markov network, the CHMMM is based on a directed graph. Compared to the HMM each latent random variable is additionally influenced by a class node. The input-output hidden Markov model (IOHMM) proposed by Bengio et al. is, except of two aspects, also closely related to the CHMM. On the one hand, the IOHMM has additional edges connecting the class with the observation node for each time step. On the other hand, the CHMM models in analogy to the HMM and in contrast to the IOHMM the observations as emissions. The strong relation to HMM has the advantage that scientific contributions achieved for HMM can be applied without effort to the CHMM.

Keywords: Temporal, multimodal, fusion, MCS, ULA, CHMM

Citation:
M. Glodek, L. Bigalke, G. Palm, and F. Schwenker. Recognizing human activities using a layered HMM architecture. In B. Hammer and T. Villmann, editors, *Machine Learning Reports*, number 5 in New Challenges in Neural Computation (NC^2), pages 38–41. Springer Berlin Heidelberg, 2011a. doi: 10.1007/978-3-642-33269-2_85
Filename: `2011_Glodek_BPS_NC211.pdf`

Multiple Classifier Systems for the Recognition of Human Emotions

Abstract:

Research in the area of human-computer interaction (HCI) increasingly addressed the aspect of integrating some type of emotional intelligence in the system. Such systems must be able to recognize, interpret and create emotions. Although, human emotions are expressed through different modalities such as speech, facial expressions, hand or body gestures, most of the research in affective computing has been done in unimodal emotion recognition. Basically, a multimodal approach to emotion recognition should be more accurate and robust against missing or noisy data. We consider multiple classifier systems in this study for the classification of facial expressions, and additionally present a prototype of an audio-visual laughter detection system. Finally, a novel implementation of a Java process engine for pattern recognition and information fusion is described.

Keywords: Multimodal, fusion, MCS

Citation:

F. Schwenker, S. Scherer, M. Schmidt, M. Schels, and M. Glodek. Multiple classifier systems for the recognition of human emotions. In *Proceedings of the International Workshop on Multiple Classifier Systems (MCS)*, volume 5997 of *Lecture Notes in Computer Science (LNCS)*, pages 315–324. Springer, 2010

Filename: `2010_Schwenker_SSSG_MCS10.pdf`

A Hybrid Information Fusion Approach to Discover Events in EEG Data

Abstract:

<no abstract available>

Keywords: Multimodal, fusion, MCS

Citation:

M. Schels, S. Scherer, M. Glodek, H. A. Kestler, F. Schwenker, and G. Palm. A hybrid information fusion approach to discover events in EEG data. In H. A. Kestler, H. Binder, B. Lausen, H.-P. Klenk, M. Schmid, and F. Leisch, editors, *Proceedings of the International Workshop on Statistical Computing*, number 5 in Ulmer Informatik Bericht, pages 34–36. University of Ulm, 2010

Filename: `2010_Schels_SGKSP_UlmerInformatikBericht.pdf`

Artificial neural networks trained by ensembles of Gaussian mixture models

Abstract:

<no abstract available>

Keywords: EGMM

Citation:

M. Glodek, M. Schels, and F. Schwenker. Artificial neural networks trained by ensembles of Gaussian mixture models. In H. A. Kestler, H. Binder, B. Lausen, H.-P. Klenk, M. Schmid, and F. Leisch, editors, *Proceedings of the International Workshop on Statistical Computing*, number 5 in Ulmer Informatik Bericht, page 7. University of Ulm, 2010

Filename: `2010_Glodek_S_UlmerInformatikBericht.pdf`

References

M. A. R. Ahad. *Computer Vision and Action Recognition: a Guide for Image Processing and Computer Vision Community for Action Understanding.* Atlantis Ambient and Pervasive Intelligence. Atlantis Press, 2011.
Cited on page 10

M. A. R. Ahad, J. Tan, H. Kim, and S. Ishikawa. Human activity recognition: Various paradigms. In *Proceedings of the International Conference on Control, Automation and Systems (ICCAS)*, pages 1896–1901. IEEE, 2008. doi: 10.1109/ICCAS.2008.4694407.
Cited on page 8

N. Ambady and R. Rosenthal. Thin slices of expressive behavior as predictors of interpersonal consequences: a meta-analysis. *Psychological Bulletin*, 111(2):256, 1992.
Cited on page 7

T. Bänziger and K. R. Scherer. Using actor portrayals to systematically study multimodal emotion expression: The GEMEP corpus. In A. C. R. Paiva, R. Prada, and R. W. Picard, editors, *Proceedings of the International Conference of Affective Computing and Intelligent Interaction (ACII)*, volume 4738 of *Lecture Notes in Computer Science (LNCS)*, pages 476–487. Springer, 2007.
Cited on page 7

Y. Bar-Shalom and X.-R. Li. *Estimation and Tracking: Principles, Techniques, and Software.* Artech House Incorporated, 1993.
Cited on page 40, 41

A. Batliner, S. Steidl, C. Hacker, and E. Nöth. Private emotions versus social interaction: A data-driven approach towards analysing emotion in speech. *User Modeling and User-Adapted Interaction*, 18(1-2):175–206, 2008.
Cited on page 8

J. Ben-Arie, Z. Wang, P. Pandit, and S. Rajaram. Human activity recognition using multidimensional indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1091–1104, 2002.
Cited on page 10

Y. Bengio and P. Frasconi. Input-output HMMs for sequence processing. *IEEE Transactions on Neural Networks*, 7(5):1231–1249, 1996.
Cited on page 59

G. Bertrand, F. Nothdurft, W. Minker, S. Walter, and H. Traue. Adapting dialogue to user emotion—a Wizard-of-Oz study for adaptation strategies. In *Proceedings of the Workshop*

on Paralinguistic Information and its Integration in Spoken Dialogue Systems, pages 285–294. Springer, 2011.
Cited on page 8

M. Bicego, V. Murino, and M. Figueiredo. Similarity-based clustering of sequences using hidden Markov models. In *Proceedings of the International Conference on Machine Learning and Data Mining (MLDM)*, volume 2734 of *Lecture Notes in Computer Science (LNCS)*, pages 95–104. Springer, 2003. doi: 10.1007/3-540-45065-3_8.
Cited on page 156

C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer New York, NY, 2006. ISBN 978-0-387-31073-2.
Cited on page 13, 15, 16, 17, 19, 20, 22, 23, 24, 25, 26, 33, 35, 40, 43, 44, 45, 46, 56, 83, 89, 120, 121, 257

R. Biswas, S. Thrun, and K. Fujimura. Recognizing activities with multiple cues. In A. Elgammal, B. Rosenhahn, and R. Klette, editors, *Proceedings of the International Conference on Human Motion—Understanding, Modeling, Capture and Animation*, volume 4814 of *Lecture Notes in Computer Science (LNCS)*, pages 255–270. Springer Berlin Heidelberg, 2007. ISBN 978-3-540-75702-3. doi: 10.1007/978-3-540-75703-0_18.
Cited on page 10

S. Blackman and R. Popoli. *Design and Analysis of Modern Tracking Systems.* Artech House Publishers, 1999.
Cited on page 40

I. Bloch, A. Hunter, A. Appriou, A. Ayoun, S. Benferhat, P. Besnard, L. Cholvy, R. Cooke, F. Cuppens, D. Dubois, et al. Fusion: General concepts and characteristics. *International Journal of Intelligent Systems*, 16(10):1107–1134, 2001. doi: 10.1002/int.1052.
Cited on page 13, 46

A. Bobick and J. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):257–267, 2001.
Cited on page 9

J. Bortz and C. Schuster. *Statistik für Human- und Sozialwissenschaftler.* Springer, 2010.
Cited on page 13

G. E. Box, G. M. Jenkins, and G. C. Reinsel. *Time Series Analysis: Forecasting and Control.* John Wiley & Sons, 4th edition, 2008.
Cited on page 46

M. Brand, N. Oliver, and A. Pentland. Coupled hidden Markov models for complex action recognition. In *Proceedings of the International IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 994–999. IEEE, 1997. doi: 10.1109/CVPR.1997.609450.
Cited on page 59

L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996. ISSN 0885-6125. doi: 10.1007/BF00058655.
Cited on page 45, 57, 125, 152, 155

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001. doi: 10.1023/A:1010933404324.
Cited on page 28, 156

F. Burkhardt, A. Paeschke, M. Rolfes, W. Sendlmeier, and B. Weiss. A database of German emotional speech. In *Proceedings of the Annual Conference of the International Speech Communication Association (ISCA), Interspeech.* ISCA, ISCA, 2005.
Cited on page 7

N. Campbell, T. Sadanobu, M. Imura, N. Iwahashi, S. Noriko, and D. Douxchamps. A multimedia database of meetings and informal interactions for tracking participant involvement and discourse flow. In *Proceedings of the International Language Resources and Evaluation Conference (LREC)*, pages 391–394. European Language Resources Association (ELRA), 2006a.
Cited on page 7

W. Campbell, D. Sturim, and D. Reynolds. Support vector machines using GMM supervectors for speaker verification. *Signal Processing Letters, IEEE*, 13(5):308–311, 2006b.
Cited on page 144

W. Cannon. The James-Lange theory of emotions: A critical examination and an alternative theory. *The American Journal of Psychology*, 39(1/4):106–124, 1927.
Cited on page 6, 7

G. Casella and R. Berger. *Statistical Inference*. Duxbury Press, 2001.
Cited on page 14

H. Chen, H. Chen, Y. Chen, and S. Lee. Human action recognition using star skeleton. In *Proceedings of the ACM International Workshop on Video Surveillance and Sensor Networks (VSSN)*, pages 171–178. ACM, 2006. doi: 10.1145/1178782.1178808.
Cited on page 4, 9

W. Cheung and G. Hamarneh. N-sift: N-dimensional scale invariant feature transform for matching medical images. In *Proceedings of the IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI)*, pages 720–723. IEEE, 2007.
Cited on page 9

C. Chow. On optimum recognition error and reject tradeoff. *IEEE Transactions on Information Theory*, 16(1):41–46, 1970.
Cited on page 46

N. Christiani and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, 2000.
Cited on page 49

D. Cook and L. Holder. *Mining Graph Data*. Wiley-Interscience, 2006.
Cited on page 28

R. Cowie, E. Douglas-Cowie, S. Savvidou, E. McMahon, M. Sawey, and M. Schröder. FEELTRACE: An instrument for recording perceived emotion in real time. In *Proceedings of the ISCA Tutorial and Research Workshop (ITRW) on Speech and Emotion*, 2000.
Cited on page 110

R. Cowie, E. Douglas-Cowie, N. Tsapatsoulis, G. Votsis, S. Kollias, W. Fellenz, and J. Taylor. Emotion recognition in human-computer interaction. *IEEE Signal Processing Magazine*, 18(1):32–80, 2001.
Cited on page 8

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893. IEEE, 2005.
Cited on page 9

L. De Raedt. *Logical and Relational Learning*, volume XV of *Cognitive Technologies*. Springer Science & Business Media, 2008. ISBN 978-3-540-68856-3.
Cited on page 41

R. de Salvo Braz, E. Amir, and D. Roth. A survey of first-order probabilistic models. In D. E. Holmes and L. C. Jain, editors, *Innovations in Bayesian Networks*, volume 156 of *Studies in Computational Intelligence*, pages 289–317. Springer, 2008. doi: 10.1007/

978-3-540-85066-3_12.
Cited on page 41

A. Dempster. A generalization of Bayesian inference. Technical Report 20, Harvard University, Department of Statistics, Cambridge, Massachusetts, November 1967.
Cited on page 14

A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1): 1–38, 1977.
Cited on page 25, 33, 35

A. K. Dey and G. D. Abowd. Towards a better understanding of context and context-awareness. In H.-W. Gellersen, editor, *Proceedings of the International Symposium on Handheld and Ubiquitous Computing*, volume 1707 of *Lecture Notes in Computer Science (LNCS)*, pages 304–307. Springer London, UK, 1999. ISBN 978-3-540-66550-2. doi: 10.1007/3-540-48157-5_29.
Cited on page 1

J. Diebel and S. Thrun. An application of Markov random fields to range sensing. In *Proceedings of the International Conference on Advances in Neural Information Processing Systems (NIPS)*, volume 18, pages 291–298. MIT Press, 2006.
Cited on page 23

C. Dietrich, G. Palm, and F. Schwenker. Decision templates for the classification of bioacoustic time series. *Information Fusion*, 3(2):101 – 109, 2003. doi: 10.1016/S1566-2535(03)00017-4.
Cited on page 47

C. Dietrich, G. Palm, K. Riede, and F. Schwenker. Classification of bioacoustic time series based on the combination of global and local decisions. *Pattern recognition*, 37(12):2293–2305, 2004. doi: 10.1016/j.patcog.2004.04.004.
Cited on page 4

C. R. Dietrich. *Temporal Sensorfusion for the Classification of Bioacoustic Time.* PhD thesis, Institut of Neural Information Processing, University of Ulm, Ulm, Germany, 2004.
Cited on page 43, 47, 81

T. G. Dietterich. Ensemble methods in machine learning. In J. Kittler and F. Roli, editors, *Proceedings of the International Workshop on Multiple Classifier Systems (MCS)*, volume 1857 of *Lecture Notes in Computer Science (LNCS)*, pages 1–15. Springer, 2000.
Cited on page 45

J. Dinsmore, D. J. Chalmers, F. Adams, K. Aizawa, G. Fuller, J. Schwartz, B. Douglas S, L. A. Meeden, J. B. Marshall, J. A. Barnden, C.-D. Lee, M. Gasser, S. C. Kwasny, K. A. Faisal, and T. E. Lange. *The Symbolic and Connectionist Paradigms: Closing the Gap.* Lawrence Erlbaum Associates Inc. Hillsdale, New Jersey, 1992. ISBN 978-0805810806.
Cited on page 83

P. Domingos. *Artificial Intelligence: The First Hundred Years. AAAI Press*, chapter What's missing in AI: The interface layer. AAAI Press, 2006.
Cited on page 41

P. Domingos, S. Kok, H. Poon, M. Richardson, and P. Singla. Unifying logical and statistical ai. In *Proceedings of the International Conferenence of the Association for the Advancement of Artificial Intelligence (AAAI)*, volume 6, pages 2–7, 2006.
Cited on page 6

E. Douglas-Cowie, N. Campbell, R. Cowie, and P. Roach. Emotional speech: Towards a new generation of databases. *Speech Communication*, 40(1–2):33–60, 2003. doi: 10.1016/

S0167-6393(02)00070-5.
Cited on page 7

E. Douglas-Cowie, C. Cox, J. Martin, L. Devillers, R. Cowie, I. Sneddon, M. McRorie, C. Pelachaud, C. Peters, O. Lowry, et al. The HUMAINE database. *Emotion-Oriented Systems*, pages 243–284, 2011.
Cited on page 7

D. Dubois. Possibility theory and statistical reasoning. *Computational statistics & data analysis*, 51(1):47–69, 2006.
Cited on page 14

P. Ekman. Universals and cultural differences in facial expressions of emotion. In *Nebraska symposium on motivation*, volume 19. University of Nebraska Press, 1972.
Cited on page 6, 7

L. Fei-Fei and P. Perona. A Bayesian hierarchical model for learning natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 524–531. IEEE, 2005. doi: 10.1109/CVPR.2005.16.
Cited on page 5

J. Fontaine, K. Scherer, E. Roesch, and P. Ellsworth. The world of emotions is not two-dimensional. *Psychological Science*, 18(12):1050, 2007.
Cited on page 107

W. Freeman and M. Roth. Orientation histograms for hand gesture recognition. Technical Report TR94-03, Mitsubishi Electrical Research Laboratories, 1995. originally published at the International Workshop on Automatic Face and Gesture Recognition.
Cited on page 114

A. Freno and E. Trentin. *Hybrid Random Fields: A Scalable Approach to Structure and Parameter Learning in Probabilistic Graphical Models*, volume 15 of *Intelligent Systems Reference Library*. Springer, 2011. ISBN 978-3-642-20307-7. doi: 10.1007/978-3-642-20308-4.
Cited on page 22

J. Frommer, B. Michaelis, D. Rösner, A. Wendemuth, R. Friesen, M. Haase, M. Kunze, R. Andrich, J. Lange, A. Panning, and I. Siegert. Towards emotion and affect detection in the multimodal LAST MINUTE corpus. In *Proceedings of the International Language Resources and Evaluation Conference (LREC)*. European Language Resources Association (ELRA), 2012.
Cited on page 8

D. Gamrad. *Modeling, Simulation, and Realization of Cognitive Technical Systems.* PhD thesis, Universität Duisburg-Essen, 2011.
Cited on page 1

T. Gärtner, T. Horváth, Q. V. Le, A. J. Smola, and S. Wrobel. *Kernel Methods for Graphs*, chapter 5, pages 253–280. Mining in Graph Data. Wiley Interscience, 2007.
Cited on page 29

C. F. Gauß. *Theoria Motus Corporum Coelestium in Sectionibus Conicis Solem Ambientium.* F. Perthes and I.H. Besser, 1809.
Cited on page 24

D. Gehrig, P. Krauthausen, L. Rybok, H. Kuehne, U. Hanebeck, T. Schultz, and R. Stiefelhagen. Combined intention, activity, and motion recognition for a humanoid household robot. In *Proceedings of the International IEEE Conference on Intelligent Robots and Systems (IROS)*, pages 4819–4825. IEEE, 2011. doi: 10.1109/IROS.2011.6095118.
Cited on page 11

T. Geier and S. Biundo. Approximate online inference for dynamic Markov logic networks. In *Proceedings of the IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 764–768. IEEE, 2011.
Cited on page 43, 199

S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741, 1984. doi: 10.1109/TPAMI.1984.4767596.
Cited on page 23

G. Giacinto and F. Roli. *Hybrid Methods in Pattern Recognition*, volume 47, chapter 8, Design of Multiple Classifier Systems, pages 199–226. World Scientific Publishing, 2002. doi: 10.1142/9789812778147_0008.
Cited on page 44

M. Glodek, M. Schels, and F. Schwenker. Artificial neural networks trained by ensembles of Gaussian mixture models. In H. A. Kestler, H. Binder, B. Lausen, H.-P. Klenk, M. Schmid, and F. Leisch, editors, *Proceedings of the International Workshop on Statistical Computing*, number 5 in Ulmer Informatik Bericht, page 7. University of Ulm, 2010.
Cited on page 56, 200

M. Glodek, L. Bigalke, G. Palm, and F. Schwenker. Recognizing human activities using a layered HMM architecture. In B. Hammer and T. Villmann, editors, *Machine Learning Reports*, number 5 in New Challenges in Neural Computation (NC^2), pages 38–41. Springer Berlin Heidelberg, 2011a. doi: 10.1007/978-3-642-33269-2_85.
Cited on page 85

M. Glodek, L. Bigalke, M. Schels, and F. Schwenker. Incorporating uncertainty in a layered HMM architecture for human activity recognition. In *Proceedings of the Joint Workshop on Human Gesture and Behavior Understanding (J-HGBU)*, pages 33–34. ACM, 2011b. doi: 10.1145/2072572.2072584.
Cited on page 84, 85

M. Glodek, S. Scherer, and F. Schwenker. Conditioned hidden Markov model fusion for multimodal classification. In *Proceedings of the Annual Conference of the International Speech Communication Association (ISCA), Interspeech*, pages 2269–2272. ISCA, ISCA, 2011c.
Cited on page 141, 146

M. Glodek, S. Tschechne, G. Layher, M. Schels, T. Brosch, S. Scherer, M. Kächele, M. Schmidt, H. Neumann, G. Palm, and F. Schwenker. Multiple classifier systems for the classification of audio-visual emotional states. In S. D'Mello, A. Graesser, B. Schuller, and J.-C. Martin, editors, *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, volume 6975 of *Lecture Notes in Computer Science (LNCS)*, pages 359–368. Springer Berlin Heidelberg, 2011d. ISBN 978-3-642-24570-1. doi: 10.1007/978-3-642-24571-8_47.
Cited on page 109, 152

M. Glodek, G. Layher, F. Schwenker, and G. Palm. Recognizing human activities using a layered Markov architecture. In A. E. Villa, W. Duch, P. Érdi, F. Masulli, and G. Palm, editors, *Proceedings of the International Conference on Artificial Neural Networks and Machine Learning (ICANN)*, volume 7552 of *Lecture Notes in Computer Science (LNCS)*, pages 677–684. Springer Berlin Heidelberg, 2012a. ISBN 978-3-642-33268-5. doi: 10.1007/978-3-642-33269-2_85.
Cited on page 85, 199

M. Glodek, M. Schels, G. Palm, and F. Schwenker. Multi-modal fusion based on classifiers using reject options and Markov fusion networks. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, pages 1084–1087. IEEE, 2012b.
Cited on page 75, 197

M. Glodek, M. Schels, G. Palm, and F. Schwenker. Multiple classifier combination using reject options and Markov fusion networks. In *Proceedings of the International ACM Conference on Multimodal Interaction (ICMI)*, pages 465–472. ACM New York, NY, 2012c. ISBN 978-1-4503-1467-1. doi: 10.1145/2388676.2388778.
Cited on page 75, 110, 167, 190, 197

M. Glodek, F. Schwenker, and G. Palm. Detecting actions by integrating sequential symbolic and sub-symbolic information in human activity recognition. In P. Perner, editor, *Proceedings of the International Conference on Machine Learning and Data Mining (MLDM)*, volume 7376 of *Lecture Notes in Computer Science (LNCS)*, pages 394–404. Springer Berlin Heidelberg, 2012d. ISBN 978-3-642-31536-7. doi: 10.1007/978-3-642-31537-4_31.
Cited on page 85, 199

M. Glodek, T. Geier, S. Biundo, F. Schwenker, and G. Palm. Recognizing user preferences based on layered activity recognition and first-order logic. In *Proceedings of the International IEEE Conference on Tools with Artificial Intelligence (ICTAI)*, pages 648–653. IEEE, 2013a.
Cited on page 85, 199

M. Glodek, S. Reuter, M. Schels, K. Dietmayer, and F. Schwenker. Kalman filter based classifier fusion for affective state recognition. In Z.-H. Zhou, F. Roli, and J. Kittler, editors, *Multiple Classifier Systems (MCS)*, volume 7872 of *Lecture Notes in Computer Science (LNCS)*, pages 85–94. Springer Berlin Heidelberg, 2013b. doi: 10.1007/978-3-642-38067-9_8.
Cited on page 81, 197

M. Glodek, M. Schels, and F. Schwenker. Ensemble Gaussian mixture models for probability density estimation. *Computational Statistics*, 28(1):127–138, 2013c. doi: 10.1007/s00180-012-0374-5.
Cited on page 56, 123, 200

M. Glodek, E. Trentin, F. Schwenker, and G. Palm. Hidden Markov models with graph densities for action recognition. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pages 964–969. IEEE, 2013d.
Cited on page 73, 198

M. Glodek, T. Geier, S. Biundo, and G. Palm. A layered architecture for probabilistic complex pattern recognition to detect user preferences. *Journal of Biologically Inspired Cognitive Architectures*, 9:46–56, 2014a.
Cited on page 66, 84, 85, 86, 179, 197, 199, 200

M. Glodek, M. Schels, and F. Schwenker. Inequality-constraint multi-class fuzzy-in fuzzy-out support vector machines. In *Proceedings of the International Workshop on Statistical Computing*, number 4 in Ulmer Informatik Bericht, pages 31–32. University of Ulm, 2014b.
Cited on page 89, 200

M. Glodek, M. Schels, F. Schwenker, and G. Palm. Combination of sequential class distributions from multiple channels using Markov fusion networks. *Journal on Multimodal User Interfaces (JMUI)*, 8(3):257–272, 2014c. doi: 10.1007/s12193-014-0149-0.
Cited on page 75, 197

M. Glodek, F. Honold, T. Geier, G. Krell, F. Nothdurft, S. Reuter, F. Schüssel, T. Hörnle, K. Dietmayer, W. Minker, S. Biundo, M. Weber, G. Palm, and F. Schwenker. Fusion paradigms in cognitive technical systems for human-computer interaction. *Neurocomputing*, 161(2015):17–37, 2015a. doi: 10.1016/j.neucom.2015.01.076.
Cited on page 2, 3, 202

M. Glodek, G. Layher, F. Heilemann, F. Gawrilowicz, G. Palm, F. Schwenker, and H. Neumann. UUlmMAD—a human action recognition dataset for ground-truth evaluation and

investigation of view invariances. In F. Schwenker, S. Scherer, and L.-P. Morency, editors, *Proceedings of the International Workshop on Multimodal Pattern Recognition of Social Signals in Human-Computer-Interaction (MPRSS)*, volume 8869 of *Lecture Notes in Computer Science (LNCS)*, pages 77–91. Springer International Publishing, 2015b. ISBN 978-3-319-14898-4. doi: 10.1007/978-3-319-14899-1_8.
Cited on page 98, 198

L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2247–2253, 2007.
Cited on page 9

C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the Alvey Vision Conference (AVC)*, pages 147–151, 1988.
Cited on page 130

R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2003.
Cited on page 116

M. Hassner and J. Sklansky. The use of Markov random fields as models of texture. *Computer Graphics and Image Processing*, 12(4):357–370, 1980.
Cited on page 23

T. Hastie and R. Tibshirani. Classification by pairwise coupling. *Annals of Statistics*, 26 (2):451–471, 1998.
Cited on page 46, 52

S. Haykin. *Neural Networks: A Comprehensive Foundation.* Prentice Hall, 1999. ISBN 0132733501.
Cited on page 153

D. Heckerman. A tutorial on learning with Bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research, Microsoft Corporation, One Microsoft Way, Redmond, WA 98052, 1998.
Cited on page 16

J. Heinsohn and R. Socher-Ambrosius. *Wissensverarbeitung. Eine Einführung.* Spektrum, Berlin, 1999.
Cited on page 14

H. Hermansky and N. Morgan. RASTA processing of speech. *IEEE Transactions on Speech and Audio Processing (Special Issue on Robust Speech Recognition)*, 2:578–589, 1994.
Cited on page 105

H. Hermansky, N. Morgan, A. Bayya, and P. Kohn. RASTA-PLP speech analysis technique. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 121–124. IEEE, 1992.
Cited on page 108

F. Honold, F. Schüssel, and M. Weber. Adaptive probabilistic fission for multimodal systems. In *Proceedings of the Australian Computer-Human Interaction Conference (OzCHI)*, pages 222–231. ACM, 2012. ISBN 978-1-4503-1438-1. doi: 10.1145/2414536.2414575.
Cited on page 3

F. Honold, F. Schüssel, M. Munding, and M. Weber. Tangible context modelling for rapid adaptive system testing. In *Proceedings of the International Conference on Intelligent Environments (IE)*, pages 278–281. IEEE, 2013. doi: 10.1109/IE.2013.9.
Cited on page 1

X. Huang, A. Acero, and H. Hon. *Spoken Language Processing: A Guide to Theory, Algorithm and System Development.* Prentice Hall, 2001.
Cited on page 114

A. Inokuchi, T. Washio, and H. Motoda. Complete mining of frequent patterns from graphs: Mining graph data. *Machine Learning*, 50(3):321–354, 2003.
Cited on page 29

H. Jaeger. Short term memory in echo state networks. Technical Report GMD Report 152, Fraunhofer Institute for Autonomous Intelligent Systems, 2002.
Cited on page 141

B. Jeon and D. A. Landgrebe. Decision fusion approach for multitemporal classification. *IEEE Transactions on Geoscience and Remote Sensing*, 37(3):1227–1233, 1999. doi: 10.1109/36.763278.
Cited on page 46

D. Joanes and C. Gill. Comparing measures of sample skewness and kurtosis. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 47(1):183–189, 1998.
Cited on page 95

M. Kächele, M. Glodek, D. Zharkov, S. Meudt, and F. Schwenker. Fusion of audio-visual features using hierarchical classifier systems for the recognition of affective states and the state of depression. In M. De Marsico, A. Tabbone, and A. Fred, editors, *Proceedings of the International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, pages 671–678. SciTePress, 2014a.
Cited on page 81, 166, 198

M. Kächele, M. Schels, S. Meudt, V. Kessler, M. Glodek, P. Thiam, S. Tschechne, G. Palm, and F. Schwenker. On annotation and evaluation of multi-modal corpora in affective human-computer interaction. In R. Böck, F. Bonin, N. Campbell, and R. Poppe, editors, *Proceedings of Multimodal Analyses enabling Artificial Agents in Human-Machine Interaction*, volume 8757 of *Lecture Notes in Computer Science (LNCS)*, pages 35–44. Springer, 2014b. doi: 10.1007/978-3-319-15557-9_4.
Cited on page 105, 190

R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering*, 82(1):35–45, 1960.
Cited on page 40, 41

T. Kanade, J. Cohn, and Y. Tian. Comprehensive database for facial expression analysis. In *Automatic Face and Gesture Recognition*, pages 46–53. IEEE, 2000.
Cited on page 7

H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 321–328. AAAI Press, 2003.
Cited on page 29

J. Kelley. An iterative design methodology for user-friendly natural language office information applications. *ACM Transactions on Information Systems (TOIS)*, 2(1):26–41, 1984.
Cited on page 8

A. Kembhavi, T. Yeh, and L. Davis. Why did the person cross the road (there)? Scene understanding using probabilistic logic models and common sense reasoning. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 6312 of *Lecture Notes in Computer Science (LNCS)*, pages 693–706. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-15551-2. doi: 10.1007/978-3-642-15552-9_50.
Cited on page 11

J. Kittler, M. Hatef, R. P. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998. doi: 10.1109/34.667881.
Cited on page 4, 43, 44

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques.* MIT Press, 2009.
Cited on page 14, 15, 16, 17, 20, 23, 31, 32, 40, 88

G. Krell, M. Glodek, A. Panning, I. Siegert, B. Michaelis, A. Wendemuth, and F. Schwenker. Fusion of fragmentary classifier decisions for affective state recognition. In F. Schwenker, S. Scherer, and L.-P. Morency, editors, *Proceedings of the Workshop on Multimodal Pattern Recognition of Social Signals in Human-Computer-Interaction (MPRSS)*, volume 7742 of *Lecture Notes in Computer Science (LNCS)*, pages 116–130. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-37080-9. doi: 10.1007/978-3-642-37081-6_13.
Cited on page 75

L. I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms.* John Wiley & Sons, Inc., 2004. ISBN 9780471210788. doi: 10.1002/0471660264.
Cited on page 3, 43, 45, 46, 56

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 282–289. Morgan Kaufmann Publishers Inc., 2001.
Cited on page 22

I. Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2): 107–123, 2005.
Cited on page 130

I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008.
Cited on page 130, 134, 140, 186, 198

G. Layher, M. A. Giese, and H. Neumann. Learning representations for animated motion sequence and implied motion recognition. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, pages 288–295, 2012.
Cited on page 8

C. Lee, S. Yildirim, M. Bulut, A. Kazemzadeh, C. Busso, Z. Deng, S. Lee, and S. Narayanan. Emotion recognition based on phoneme classes. In *Proceedings of the Annual Conference of the International Speech Communication Association (ISCA), Interspeech.* ISCA, ISCA, 2004.
Cited on page 7

G. Littlewort, J. Whitehill, T. Wu, I. Fasel, M. Frank, J. Movellan, and M. Bartlett. The computer expression recognition toolbox (CERT). In *Proceedings of the IEEE Conference on Automatic Face and Gesture Recognition (FG)*, pages 298–305. IEEE, 2011. doi: 10.1109/FG.2011.5771414.
Cited on page 108, 110

B. D. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI)*, volume 81, pages 674–679, 1981.
Cited on page 9

H. Maganti, S. Scherer, and G. Palm. A novel feature for emotion recognition in voice based applications. *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 710–711, 2007.
Cited on page 7, 105

S. Marsella and J. Gratch. EMA: A process model of appraisal dynamics. *Cognitive Systems Research*, 10(1):70–90, 2009.
Cited on page 6

O. Martin, I. Kotsia, B. Macq, and I. Pitas. The eNTERFACE'05 audio-visual emotion database. In *Proceedings of the International Conference on Data Engineering Workshops*, page 8. IEEE, 2006.
Cited on page 7

G. McKeown, M. Valstar, R. Cowie, and M. Pantic. The SEMAINE corpus of emotionally coloured character interactions. In *Proceedings of the International Conference on Multimedia and Expo (ICME)*, pages 1079–1084. IEEE, 2010. doi: 10.1109/ICME.2010.5583006.
Cited on page 7, 107

A. Mehrabian. Pleasure-Arousal-Dominance: A general framework for describing and measuring individual differences in temperament. *Current Psychology*, 14(4):261–292, 1996.
Cited on page 6, 7

H. Meng and N. Bianchi-Berthouze. Naturalistic affective expression classification by a multi-stage approach based on hidden Markov models. In S. D'Mello, A. Graesser, B. Schuller, and J.-C. Martin, editors, *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, volume 6975 of *Lecture Notes in Computer Science (LNCS)*, pages 378–387. Springer, 2011. ISBN 978-3-642-24570-1. doi: 10.1007/978-3-642-24571-8_49.
Cited on page 165

H. Meng, D. Huang, H. Wang, H. Yang, M. AI-Shuraifi, and Y. Wang. Depression recognition based on dynamic facial and vocal expression features using partial least square regression. In *Proceedings of the ACM international workshop on Audio/visual emotion challenge*, pages 21–30. ACM, 2013.
Cited on page 168

A. Metallinou, M. Wöllmer, A. Katsamanis, F. Eyben, B. Schuller, and S. Narayanan. Context-sensitive learning for enhanced audiovisual emotion classification. *IEEE Transactions on Affective Computing*, 3(2):184–198, 2012.
Cited on page 8

S. Meudt, M. Glodek, M. Schels, and F. Schwenker. Multi-view video based tracking and audio-visual identification of persons in a human-computer-interaction scenario. In *Proceedings of the IEEE International Conference on Cybernetics (CYBCONF)*, pages 116–121, June 2013. doi: 10.1109/CYBConf.2013.6617454.
Cited on page 46

T. Mitchell. Generative and discriminative classifiers: Naive Bayes and logistic regression. Draft Version, 2005.
Cited on page 18

R. Möller and B. Neumann. Ontology-based reasoning techniques for multimedia interpretation and retrieval. In Y. Kompatsiaris and P. Hobson, editors, *Semantic Multimedia and Ontologies*, Part II, pages 55–98. Springer London, 2008. ISBN 978-1-84800-075-9. doi: 10.1007/978-1-84800-076-6_3.
Cited on page 5

L. Morency, A. Quattoni, and T. Darrell. Latent-dynamic discriminative models for continuous gesture recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, IEEE, 2007.
Cited on page 38, 39, 59

J. Mutch and D. Lowe. Object class recognition and localization using sparse features with limited receptive fields. *International Journal of Computer Vision*, 80(1):45–57, 2008.
Cited on page 109

J. Nicolle, V. Rapp, K. Bailly, L. Prevost, and M. Chetouani. Robust continuous prediction of human emotions using multiscale dynamic cues. In *Proceedings of the ACM International Conference on Multimodal Interaction (ICMI)*, pages 501–508. ACM, 2012.
Cited on page 112

N. Oliver, E. Horvitz, and A. Garg. Layered representations for human activity recognition. In *Proceedings of the International Conference on Multimodal Interfaces (ICMI)*, pages 1–6. IEEE, 2002. ISBN 0-7695-1834-6. doi: http://dx.doi.org/10.1109/ICMI.2002.1166960.
Cited on page 47, 49, 255

N. Oliver, A. Garg, and E. Horvitz. Layered representations for learning and inferring office activity from multiple sensory channels. *Journal of Computer Vision and Image Understanding (Special Issue on Event Detection in Video)*, 96(2):163–180, 2004. doi: 10.1016/j.cviu.2004.02.004.
Cited on page 11, 47, 49, 84, 257

D. Ormoneit and V. Tresp. Improved Gaussian mixture density estimates using Bayesian penalty terms and network averaging. *Proceedings of the International Conference on Advances in Neural Information Processing Systems (NIPS)*, pages 542–548, 1996.
Cited on page 26, 94, 125, 210, 259

D. Ormoneit and V. Tresp. Averaging, maximum penalized likelihood and Bayesian estimation for improving Gaussian mixture probability density estimates. *IEEE Transactions on Neural Networks*, 9(4):639–650, 1998.
Cited on page 56, 124, 125, 183, 184

G. Palm. *Novelty, Information and Surprise.* Springer, 2012. ISBN 978-3-642-29075-6.
Cited on page 13

G. Palm and M. Glodek. Towards emotion recognition in human computer interaction. In A. Esposito, S. Squartini, and G. Palm, editors, *Neural Nets and Surroundings*, volume 19 of *Smart Innovation, Systems and Technologies*, pages 323–336. Springer Berlin Heidelberg, 2013. doi: 10.1007/978-3-642-35467-0_32.
Cited on page 7, 75

E. Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.
Cited on page 24

H. Pasula and S. Russell. Approximate inference for first-order probabilistic languages. In *International Joint Conference on Artificial Intelligence (IJCAI)*, volume 17, pages 741–748, 2001.
Cited on page 41

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1988.
Cited on page 13, 16

P. Peursum, H. Bui, S. Venkatesh, and G. West. Human action recognition with an incomplete real-time pose skeleton. Technical Report 1, Department of Computing, Curtin University of Technology, Perth, Western Australia, May 2004.
Cited on page 4, 10

J. Platt. *Probabilistic Outputs for SV Machines*, chapter 5, pages 61–74. Neural Information Processing Series. MIT Press, 2000. ISBN 0262194481.
Cited on page 46, 49, 52, 144, 152, 169

J. C. Platt. Fast training of support vector machines using sequential minimal optimization. *Advances in Kernel Methods*, pages 185 – 208, 1999.
Cited on page 51

R. Poppe. A survey on vision-based human action recognition. *Image Vision Computing*, 28(6):976–990, 2010.
Cited on page 8, 9

L. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition.* Prentice Hall, 1993.
Cited on page 107

L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
Cited on page 31, 35

L. R. Rabiner and R. W. Schafer. *Digital Processing of Speech Signals.* Prentice Hall Signal Processing Series. Prentice Hall, 1978.
Cited on page 32

G. A. Ramirez, T. Baltrušaitis, and L.-P. Morency. Modeling latent discriminative dynamic of multi-dimensional affective signals. In S. D'Mello, A. Graesser, B. Schuller, and J.-C. Martin, editors, *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, volume 6975 of *Lecture Notes in Computer Science (LNCS)*, pages 396–406. Springer, 2011. doi: 10.1007/978-3-642-24571-8_51.
Cited on page 165

C. Rao, A. Yilmaz, and M. Shah. View-invariant representation and recognition of actions. *International Journal of Computer Vision*, 50(2):203–226, 2002.
Cited on page 10

M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006. doi: 10.1007/s10994-006-5833-1.
Cited on page 20, 41

M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Journal of Nature Neuroscience*, 2(11):1019–1025, 1999.
Cited on page 109

D. Roetenberg, H. Luinge, and P. Slycke. Xsens MVN: Full 6DOF human motion tracking using miniature inertial sensors. Technical report, Xsens Technologies B. V., 2009.
Cited on page 99

E. Rolls. Brain mechanisms for invariant visual recognition and learning. *Behavioural Processes*, 33(1-2):113–138, 1994.
Cited on page 109

J. Russell. A circumplex model of affect. *Journal of Personality and Social Psychology*, 39 (6):1161, 1980.
Cited on page 6, 7

E. Sánchez-Lozano, P. Lopez-Otero, L. Docio-Fernandez, E. Argones-Rúa, and J. L. Alba-Castro. Audiovisual three-level fusion for continuous estimation of Russell's emotion circumplex. In *Proceedings of the ACM International Workshop on Audio/visual Emotion Challenge (AVEC)*, pages 31–40. ACM, 2013.
Cited on page 166, 167, 168, 169, 260

C. Sanderson and K. K. Paliwal. Identity verification using speech and face information. *Digital Signal Processing*, 14(5):449–480, 2004.
Cited on page 44

A. Savran, H. Cao, M. Shah, A. Nenkova, and R. Verma. Combining video, audio and lexical indicators of affect in spontaneous conversation via particle filtering. In *Proceedings of the ACM International Conference on Multimodal Interaction (ICMI)*, pages 485–492. ACM, 2012.
Cited on page 111, 112

M. Schels, S. Scherer, M. Glodek, H. A. Kestler, F. Schwenker, and G. Palm. A hybrid information fusion approach to discover events in EEG data. In H. A. Kestler, H. Binder, B. Lausen, H.-P. Klenk, M. Schmid, and F. Leisch, editors, *Proceedings of the International Workshop on Statistical Computing*, number 5 in Ulmer Informatik Bericht, pages 34–36. University of Ulm, 2010.
Cited on page 44

M. Schels, M. Glodek, S. Meudt, M. Schmidt, D. Hrabal, R. Böck, S. Walter, and F. Schwenker. Multi-modal classifier-fusion for the classification of emotional states in WOZ scenarios. In Y. G. Ji, editor, *Advances in Affective and Pleasurable Design*, number 22 in Advances in Human Factors and Ergonomics Series, pages 644–653. CRC Press, 2012a. ISBN 978-1439871188. doi: 10.1201/b12525-78.
Cited on page 105

M. Schels, M. Kächele, D. Hrabal, S. Walter, H. Traue, and F. Schwenker. Classification of emotional states in a Woz scenario exploiting labeled and unlabeled bio-physiological data. In F. Schwenker and E. Trentin, editors, *Proceedings of the International Conference on Partially Supervised Learning (PSL)*, volume 7081 of *Lecture Notes in Computer Science (LNCS)*, pages 138–147. Springer, 2012b. doi: 10.1007/978-3-642-28258-4_15.
Cited on page 75

M. Schels, M. Glodek, S. Meudt, S. Scherer, M. Schmidt, G. Layher, S. Tschechne, T. Brosch, D. Hrabal, S. Walter, G. Palm, H. Neumann, H. Traue, and F. Schwenker. Multi-modal classifier-fusion for the recognition of emotions. In *Coverbal synchrony in Human-Machine Interaction*, pages 73–97. CRC Press, 2013a.
Cited on page 44, 110

M. Schels, M. Glodek, G. Palm, and F. Schwenker. Revisiting AVEC 2011—an information fusion architecture. In A. Esposito, S. Squartini, G. Palm, B. Apolloni, S. Bassis, A. Esposito, and F. C. Morabito, editors, *Neural Nets and Surroundings*, volume 19 of *Smart Innovation, Systems and Technologies*, pages 385–393. Springer Berlin Heidelberg, 2013b. ISBN 978-3-642-35466-3. doi: 10.1007/978-3-642-35467-0_38.
Cited on page 75

M. Schels, S. Scherer, M. Glodek, H. Kestler, G. Palm, and F. Schwenker. On the discovery of events in EEG data utilizing information fusion. *Computational Statistics*, 28(1):5–18, 2013c. doi: 10.1007/s00180-011-0292-y.
Cited on page 44

M. Schels, M. Kächele, M. Glodek, D. Hrabal, S. Walter, and F. Schwenker. Using unlabeled data to improve classification of emotional states in human computer interaction. *Journal on Multimodal User Interfaces (JMUI) (Special Issue on From Multimodal Analysis to Real-Time Interactions with Virtual Agents)*, 8(1):5–16, 2014. doi: 10.1007/s12193-013-0133-0.
Cited on page 47, 200

S. Scherer. *Analyzing the User's State in HCI: From Crisp Emotions to Conversational Dispositions*. PhD thesis, Institut of Neural Information Processing, University of Ulm, Ulm, Germany, 2011.
Cited on page 8

S. Scherer, M. Glodek, G. Layher, M. Schels, M. Schmidt, T. Brosch, S. Tschechne, F. Schwenker, H. Neumann, and G. Palm. A generic framework for the inference of

user states in human computer interaction: How patterns of low level behavioral cues support complex user states in HCI. *Journal on Multimodal User Interfaces (JMUI)*, 6 (3–4):117–141, 2012a. doi: 10.1007/s12193-012-0093-9.
Cited on page 1, 8, 200

S. Scherer, M. Glodek, F. Schwenker, N. Campbell, and G. Palm. Spotting laughter in natural multiparty conversations: A comparison of automatic online and offline approaches using audiovisual data. *ACM Transactions on Interactive Intelligent Systems (TiiS) (Special Issue on Affective Interaction in Natural Environments)*, 2(1):4:1–4:31, 2012b. doi: 10.1145/2133366.2133370.
Cited on page 106, 141, 143, 146, 175, 200, 257

A. Schmidt. Implicit human computer interaction through context. *Personal Technologies*, 4(2–3):191–199, 2000. doi: 10.1007/BF01324126.
Cited on page 1

M. Schmidt and F. Schwenker. Classification of graph sequences utilizing the eigenvalues of the distance matrices and hidden Markov models. In X. Jiang, M. Ferrer, and A. Torsello, editors, *Proceedings of the International Workshop on Graph-Based Representations in Pattern Recognition (GbRPR)*, volume 6658 of *Lecture Notes in Computer Science (LNCS)*, pages 325–334. Springer, 2011.
Cited on page 29

B. Schölkopf, A. Smola, R. Williamson, and P. Bartlett. New support vector algorithms. *Journal of Neural computation*, 12(5):1207–1245, 2000.
Cited on page 152

C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: A local SVM approach. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, volume 3, pages 32–36. IEEE, 2004.
Cited on page 9

B. Schuller, D. Seppi, A. Batliner, A. Maier, and S. Steidl. Towards more reality in the recognition of emotional speech. In *Proceedings of the International IEEE Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, pages 941–944. IEEE, 2007. doi: 10.1109/ICASSP.2007.367226.
Cited on page 7

B. Schuller, S. Steidl, A. Batliner, F. Burkhardt, L. Devillers, C. A. Müller, and S. S. Narayanan. The Interspeech 2010 paralinguistic challenge. In *Proceedings of the Annual Conference of the International Speech Communication Association (ISCA), Interspeech*, pages 2794–2797. ISCA, ISCA, 2010.
Cited on page 155

B. Schuller, M. Valstar, F. Eyben, G. McKeown, R. Cowie, and M. Pantic. AVEC 2011—The first international audio visual emotion challenges. In S. D'Mello, A. Graesser, B. Schuller, and J.-C. Martin, editors, *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, volume 6975 of *Lecture Notes in Computer Science (LNCS)*, pages 415–424. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-24570-1. doi: 10.1007/978-3-642-24571-8_53. Part II.
Cited on page 107, 108

B. Schuller, M. Valstar, F. Eyben, R. Roddy Cowie, and M. Pantic. AVEC 2012—The continuous audio/visual emotion challenges. In *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*. ACM, 2012.
Cited on page 111

F. Schwenker, C. R. Dietrich, C. Thiel, and G. Palm. Learning of decision fusion mappings for pattern recognition. *Journal on Artificial Intelligence and Machine Learning (AIML) (Special Issue on Multiple Classifier Systems)*, pages 17–21, 2006.
Cited on page 4, 44

F. Schwenker, S. Scherer, M. Schmidt, M. Schels, and M. Glodek. Multiple classifier systems for the recognition of human emotions. In *Proceedings of the International Workshop on Multiple Classifier Systems (MCS)*, volume 5997 of *Lecture Notes in Computer Science (LNCS)*, pages 315–324. Springer, 2010.
Cited on page 44

F. Schwenker, F. Markus, M. Glodek, M. Kächele, S. Meudt, M. Schels, and M. Schmidt. A new multi-class fuzzy support vector machine algorithm. In N. El Gayar, F. Schwenker, and C. Suen, editors, *Proceedings of the International Workshop on Artificial Neural Networks in Pattern Recognition (ANNPR)*, volume 8774 of *Lecture Notes in Computer Science (LNCS)*, pages 153–164. Springer, 2014. doi: 10.1007/978-3-319-11656-3_14.
Cited on page 52, 200

T. Serre, L. Wolf, and T. Poggio. Object recognition with features inspired by visual cortex. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 994–1000, 2005.
Cited on page 109

G. Shafer. *A Mathematical Theory of Evidence*, volume 1. Princeton University Press, 1976.
Cited on page 16

G. Shafer. The combination of evidence. *International Journal of Intelligent Systems*, 1(3): 155–179, 1986.
Cited on page 13

Y. Sheikh, M. Sheikh, and M. Shah. Exploring the space of a human action. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 144–149. IEEE, 2005.
Cited on page 8

T. Shinozaki and T. Kawahara. GMM and HMM training by aggregated EM algorithm with increased ensemble sizes for robust parameter estimation. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4405–4408. IEEE, 2008.
Cited on page 56, 57

J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013.
Cited on page 9

I. Siegert, M. Glodek, and G. Krell. Using speaker group dependent modelling to improve fusion of fragmentary classifier decisions. In *Proceedings of the International IEEE Conference on Cybernetics (CYBCONF)*, pages 132–137. IEEE, 2013. doi: 10.1109/CYBConf.2013.6617458.
Cited on page 75

P.-M. Strauß, H. Hoffmann, W. Minker, H. Neumann, G. Palm, S. Scherer, H. Traue, and U. Weidenbacher. The PIT corpus of German multi-party dialogues. In *Proceedings of the International Language Resources and Evaluation Conference (LREC)*. European Language Resources Association (ELRA), 2008.
Cited on page 7, 8

C. Sutton and A. McCallum. An introduction to conditional random fields for relational learning. *Introduction to statistical relational learning*, pages 93–128, 2007.
Cited on page 20, 22, 23, 39

M. Swain and D. Ballard. Color indexing. *Journal of Computer Vision*, 7(1):11–32, 1991.
Cited on page 114

M. Szczot, O. Löhlein, and G. Palm. Dempster-Shafer fusion of context sources for pedestrian recognition. In T. Denoeux and M.-H. Masson, editors, *Belief Functions: Theory and Applications*, volume 164 of *Advances in Intelligent and Soft Computing*, pages 319–326. Springer, 2012.
Cited on page 4

S. Theodoridis and K. Koutroumbas. *Pattern Recognition.* Academic Press, 3rd edition, 2006.
Cited on page 46

C. Thiel. *Multiple Classifier Systems Incorporating Uncertainty.* Verlag Dr. Hut, 2010. ISBN 978-3-86853-675-1.
Cited on page 4, 13, 44, 46

C. Thiel, S. Scherer, and F. Schwenker. Fuzzy-input fuzzy-output one-against-all support vector machines. In *Proceedings of the International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES), Part III*, volume 4694 of *Lecture Notes in Computer Science (LNCS)*, pages 156–165. Springer, 2007.
Cited on page 49, 51, 52

Y. Tian, T. Kanade, and J. Cohn. Recognizing action units for facial expression analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):97–115, 2001.
Cited on page 7

D. Tran and A. Sorokin. Human activity recognition with metric learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 548–561. Springer, 2008.
Cited on page 9

S. Tran and L. Davis. Event modeling and recognition using Markov logic networks. In D. Forsyth, P. Torr, and A. Zisserman, editors, *Proceedings of the European Conference on Computer Vision (ECCV): Part II*, volume 5303 of *Lecture Notes in Computer Science (LNCS)*, pages 610–623. Springer Berlin Heidelberg, 2008. doi: 10.1007/978-3-540-88688-4_45.
Cited on page 11

E. Trentin and E. Di Iorio. Unbiased SVM density estimation with application to graphical pattern recognition. In J. de Sá, L. Alexandre, W. Duch, and D. Mandic, editors, *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, volume 4669 of *Lecture Notes in Computer Science (LNCS)*, pages 271–280. Springer, 2007a. doi: 10.1007/978-3-540-74695-9_28.
Cited on page 29

E. Trentin and E. Di Iorio. A simple and effective neural model for the classification of structured patterns. In B. Apolloni, R. Howlett, and L. Jain, editors, *Proceedings of the International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES)*, volume 4692 of *Lecture Notes in Computer Science (LNCS)*, pages 9–16. Springer, 2007b. doi: 10.1007/978-3-540-74819-9_2.
Cited on page 29

E. Trentin and E. Di Iorio. Classification of molecular structures made easy. In *IEEE International Joint Conference on Neural Networks (IJCNN)*, pages 3241–3246, 2008. doi: 10.1109/IJCNN.2008.4634258.
Cited on page 185, 198

E. Trentin and E. Di Iorio. Classification of graphical data made easy. *Neurocomputing*, 73 (1):204–212, 2009.
Cited on page 29

P. Turaga, R. Chellappa, V. Subrahmanian, and O. Udrea. Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*,

18(11):1473–1488, 2008.
Cited on page 8, 10

M. F. Valstar, H. Gunes, and M. Pantic. How to distinguish posed from spontaneous smiles using geometric features. In *Proceedings of the International Conference on Multimodal Interfaces (ICMI)*, pages 38–45. ACM, 2007.
Cited on page 44

V. Vapnik. *The Nature of Statistical Learning Theory.* Springer, 1999.
Cited on page 15, 44, 49

A. Vinciarelli, M. Pantic, H. Bourlard, and A. Pentland. Social signal processing: State-of-the-art and future perspectives of an emerging domain. In *Proceedings of the International ACM Conference on Multimedia (MM)*, pages 1061–1070. ACM New York, NY, 2008. ISBN 978-1-60558-303-7. doi: 10.1145/1459359.1459573.
Cited on page 8

P. Viola and M. Jones. Robust real-time object detection. *International Journal of Computer Vision*, 57(2):137–154, 2002.
Cited on page 105

J. Wagner, T. Vogt, and E. André. A systematic comparison of different HMM designs for emotion recognition from acted and spontaneous speech. *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 114–125, 2007.
Cited on page 7

S. Walter, S. Scherer, M. Schels, M. Glodek, D. Hrabal, M. Schmidt, R. Böck, K. Limbrecht, H. Traue, and F. Schwenker. Multimodal emotion classification in naturalistic user behavior. In J. Jacko, editor, *Proceedings of the International Conference on Human-Computer Interaction: Towards Mobile and Intelligent Interaction Environments*, volume 6763 of *Lecture Notes in Computer Science (LNCS)*, pages 603–611. Springer, 2011.
Cited on page 8

C. Wendt, M. Popp, M. Karg, and K. Kuhnlenz. Physiology and HRI: Recognition of over-and underchallenge. In *Proceedings of the IEEE Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 448–452. IEEE, 2008.
Cited on page 8

J. Weston and C. Watkin. Multi-class support vector machines. Technical Report CSD-TR-98-04, Department of Computer Science, Royal Holloway University of London, Egham, Surrey TW20 0EX, England, May 1998.
Cited on page 52

M. Wöllmer, B. Schuller, F. Eyben, and G. Rigoll. Combining long short-term memory and dynamic Bayesian networks for incremental emotion-sensitive artificial listening. *IEEE Journal of Selected Topics in Signal Processing*, 4(5):867–881, 2010.
Cited on page 8

S. Wrede, M. Hanheide, C. Bauckhage, and G. Sagerer. An active memory as a model for information fusion. In *Proc. Int. Conf. on Information Fusion*, volume 1, pages 198–205. Citeseer, 2004.
Cited on page 10

Z. Zeng, M. Pantic, G. I. Roisman, and T. S. Huang. A survey of affect recognition methods: Audio, visual, and spontaneous expressions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(1):39–58, 2009. doi: 10.1109/TPAMI.2008.52.
Cited on page 7

F. Zheng, G. Zhang, and Z. Song. Comparison of different implementations of MFCC. *Journal of Computer Science and Technology*, 16(6):582–589, 2001.
Cited on page 108

Main Contributions

[1] Glodek, M., Layher, G., Heilemann, F., Gawrilowicz, F., Palm, G., Schwenker, F., and Neumann, H. UUlmMAD—a human action recognition dataset for ground-truth evaluation and investigation of view invariances. In Schwenker, F., Scherer, S., and Morency, L.-P., editors, *Proceedings of the International Workshop on Multimodal Pattern Recognition of Social Signals in Human-Computer-Interaction (MPRSS)*, volume 8869 of *Lecture Notes in Computer Science (LNCS)*, pages 77–91. Springer International Publishing, 2015a. ISBN 978-3-319-14898-4. doi: 10.1007/978-3-319-14899-1_8.
Cited on page 98, 198

[2] Glodek, M., Honold, F., Geier, T., Krell, G., Nothdurft, F., Reuter, S., Schüssel, F., Hörnle, T., Dietmayer, K., Minker, W., Biundo, S., Weber, M., Palm, G., and Schwenker, F. Fusion paradigms in cognitive technical systems for human-computer interaction. *Neurocomputing*, 161(2015):17–37, 2015b. doi: 10.1016/j.neucom.2015.01.076.
Cited on page 2, 3, 202

[3] Glodek, M., Geier, T., Biundo, S., and Palm, G. A layered architecture for probabilistic complex pattern recognition to detect user preferences. *Journal of Biologically Inspired Cognitive Architectures*, 9:46–56, 2014a.
Cited on page 66, 84, 85, 86, 179, 197, 199, 200

[4] Glodek, M., Schels, M., Schwenker, F., and Palm, G. Combination of sequential class distributions from multiple channels using Markov fusion networks. *Journal on Multimodal User Interfaces (JMUI)*, 8(3):257–272, 2014b. doi: 10.1007/s12193-014-0149-0.
Cited on page 75, 197

[5] Glodek, M., Schels, M., and Schwenker, F. Inequality-constraint multi-class fuzzy-in fuzzy-out support vector machines. In *Proceedings of the International Workshop on Statistical Computing*, number 4 in Ulmer Informatik Bericht, pages 31–32. University of Ulm, 2014c.
Cited on page 89, 200

[6] Glodek, M., Geier, T., Biundo, S., Schwenker, F., and Palm, G. Recognizing user preferences based on layered activity recognition and first-order logic. In *Proceedings of the International IEEE Conference on Tools with Artificial Intelligence (ICTAI)*, pages 648–653. IEEE, 2013a.
Cited on page 85, 199

[7] Glodek, M., Trentin, E., Schwenker, F., and Palm, G. Hidden Markov models with graph densities for action recognition. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pages 964–969. IEEE, 2013b.
Cited on page 73, 198

[8] Glodek, M., Reuter, S., Schels, M., Dietmayer, K., and Schwenker, F. Kalman filter based classifier fusion for affective state recognition. In Zhou, Z.-H., Roli, F., and Kittler, J., editors, *Multiple Classifier Systems (MCS)*, volume 7872 of *Lecture Notes in Computer Science (LNCS)*, pages 85–94. Springer Berlin Heidelberg, 2013c. doi: 10.1007/978-3-642-38067-9_8.
Cited on page 81, 197

[9] Glodek, M., Schels, M., and Schwenker, F. Ensemble Gaussian mixture models for probability density estimation. *Computational Statistics*, 28(1):127–138, 2013d. doi: 10.1007/s00180-012-0374-5.
Cited on page 56, 123, 200

[10] Glodek, M., Schwenker, F., and Palm, G. Detecting actions by integrating sequential symbolic and sub-symbolic information in human activity recognition. In Perner, P., editor, *Proceedings of the International Conference on Machine Learning and Data Mining (MLDM)*, volume 7376 of *Lecture Notes in Computer Science (LNCS)*, pages 394–404. Springer Berlin Heidelberg, 2012a. ISBN 978-3-642-31536-7. doi: 10.1007/978-3-642-31537-4_31.
Cited on page 85, 199

[11] Glodek, M., Layher, G., Schwenker, F., and Palm, G. Recognizing human activities using a layered Markov architecture. In Villa, A. E., Duch, W., Érdi, P., Masulli, F., and Palm, G., editors, *Proceedings of the International Conference on Artificial Neural Networks and Machine Learning (ICANN)*, volume 7552 of *Lecture Notes in Computer Science (LNCS)*, pages 677–684. Springer Berlin Heidelberg, 2012b. ISBN 978-3-642-33268-5. doi: 10.1007/978-3-642-33269-2_85.
Cited on page 85, 199

[12] Glodek, M., Schels, M., Palm, G., and Schwenker, F. Multiple classifier combination using reject options and Markov fusion networks. In *Proceedings of the International ACM Conference on Multimodal Interaction (ICMI)*, pages 465–472. ACM New York, NY, 2012c. ISBN 978-1-4503-1467-1. doi: 10.1145/2388676.2388778.
Cited on page 75, 110, 167, 190, 197

[13] Glodek, M., Schels, M., Palm, G., and Schwenker, F. Multi-modal fusion based on classifiers using reject options and Markov fusion networks. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, pages 1084–1087. IEEE, 2012d.
Cited on page 75, 197

[14] Glodek, M., Bigalke, L., Palm, G., and Schwenker, F. Recognizing human activities using a layered HMM architecture. In Hammer, B. and Villmann, T., editors, *Machine Learning Reports*, number 5 in New Challenges in Neural Computation (NC^2), pages 38–41. Springer Berlin Heidelberg, 2011a. doi: 10.1007/978-3-642-33269-2_85.
Cited on page 85

[15] Glodek, M., Bigalke, L., Schels, M., and Schwenker, F. Incorporating uncertainty in a layered HMM architecture for human activity recognition. In *Proceedings of the Joint Workshop on Human Gesture and Behavior Understanding (J-HGBU)*, pages 33–34. ACM, 2011b. doi: 10.1145/2072572.2072584.
Cited on page 84, 85

[16] Glodek, M., Tschechne, S., Layher, G., Schels, M., Brosch, T., Scherer, S., Kächele, M., Schmidt, M., Neumann, H., Palm, G., and Schwenker, F. Multiple classifier systems for the classification of audio-visual emotional states. In D'Mello, S., Graesser, A., Schuller, B., and Martin, J.-C., editors, *Proceedings of the International Conference on Affective Computing and Intelligent Interaction (ACII)*, volume 6975 of *Lecture Notes in Computer Science (LNCS)*, pages 359–368. Springer Berlin Heidelberg, 2011c. ISBN 978-3-642-24570-1. doi: 10.1007/978-3-642-24571-8_47.
Cited on page 109, 152

[17] Glodek, M., Scherer, S., and Schwenker, F. Conditioned hidden Markov model fusion for multimodal classification. In *Proceedings of the Annual Conference of the International Speech Communication Association (ISCA), Interspeech*, pages 2269–2272. ISCA, ISCA, 2011d.
Cited on page 141, 146

[18] Glodek, M., Schels, M., and Schwenker, F. Artificial neural networks trained by ensembles of Gaussian mixture models. In Kestler, H. A., Binder, H., Lausen, B., Klenk, H.-P., Schmid, M., and Leisch, F., editors, *Proceedings of the International Workshop on Statistical Computing*, number 5 in Ulmer Informatik Bericht, page 7.

University of Ulm, 2010.
Cited on page 56, 200

[19] Kächele, M., Schels, M., Meudt, S., Kessler, V., Glodek, M., Thiam, P., Tschechne, S., Palm, G., and Schwenker, F. On annotation and evaluation of multi-modal corpora in affective human-computer interaction. In Böck, R., Bonin, F., Campbell, N., and Poppe, R., editors, *Proceedings of Multimodal Analyses enabling Artificial Agents in Human-Machine Interaction*, volume 8757 of *Lecture Notes in Computer Science (LNCS)*, pages 35–44. Springer, 2014a. doi: 10.1007/978-3-319-15557-9_4.
Cited on page 105, 190

[20] Kächele, M., Glodek, M., Zharkov, D., Meudt, S., and Schwenker, F. Fusion of audio-visual features using hierarchical classifier systems for the recognition of affective states and the state of depression. In De Marsico, M., Tabbone, A., and Fred, A., editors, *Proceedings of the International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, pages 671–678. SciTePress, 2014b.
Cited on page 81, 166, 198

[21] Krell, G., Glodek, M., Panning, A., Siegert, I., Michaelis, B., Wendemuth, A., and Schwenker, F. Fusion of fragmentary classifier decisions for affective state recognition. In Schwenker, F., Scherer, S., and Morency, L.-P., editors, *Proceedings of the Workshop on Multimodal Pattern Recognition of Social Signals in Human-Computer-Interaction (MPRSS)*, volume 7742 of *Lecture Notes in Computer Science (LNCS)*, pages 116–130. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-37080-9. doi: 10.1007/978-3-642-37081-6_13.
Cited on page 75

[22] Meudt, S., Glodek, M., Schels, M., and Schwenker, F. Multi-view video based tracking and audio-visual identification of persons in a human-computer-interaction scenario. In *Proceedings of the IEEE International Conference on Cybernetics (CYBCONF)*, pages 116–121, June 2013. doi: 10.1109/CYBConf.2013.6617454.
Cited on page 46

[23] Palm, G. and Glodek, M. Towards emotion recognition in human computer interaction. In Esposito, A., Squartini, S., and Palm, G., editors, *Neural Nets and Surroundings*, volume 19 of *Smart Innovation, Systems and Technologies*, pages 323–336. Springer Berlin Heidelberg, 2013. doi: 10.1007/978-3-642-35467-0_32.
Cited on page 7, 75

[24] Scherer, S., Glodek, M., Layher, G., Schels, M., Schmidt, M., Brosch, T., Tschechne, S., Schwenker, F., Neumann, H., and Palm, G. A generic framework for the inference of user states in human computer interaction: How patterns of low level behavioral cues support complex user states in HCI. *Journal on Multimodal User Interfaces (JMUI)*, 6(3–4):117–141, 2012a. doi: 10.1007/s12193-012-0093-9.
Cited on page 1, 8, 200

[25] Scherer, S., Glodek, M., Schwenker, F., Campbell, N., and Palm, G. Spotting laughter in natural multiparty conversations: A comparison of automatic online and offline approaches using audiovisual data. *ACM Transactions on Interactive Intelligent Systems (TiiS) (Special Issue on Affective Interaction in Natural Environments)*, 2(1): 4:1–4:31, 2012b. doi: 10.1145/2133366.2133370.
Cited on page 106, 141, 143, 146, 175, 200, 257

[26] Schels, M., Kächele, M., Glodek, M., Hrabal, D., Walter, S., and Schwenker, F. Using unlabeled data to improve classification of emotional states in human computer interaction. *Journal on Multimodal User Interfaces (JMUI) (Special Issue on From Multimodal Analysis to Real-Time Interactions with Virtual Agents)*, 8(1):5–16, 2014. doi: 10.1007/s12193-013-0133-0.
Cited on page 47, 200

[27] Schels, M., Glodek, M., Meudt, S., Scherer, S., Schmidt, M., Layher, G., Tschechne, S., Brosch, T., Hrabal, D., Walter, S., Palm, G., Neumann, H., Traue, H., and Schwenker, F. Multi-modal classifier-fusion for the recognition of emotions. In *Coverbal synchrony in Human-Machine Interaction*, pages 73–97. CRC Press, 2013a.
Cited on page 44, 110

[28] Schels, M., Glodek, M., Palm, G., and Schwenker, F. Revisiting AVEC 2011—an information fusion architecture. In Esposito, A., Squartini, S., Palm, G., Apolloni, B., Bassis, S., Esposito, A., and Morabito, F. C., editors, *Neural Nets and Surroundings*, volume 19 of *Smart Innovation, Systems and Technologies*, pages 385–393. Springer Berlin Heidelberg, 2013b. ISBN 978-3-642-35466-3. doi: 10.1007/978-3-642-35467-0_38.
Cited on page 75

[29] Schels, M., Glodek, M., Meudt, S., Schmidt, M., Hrabal, D., Böck, R., Walter, S., and Schwenker, F. Multi-modal classifier-fusion for the classification of emotional states in WOZ scenarios. In Ji, Y. G., editor, *Advances in Affective and Pleasurable Design*, number 22 in Advances in Human Factors and Ergonomics Series, pages 644–653. CRC Press, 2012. ISBN 978-1439871188. doi: 10.1201/b12525-78.
Cited on page 105

[30] Schels, M., Scherer, S., Glodek, M., Kestler, H., Palm, G., and Schwenker, F. On the discovery of events in EEG data utilizing information fusion. *Computational Statistics*, 28(1):5–18, 2013. doi: 10.1007/s00180-011-0292-y.
Cited on page 44

[31] Schels, M., Scherer, S., Glodek, M., Kestler, H. A., Schwenker, F., and Palm, G. A hybrid information fusion approach to discover events in EEG data. In Kestler, H. A., Binder, H., Lausen, B., Klenk, H.-P., Schmid, M., and Leisch, F., editors, *Proceedings of the International Workshop on Statistical Computing*, number 5 in Ulmer Informatik Bericht, pages 34–36. University of Ulm, 2010.
Cited on page 44

[32] Siegert, I., Glodek, M., and Krell, G. Using speaker group dependent modelling to improve fusion of fragmentary classifier decisions. In *Proceedings of the International IEEE Conference on Cybernetics (CYBCONF)*, pages 132–137. IEEE, 2013. doi: 10.1109/CYBConf.2013.6617458.
Cited on page 75

[33] Schwenker, F., Markus, F., Glodek, M., Kächele, M., Meudt, S., Schels, M., and Schmidt, M. A new multi-class fuzzy support vector machine algorithm. In El Gayar, N., Schwenker, F., and Suen, C., editors, *Proceedings of the International Workshop on Artificial Neural Networks in Pattern Recognition (ANNPR)*, volume 8774 of *Lecture Notes in Computer Science (LNCS)*, pages 153–164. Springer, 2014. doi: 10.1007/978-3-319-11656-3_14.
Cited on page 52, 200

[34] Schwenker, F., Scherer, S., Schmidt, M., Schels, M., and Glodek, M. Multiple classifier systems for the recognition of human emotions. In *Proceedings of the International Workshop on Multiple Classifier Systems (MCS)*, volume 5997 of *Lecture Notes in Computer Science (LNCS)*, pages 315–324. Springer, 2010.
Cited on page 44

[35] Walter, S., Scherer, S., Schels, M., Glodek, M., Hrabal, D., Schmidt, M., Böck, R., Limbrecht, K., Traue, H., and Schwenker, F. Multimodal emotion classification in naturalistic user behavior. In Jacko, J., editor, *Proceedings of the International Conference on Human-Computer Interaction: Towards Mobile and Intelligent Interaction Environments*, volume 6763 of *Lecture Notes in Computer Science (LNCS)*, pages 603–611. Springer, 2011.
Cited on page 8

Index of Figures

All figures, except those how were adapted from the specified sources, are taken from own publications. The author likes to acknowledge the creative effort of all figures contributed by others.

Figure 2.1 adapted from Bishop (2006), page 13.
Figure 2.12 adapted from Oliver et al. (2004).

Figure 4.7 was created by Georg Layher.
Figure 4.6 was created by Georg Layher.
Figure 4.12 taken from Scherer et al. (2012b) with kind permission of Stefan Scherer.
Figure 4.13 was created by Georg Layher.
Figure 4.14 was created by Martin Schels.
Figure 4.18 was created by Georg Layher.

Table 5.3 was created by Felix Heilemann.
Figure 5.5 adapted from Scherer et al. (2012b).
Figure 5.17 was created by Thomas Geier.

Index of Tables

Index