TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Mensch-Maschine-Kommunikation

Context-Sensitive Machine Learning for Intelligent Human Behavior Analysis

Martin Wöllmer

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr.-Ing. K. Diepold
Prüfer der Dissertation:	1. Priv.-Doz. Dr.-Ing. habil. B. W. Schuller
	2. Univ.-Prof. Dr. E. André (Universität Augsburg)

Die Dissertation wurde am 16.08.2012 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 21.12.2012 angenommen.

Acknowledgment

I would like to thank my supervisor Priv.-Doz. Björn Schuller for his excellent inspiration, guidance, and support. I am also grateful to Prof. Elisabeth André for reviewing this thesis and to Prof. Gerhard Rigoll for having given me the opportunity to work at his institute. For the good cooperation I would like to thank my colleagues at the Institute for Human-Machine Communication, especially Florian Eyben, Felix Weninger, Erik Marchi, Jürgen Geiger, Moritz Kaiser, Peter Brand, and Heiner Hundhammer. I would also like to thank Alex Graves, Angeliki Metallinou, Shrikanth Narayanan, Jort Gemmeke, Emanuele Principi, Rudy Rotili, Stefano Squartini, Marc Schröder, Roddy Cowie, Ellen Douglas-Cowie, Stefan Steidl, Anton Batliner, Christoph Blaschke, Dejan Arsić, Jasha Droppo, Joseph Keshet, Sun Yang, and Thomas Schindl for their technical contributions, assistance, and collaboration. A very special thanks goes to Marc Al-Hames for triggering my passion for pattern recognition. Most of all, I would like to thank my parents Irmgard and Leonhard Wöllmer and my lovely wife Anne for her constant encouragement, support, and love.

This research was supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 211486 (SEMAINE) and by the Federal Republic of Germany through the German Research Foundation (DFG) under grant no. SCHU 2508/4-1.

Abstract

Intelligent automatic human behavior analysis is an essential precondition for conversational agent systems that aim to enable natural, intuitive, emotionally sensitive, and enjoyable human-computer interaction. This thesis focuses on automatic verbal and non-verbal behavior analysis and introduces novel speech processing and machine learning architectures that are capable of inferring the spoken content as well as the user's affective state from the speech and video signal. The aim is to advance the state-of-the-art in automatic speech and emotion recognition via suited Graphical Model structures and context-sensitive neural network architectures. As Long Short-Term Memory (LSTM) recurrent neural networks are known to be well-suited for modeling and exploiting an arbitrary amount of self-learned temporal context for sequence labeling and pattern recognition, this thesis illustrates how LSTM modeling can be applied for linguistic and affective information extraction from speech. Extensive experiments concentrating on naturalistic, spontaneous, and affective interactions show that the proposed LSTM-based recognition frameworks prevail over current state-of-the-art techniques for speech and emotion recognition.

Zusammenfassung

Intelligente automatische Analyse menschlichen Verhaltens ist eine essenzielle Voraussetzung für Dialogsysteme, welche eine natürliche, intuitive, emotionssensitive und angenehme Mensch-Maschine-Interaktion ermöglichen sollen. Diese Arbeit beschäftigt sich mit der automatischen verbalen und nicht-verbalen Verhaltensanalyse und stellt neue Architekturen zur Sprachverarbeitung und zum maschinellen Lernen vor, welche die Extraktion des gesprochenen Inhalts sowie des emotionalen Zustands des Nutzers aus dem Sprach- und Videosignal ermöglichen. Ziel ist es, den Stand der Technik im Bereich der automatischen Sprach- und Emotionserkennung durch geeignete graphische Modellstrukturen und kontextsensitive neuronale Netzwerk Architekturen voranzutreiben. Da Long Short-Term Memory (LSTM) rekurrente neuronale Netze zur Sequenztranskription und Mustererkennung bekanntermaßen gut dazu geeignet sind, ein beliebiges Maß an selbstgelerntem zeitlichem Kontext zu modellieren und auszunutzen, zeigt diese Arbeit, wie LSTM-Modellierung dazu verwendet werden kann, linguistische und affektive Information aus der Sprache zu extrahieren. Umfangreiche Experimente, welche sich auf naturalistische, spontane, und emotionale Interaktionen konzentrieren, zeigen, dass die vorgestellten LSTM-basierten Erkennungssysteme den aktuellen Stand der Technik im Bereich der Sprach- und Emotionserkennung übertreffen.

Contents

1

Introduction

Despite recent advances in the design and implementation of modern interfaces for human-machine communication, there still exists a large discrepancy between the efficiency and versatility of interhuman communication and the way we communicate and interact with computers. Humans are able to to express, perceive, process, and memorize a rich set of behavioral cues that enable natural and multimodal communication and social information exchange via speech, non-linguistic vocalizations, facial expressions, and gestures. By contrast, interaction with computers has long been restricted to rather unnatural input and output modalities such as keyboard or mouse, and abstract text or sound output, respectively. This mismatch has triggered massive research in alternative and more human-like methods for human-machine communication, including automatic speech recognition (ASR) [183], handwriting recognition [92], facial expression recognition [41], natural language understanding [268], dialog management [156], speech synthesis [208], and animated virtual agents [20]. However, today's virtual agent systems supporting speech- and video-based in- and output are still far from being perceived as natural, efficient, and comparable to humans due to limitations in the aforementioned system capabilities. This thesis aims to advance the state-of-the-art in the first component of a dialog system's processing chain: automatic human behavior analysis. Mainly focusing on the processing of the user's speech signal, we subdivide human behavior analysis into verbal and non-verbal behavior analysis. While verbal behavior analysis refers to the extraction of the spoken content encoded in the speech signal (automatic speech recognition and keyword detection), the term 'non-verbal behavior analysis' subsumes the recognition of information beyond the spoken content and includes the detection of various paralinguistic cues such as the user's emotional state, level of interest, etc. Recognizing and considering these non-verbal cues and affective user states in a conversational agent framework was shown to be highly relevant for increasing the naturalness, acceptance, joy of use, and efficiency of human-computer interaction as this allows, e. g., virtual agents to react to the user's emotion in an appropriate way [49].

The aim of the SEMAINE project [206] is to build a dialog system that focuses exactly on these non-verbal aspects of communication to enable emotionally sensitive, human-like conversation about arbitrary topics. A central goal is to integrate emotional and situational awareness into virtual agents in order to establish a basis for future intelligent dialog systems that use the developed principles and components for affective computing within task-oriented agent systems. The realization of robust recognition systems for intelligent verbal and non-verbal behavior analysis as needed in the SEMAINE framework is the central aspect dealt with in this thesis and involves various research disciplines such as speech feature extraction, speech and feature enhancement, machine learning, pattern recognition, multimodal data fusion, and affective computing.

A successful integration of human behavior analysis technology into conversational agent systems like the SEMAINE system implies coping with several challenges that accompany the application of speech signal processing and interpretation in real-life scenarios: Even though ASR can reach very high accuracies for the recognition of well articulated, read speech in well-defined clean acoustic conditions, the influence of background noise, reverberation, as well as conversational, disfluent, and emotional speaking styles that are to be expected in natural dialog situations, are known to heavily downgrade recognition performance and demand for novel, robust recognition engines that go beyond standard technology applied, e. g., in dictation software. Similar challenges hold for the field of non-verbal human behavior analysis: First studies in speech-based emotion recognition concentrated on the classification of pre-segmented spoken utterances containing prototypical and acted emotions that are easy to characterize with current pattern recognition techniques and thus lead to high recognition accuracies. Yet, realistic interactions between humans or between humans and computers tend to evoke ambiguous, non-prototypical spontaneous emotions that are hard to distinguish by state-of-the-art approaches for affect recognition.

In this thesis, all of these challenges are addressed by exploring novel concepts for intelligent human behavior analysis via appropriate machine learning and signal processing techniques. As speech is a dynamic process, we mainly consider dynamic classification methods that capture the evolution of speech features over time. We investigate ASR model architectures that deviate from the commonly used Hidden Markov Model (HMM) framework by deriving new Graphical Model (GM) structures that allow for a reliable detection of keywords in running speech and for the integration of multiple complementary feature streams. A central topic addressed and advanced in this thesis is the efficient modeling and incorporation of temporal context information for improved human behavior analysis. Thus, we focus on *context-sensitive* machine learning that models speech feature frames or spoken utterances in the context of neighboring frames and utterances, respectively. State-of-the-art speech and emotion recognition systems already consider contextual information by various methods like Markov modeling of feature vectors, computa-

tion of delta features, application of triphones, language modeling, calculation of statistical functionals of low-level features, or recurrent connections in neural networks, aiming to model state, phoneme, and word transitions, co-articulation effects in human speech, or speech feature dynamics encoding information about the speaker's emotional state. However, all these techniques have their drawbacks and have little in common with the way humans memorize and exploit context information over time.

In [111], a promising approach to model temporal long-range context via so-called Long Short-Term Memory (LSTM) neural networks has been proposed. LSTM networks consist of recurrently connected *memory blocks* that replace the conventional neurons in the network's hidden layer. They provide access to an arbitrary amount of self-learned long-range context, overcoming the well-known *vanishing gradient problem* which limits the amount of context a conventional recurrent neural network can model. Motivated by excellent results reported in first studies that apply LSTM-modeling for speech-based recognition tasks [93], this thesis proposes and evaluates various techniques to incorporate the LSTM architecture into systems for verbal and non-verbal behavior analysis.

The aims of this thesis can be summarized as follows:

- to develop, optimize, evaluate, and compare novel model architectures for improved recognition of verbal and non-verbal cues in speech signals via efficient context-sensitive machine learning based on suited Graphical Model and neural network structures;

- to create recognition systems that are robust with respect to signal distortions and can be applied in real-life scenarios involving realistic, non-prototypical, and spontaneous speaking styles and emotions;

- to explore the benefit of integrating multiple information sources and modalities into the behavior recognition process, including acoustic, linguistic, and visual cues;

- to accomplish a better understanding and assessment of the role that appropriate long-range context modeling plays in human behavior analysis and to establish innovative solutions proposed by the machine learning community within the speech and affective computing communities;

- to demonstrate that the developed solutions tailored for affective computing can be effectively transferred to other pattern recognition and sequence modeling tasks and domains.

The following chapters outline in what respect these goals are addressed and achieved:

Chapter 2 is devoted to the theoretical background that can be seen as a basis for the development of intelligent human behavior analysis techniques in the following chapters. We start by looking at the architecture and specification of the emotionally sensitive SEMAINE dialogue system which is the major motivation for the recognition modules developed in this thesis. Next, as speech is the primarily considered modality for human behavior analysis as investigated in this thesis, the most relevant prosodic, spectral, and voice quality features are discussed. Finally, the basic concept of various pattern recognition, sequence modeling, classification, and fusion algorithms as employed in the following chapters are explained.

Chapter 3 concentrates on *verbal* behavior analysis and proposes methods for vocabulary independent keyword detection, conversational speech recognition, and enhanced noise robustness. The focus will be on the development and evaluation of novel DBN- and LSTM-based model architectures that deviate from standard HMM approaches. Various methods for increasing the noise robustness of speech recognition are introduced and all systems and methods are evaluated under unified and challenging experimental conditions.

Chapter 4 focuses on *non-verbal* behavior analysis, meaning the recognition of emotions or other user states such as the speaker's level of interest. First, techniques for speech-based affect recognition are proposed, including the modeling of acoustic and linguistic cues. Furthermore, we will study the effect reverberation has on emotion recognition performance. The second part of Chapter 4 deals with audio-visual approaches for assessing human affect and contains an analysis of the *sequential Jacobian* in order to determine the amount of context that is exploited by LSTM networks for emotion recognition. All experiments consider natural, spontaneous, and non-prototypical emotions and reflect recognition performances in realistic conditions.

Chapter 5 shows how methods developed for (speech-based) affective computing can be transferred to other pattern recognition disciplines. We consider the task of driving behavior analysis by attempting to detect driver distraction from head tracking and driving data. Similar to the emotion recognition frameworks developed in Chapter 4, a large set of statistical functionals is computed from informative low-level signals and subsequently modeled and classified via Long Short-Term Memory neural networks.

Chapter 6 summarizes the thesis by providing an overview over the developed recognition engines, the addressed challenges, the experimental results, and possible future work.

2

Theoretical Background

This chapter outlines relevant theoretical background that serves as a basis for the development of the intelligent human behavior analysis systems discussed in Chapters 3 to 5. We start with a brief motivation of natural human behavior analysis by introducing the SEMAINE system [206] – a multi-modal conversational agent framework that takes into account the user's verbal, non-verbal, and visual behavioral cues to enable natural and emotion-sensitive human-machine communication (Section 2.1). Intelligent computer interfaces like the SEMAINE system can be seen as one of the main use cases for the speech and affect recognition techniques developed in this thesis and thus define the requirements these techniques have to meet, including the processing and recognition of real-life, non-prototypical, spontaneous human behavior, the efficient exploitation of contextual information, and noise robustness – just to name a few. As our main focus will be on the recognition of verbal and non-verbal cues from the speech signal, Section 2.2 reviews a set of acoustic features commonly used for speech and emotion recognition. Finally, Section 2.3 provides an overview over the main machine learning techniques applied and advanced in this thesis.

2.1 The SEMAINE System

One of the key capabilities of human-computer interaction systems designed for natural, intuitive, and human-like communication is the sensitivity to emotion-related and non-verbal cues [6, 49]. Thus, the aim of the SEMAINE project[1] is to focus on exactly these kinds of 'social skills' in order to advance the state-of-the-art in affect-sensitive conversational agent systems and to pioneer dialog systems that show a certain degree of 'social competence'. The so-called Sensitive Artificial Listeners (SAL) [206] developed in the project are virtual agent characters that chat with the user about arbitrary topics without having to fulfill a certain task, i. e., without having to face the constraints of typical task-oriented dialog systems, such as

[1] www.semaine-project.eu

information kiosks or ordering systems. Both, input and output components are multi-modal, involving speech, facial expressions, and head movements. The SAL scenario is designed to evoke and model typical realistic 'everyday' emotions [50] rather than exaggerated 'basic' affective states that are unlikely to occur in natural conversations.

This implies a number of challenges comprising both, computational perception and analysis of user behavior as well as automatic behavior generation and synthesis. The system has to recognize the words spoken by the user (verbal behavior) and the prosody with which they are spoken (non-verbal behavior). Video-based recognition modules have to capture behavioral cues such as head movements and facial expressions [98]. Further, the virtual agent has to show appropriate listener behavior, such as audible and visual *backchannels* [304] in the form of head nods, smiles, or short vocalizations, while the user is speaking. As soon as the system has decided to 'take the turn' [200], it has to produce an utterance that fits the dialog context and encourages the user to continue the conversation.

The challenges and requirements that arise when incorporating emotional intelligence into human-computer interaction are quite different to the specifications of first dialogue systems built in the nineties, which primarily served as human language interfaces to information [87]. More advanced dialogues can be observed during interaction with conversational dialog systems incorporating system goals [3]. One example for a *multi-modal* dialog system including visual information and non-verbal behavior analysis is the SmartKom system [254] – an information kiosk with initial support for speech- and vision-based emotion recognition. Unlike purely speech-based dialog systems, Embodied Conversational Agents (ECA) [5, 35] take the appearance of an animated human-like face and are thus able to show expressive facial behavior. As shown in [170], such expressive behavior of the ECA can positively affect dialog success if displayed in an appropriate way. Dialog systems that analyze the *user's* expressive behavior usually focus on facial expressions and voice and find application for example in emotion-aware voice portals [30] that detect anger in the customer's speech. Especially in ECA systems, the notion of 'social presence' is essential for the perceived naturalness of the conversation. Social presence can be achieved by psychological involvement, behavioral engagement, showing responsiveness, and taking into account verbal and non-verbal user behavior for dialogue planning [23, 34]. Even though these aspects have partially been considered in previous dialog systems, the SEMAINE framework is the first full-scale conversational agent system that takes into account the user's emotion, verbal, and non-verbal behavior via analysis of audio and visual cues and interacts while speaking and listening by means of a fully multi-modal one-to-one dialog setup.

2.1.1 Sensitive Artificial Listening

The SAL scenario was originally inspired by chat shows where hosts typically follow a simple and effective strategy: They register the emotion of the guest before uttering a phrase that has little content but makes the guest likely to adopt his or her own emotional state. Even though this conversation strategy requires only limited competence and language understanding, a computer system imitating this conversational behavior obviously needs automatic emotion recognition from voice and face. By uttering stereotyped and emotionally colored expressions and considering basic conversation strategies including turn-taking and backchanneling, a conversational agent system then has the potential to generate a feeling of 'social presence' without relying on perfect automatic speech recognition and natural language understanding [207]. Another example for the human capability to have conversations that are almost exclusively based on sensitivity to emotion are interactions taking place at parties where emotional messages can be exchanged even though noise masks large parts of the conversational partner's speech. It is these kinds of emotion-driven human conversations that are modeled in the SAL scenario. An important precondition for transferring these social competences into a human-computer interaction scenario is the definition of a representation that covers the emotional states a user is likely to show in the considered type of interaction. Here, the SEMAINE project focuses on the widely used affective dimensions *valence* (negative vs. positive emotional state) and *arousal* (active vs. passive state) [199]. The quadrants in the two-dimensional valence-arousal space (i. e., the four possible combinations of negative/positive and active/passive) are represented by four different SAL characters: 'Spike' is angry (negative-active), 'Poppy' is happy (positive-active), 'Obadiah' is sad (negative-passive), and 'Prudence' is matter-of-fact (positive-passive or neutral). Depending on the character the user currently talks to, the virtual agent system has a preferred emotional state. In case the affect recognition components of the system indicate that the user is in that preferred state, the system indicates approval. If not, the system tries to evoke its preferred state in the user. This simple conversation strategy defines the system's comments based on the current affective state of the user and was shown to enable emotionally colored human-computer interactions that are not focused on a specific topic but still can last for a few minutes and create the impression that the system is listening and commenting on what the user says [206].

2.1.2 System Architecture

Figure 2.1 shows the basic architecture of the autonomous SAL system as developed in the SEMAINE project. A microphone and a camera capture the user's speech and face. From the raw signals, features are extracted using a set of feature extraction components for audio and video. Speech feature extraction is performed

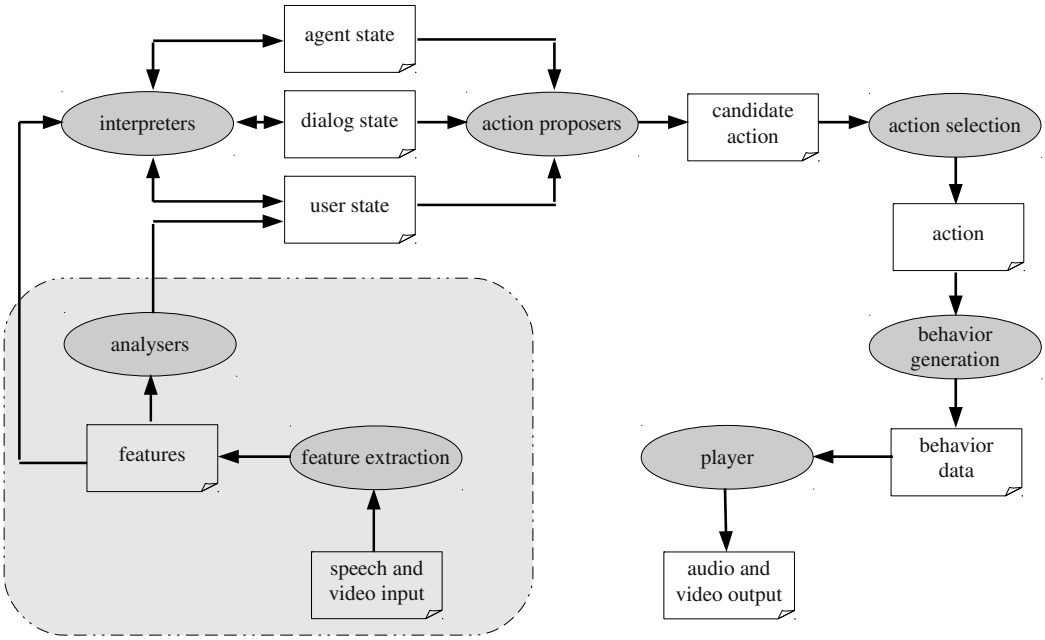


Figure 2.1: Architecture of the SEMAINE system.

using the on-line audio processing toolkit openSMILE [73] developed during the SEMAINE project at the Technische Universität München (for an overview over the most important low-level features applied for speech-based emotion recognition, see Section 2.2). Multiple analyzer components attempt to infer the user's verbal and non-verbal behavior. Most of these analyzers are classification techniques which will be dealt with in Section 2.3 and in Chapters 3 and 4. To exploit mutual information coming from different modalities, fusion components are applied. The interpreter components process low-level features as well as fused analysis results in the context of all information that is available in order to generate a current 'best guess' of the user state, dialogue state, and agent state. The task of the action proposers is to continuously decide whether the current state information should trigger the proposition of an action. As only one action can be realized at a time, an action selection module coordinates the proposed candidate actions. Finally, a behavior generation component creates the concrete vocal, facial, and gestural behavior that is rendered by a player module.

In this thesis, we mostly focus on the audio input side of the system which is indicated by a shaded box in Figure 2.1. It includes feature extraction as well as machine learning methods for classification and human behavior analysis. For details

on other components necessary to create a fully autonomous SAL system, the reader is referred to publications such as [20, 74, 119, 154, 209, 244].

2.2 Acoustic Feature Extraction

In order to obtain a compact representation of relevant information contained in the speech signal, acoustic features are periodically extracted prior to classification. Before features are extracted from the speech signal, the temporal resolution of the resulting feature vector sequence has to be defined. As we assume the signal to be quasi-stationary within the time span represented by one feature vector, the frame rate may not be chosen too low so that fast changes in the speech signal can be captured. Contrariwise, a too high frame rate degrades the accuracy of the estimated spectral features. An appropriate compromise commonly applied in speech and emotion recognition is to extract features from overlapping time windows of length 25 ms every 10 ms.

After windowing the raw time signal s^{raw}, it is common practice to pre-emphasize the signal applying the first order difference equation

$$s_n^{pre} = s_n^{raw} - k \cdot s_{n-1}^{raw} \tag{2.1}$$

$$n = 1, \ldots, N$$

to all N samples in each window. The parameter k is called the pre-emphasis coefficient and is in the range $0 \leq k < 1$. To attenuate discontinuities at the window edges, the samples in a window are usually tapered by multiplying the signal segments with a Hamming window:

$$s_n = \left\{ 0.54 - 0.46 \cos\left(\frac{2\pi(n-1)}{N-1} \right) \right\} \cdot s_n^{pre}. \tag{2.2}$$

The following sections briefly introduce a set of widely used features for speech and emotion recognition systems. Speech recognition is mostly based on spectral features such as Mel-Frequency Cepstral Coefficients (MFCC), whereas emotion recognition frameworks tend to employ a larger set of different low-level descriptors (LLD) that can be grouped into prosodic, spectral, and voice quality features (see [224], for example).

2.2.1 Prosodic Features

Energy

The short-time energy of a speech signal frame $s_{1:N}$ can be computed as follows:

$$E = \log \sum_{n=1}^{N} s_n^2. \tag{2.3}$$

Applying the logarithm accounts for the fact that the sensation of loudness increases logarithmically as the intensity of a stimulus grows. Usually, the short-time energy is normalized since parameters such as the distance to the microphone heavily influence the intensity of the recorded signal.

Pitch

The fundamental frequency F_0 (or pitch) plays an essential role in the expression of emotions via speech and thus is an important feature in the domain of speech-based affective computing. It can be estimated from voiced regions of speech applying methods based either on the time-signal or on the spectral characteristics of the signal. A popular approach focusing on the time-signal is the exploitation of the autocorrelation function (ACF) [26]. The ACF of a signal can be interpreted as a transformation that represents the similarity of the signal and a time-shifted version of the signal. Hence, the value of the ACF depends on the time-shift k:

$$ACF_k^s = \sum_{n=1}^{N-k} s_n \cdot s_{n+k}. \tag{2.4}$$

For periodic signals, a global maximum of the ACF can be found at integer multiples of the period T_0. The first maximum can be found in the origin of the ACF of a signal with its value corresponding to the signal power of s_n. If a voiced sound is detected, the fundamental frequency can be computed as the reciprocal value of the maximum T_0. To compensate distortions caused by windowing s_n (see Equation 2.2), the ACF of the speech signal is divided by the normalized ACF of the window function ACF_k^w [26]. Thus, with f_s being the sampling frequency, the short-time fundamental frequency of a voiced signal can be written as

$$F_0 = \frac{f_s}{N} \cdot \underset{k,k\neq 0}{\operatorname{argmax}} \frac{ACF_k^s}{ACF_k^w}. \tag{2.5}$$

2.2.2 Spectral Features

Formants

Formants are spectral maxima that are known to model spoken content – especially lower order formants which characterize a spoken vowel. Higher order formants also encode speaker characteristics. Normally, formant frequencies are higher than the fundamental frequency F_0 and can be represented by their center frequency, amplitude, and bandwidth. The spectral position of a formant was found to be independent of the perceived fundamental frequency. To estimate the formant frequencies and bandwidths, methods based on Linear Prediction Coding (LPC) can be applied [26].

Mel-Frequency Cepstral Coefficients

MFCCs are one of the most popular feature types used in automatic speech recognition as they efficiently encode spoken content while being relatively independent of speaker characteristics. They are based on a filterbank analysis that takes into account the non-linear frequency resolution of the human ear. Since filterbank amplitudes are highly correlated, a cepstral transformation is necessary to decorrelate the features. The filters are triangular and equally spaced on a Mel-scale:

$$MEL(f) = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right). \tag{2.6}$$

Before the filterbank can be applied, the time window of speech data is Fourier transformed and the magnitude is taken. Then, the magnitude coefficients are *binned* which means that each coefficient is multiplied by the corresponding filter gain with the results being accumulated. Consequently, a bin corresponds to a weighted sum representing the energy in the filterbank channel. Finally, Mel-frequency cepstral coefficients c_i are computed from the log filterbank amplitudes m_j by applying the Discrete Cosine Transform (DCT):

$$c_i = \sqrt{\frac{2}{N_{FB}}} \sum_{j=1}^{N_{FB}} m_j \cos\left(\frac{\pi i}{N_{FB}}(j - 0.5)\right). \tag{2.7}$$

The parameter N_{FB} denotes the number of filterbank channels.

Perceptual Linear Prediction

A well-known alternative to MFCCs are features based on Perceptual Linear Prediction (PLP) as introduced in [106]. Unlike conventional linear prediction analysis of speech which applies an all-pole model approximating the short-term power spectrum of speech equally well at all frequency regions, PLP analysis accounts for the

fact that for frequencies higher than 800 Hz the spectral resolution of human hearing decreases with frequency, whereas for amplitude levels which are typical for speech, human perception is more sensitive to frequencies in the middle of the audible spectrum. PLP analysis exploits these psychophysical facts to derive features which have been shown to be a better representation of speech than conventional linear prediction coefficients, as they consider human perception. In [106], the auditory spectrum used for all-pole modeling is obtained by convolving the power spectrum with a simulated critical-band masking pattern, resampling the critical-band spectrum under consideration of the Bark scale, pre-emphasizing by an equal-loudness curve, and compressing the spectrum by taking the cubic root which simulates the intensity-loudness power law.

2.2.3 Voice Quality Features

Harmonics-to-Noise Ratio

The Harmonics-to-Noise Ratio (HNR) is a frequently used low-level descriptor for speech-based emotion recognition. Similar to F_0, it is computed from voiced regions of the speech signal. It can be interpreted as the signal power contained in periodic parts of the signal in relation to the power of the surrounding noise. Given a periodic signal superposed by additive noise, the local maximum of the ACF, which does not correspond to the origin, is determined. The value of the ACF at the local maximum T_0 then corresponds to the signal power of the periodic part of the signal. The power of the noisy part of the signal is assumed to be the difference between the total signal power ACF_0^s and the power of the periodic part $ACF_{T_0}^s$. Consequently, the HNR in dB can be computed as

$$HNR = 10 \cdot \log \frac{ACF_{T_0}^s}{ACF_0^s - ACF_{T_0}^s}. \tag{2.8}$$

Jitter and Shimmer

Jitter and shimmer are further LLDs which were found to be useful for automatic emotion recognition [139]. They reflect voice quality properties such as breathiness and harshness and can be computed from pitch and energy contours, respectively. Jitter (J) indicates period-to-period fluctuations in the fundamental frequency and is calculated between successive voiced periods of the signal:

$$J = \frac{|T_i - T_{i+1}|}{\frac{1}{N_V}\sum_{i=1}^{N_V} T_i}. \tag{2.9}$$

T_i and T_{i+1} denote the durations of two consecutive pitch periods within an utterance consisting of N_V voiced frames. Shimmer (S) determines the period-to-period variability of the amplitude and is computed as follows:

$$S = \frac{|A_i - A_{i+1}|}{\frac{1}{N_V}\sum_{i=1}^{N_V} A_i}. \tag{2.10}$$

Here, A_i is the peak amplitude in the i^{th} time window.

2.3 Classification

Once a set of relevant speech features is extracted from the signal, some sort of classification algorithm has to be applied in order to assign a class or label to the given pattern vector or sequence of pattern vectors. Before a classifier can be applied, it has to be *trained* using appropriate machine learning techniques. Usually, a training set is available, consisting of a set of input-target pairs, i. e., data that have been manually labeled to provide a ground truth for the machine learning algorithm. These problems will be referred to as *supervised learning* tasks and will be the main focus of this thesis. Other machine learning problems not considered in the ongoing include *reinforcement learning*, where training is only based on positive or negative reward values instead of classification targets, and *unsupervised learning*, where training is not performed task-specifically, meaning that the algorithm has to reveal the structure of the data 'by inspection'.

In addition to a training set S consisting of input-target pairs (x, l) for supervised learning, a disjoint test set S' is needed to evaluate the performance of the classification algorithm on unseen data. In case the system building process includes some sort of parameter tuning, an additional validation set used for repeated validations of the system parameter settings ensures that the system is not optimized on the test data, which in turn would lead to an unrealistic final performance assessment. The overall goal is to apply the training set for minimizing a task-related error measure evaluated on data not contained in the training set, i. e., it is important that the classifier has good *generalization* properties. Generalization refers to the ability of a classifier to transfer performance from the training set to the test set and is the opposite of *over-fitting* which means that the algorithm tends to model only the training data correctly and fails to learn the general classification task. A popular approach towards error minimization for parametric algorithms is to optimize an objective function O by incrementally adjusting the classifier's parameters. This objective function is usually related to the task-specific error measure used for evaluation.

The term *pattern classification* usually denotes the static classification of non-sequential data, with standard methods comprising Support Vector Machines (SVM) [45], as outlined in Section 2.3.1, or artificial neural networks (ANN) and multilayer perceptrons (MLP) [24], as explained in Section 2.3.6. Here, an input x consists of a real-valued vector of fixed length and a target l corresponds to one single class out of a set of K possible classes. If a pattern classification algorithm is trained to directly map from inputs to classes (e. g., like SVMs), it is referred to

as a *discriminant function*. Alternatively, *probabilistic classification* determines the conditional probabilities $p(C_k|x)$ of the classes given the input and decides for the most likely class:

$$h(x) = \underset{k}{\operatorname{argmax}}\, p(C_k|x). \tag{2.11}$$

Here, $h(x)$ is the classifier output and k is the index representing the class C_k. If we design a probabilistic classifier characterized by a set of adjustable parameters w, it results in a conditional probability distribution $p(C_k|x, w)$ over the classes C_k. When taking the product over the input-target pairs contained in the training set S, we obtain

$$p(S|w) = \prod_{(x,l)\in S} p(l|x, w). \tag{2.12}$$

The application of Bayes' theorem results in

$$p(w|S) = \frac{p(S|w)p(w)}{p(S)}. \tag{2.13}$$

The posterior distribution $p(C_k|x, S)$ can be obtained by integrating over all possible values of w which, however, is infeasible in practice as w tends to be of high dimensionality and since the calculation of $p(C_k|x, S)$ is intractable. Yet, we can apply the maximum a priori (MAP) approximation by determining a single parameter set w' that maximizes Equation 2.13. We can drop $p(S)$ as it does not depend on w so that we get

$$w' = \underset{w}{\operatorname{argmax}}\, p(S|w)p(w). \tag{2.14}$$

In case we can assume a uniform prior over the set of parameters, we can also drop $p(w)$ which results in the maximum likelihood (ML) vector

$$w^\star = \underset{w}{\operatorname{argmax}}\, p(S|w) = \underset{w}{\operatorname{argmax}} \prod_{(x,l)\in S} p(l|x, w). \tag{2.15}$$

A popular approach for determining $w^\star$ is to minimize an objective function O. If we use the negative logarithm of $p(S|w)$ as objective function, we obtain

$$O = -\ln \prod_{(x,l)\in S} p(l|x, w) \tag{2.16}$$

which can also be written as

$$O = -\sum_{(x,l)\in S} \ln p(l|x, w). \tag{2.17}$$

As the logarithm is monotonically increasing, minimizing $-\ln p(S|w)$ has the same effect as maximizing $p(S|w)$.

Unlike *discriminative models* which directly compute $p(C_k|x)$, *generative models* first determine the class conditional probabilities $p(x|C_k)$ and then use the Bayes theorem as well as the class priors $p(C_k)$ to obtain $p(C_k|x)$. Generative techniques have the advantage that models for the individual classes can be trained independently from each other, while discriminative models have to be retrained as soon as a new class is added. Yet, it is well known that discriminative approaches tend to enable better classification results as they focus their modeling power on determining accurate class borders. Widely used examples for discriminative classifiers are neural networks (see Section 2.3.6), whereas Dynamic Bayesian Networks (DBN) and Hidden Markov Models are popular examples for generative models (see Sections 2.3.2 and 2.3.3).

In this thesis, we will concentrate on the analysis of *time series* such as speech signals, rather than on static classification of single input vectors x. In other words, we will focus on *sequence labeling*, meaning that a sequence of labels $l_{1:V}$ has to be assigned to a sequence of input data $x_{1:T}$, where T is the total number of time steps of a pattern vector sequence representing, e. g., the characteristics of the speech signal. We assume that the length V of the label sequence is smaller or equal to the length of the input sequence, i. e., $V \leq T$. For the case $V = T$, there exists a label for each input and the corresponding task can be called *framewise classification* (a special case of *segment classification* in which each feature frame corresponds to one segment).

The usage of *context* is essential for most segment classification algorithms. By context, we mean data on either side of the segments that are to be classified. Standard pattern classifiers which are designed to process one input at a time (e. g., SVMs) may use context by simultaneously processing data on either side of the segment, i. e., processing an extended, stacked feature vector representing data within a defined time-window. An important shortcoming of this approach is that the *range* of relevant context is generally not known and may be different for each segment. Recurrent neural networks (RNN) and Long Short-Term Memory [111] neural networks are examples for sequence labeling algorithms that model context *within* the classification framework and therefore do not need the time-window approach (see Sections 2.3.7 and 2.3.9).

The general case $V \leq T$ will be referred to as *temporal classification* in the ongoing. In contrast to framewise classification, temporal classification presumes an algorithm that is able to decide where in the input sequence the classifications should be made. This requires methods for determining the *temporal warping*, modeling the global structure of the sequence. Examples for such temporal classifiers are Dynamic Time Warping (DTW) [113], Hidden Markov Models [183], and Connectionist Temporal Classification (CTC) [90] (see Sections 2.3.3 and 2.3.10). These techniques can be extended in a way that they allow for processing multi-modal input at different

frame rates and different input sequence lengths, respectively. Asynchronous Hidden Markov Models (AHMM) [17] and Multi-Dimensional Dynamic Time Warping (MDDTW) [270] are two multi-modal classification frameworks that can cope with potentially asynchronous input modalities (see Sections 2.3.4 and 2.3.5).

In the following sections, the most important classification techniques used in this thesis will be introduced. First, in Section 2.3.1, we will briefly review the principle of Support Vector Machines, as a popular example of a static classification framework. Next, we will deal with Dynamic Bayesian Networks (Section 2.3.2) and Hidden Markov Models (Section 2.3.3), which can be used for dynamic temporal classification, being examples of generative models. Then, we switch to the multi-modal case and introduce two methods for hybrid fusion and dynamic classification of bi-modal inputs: Asynchronous Hidden Markov Models (Section 2.3.4) and Multi-Dimensional Dynamic Time Warping (Section 2.3.5). The next four sections will be devoted to neural network architectures for framewise classification: Section 2.3.6 outlines the basic principle of artificial neural networks, Section 2.3.7 introduces recurrent neural networks, Section 2.3.8 shows how *bidirectional* processing can be applied within RNNs, and Section 2.3.9 explains the concept of Long Short-Term Memory networks for enhanced RNN-based long-range temporal context modeling. Finally, Section 2.3.10 shows how neural networks can be used for temporal classification, applying the Connectionist Temporal Classification technique.

2.3.1 Support Vector Machines

Support Vector Machines [45] are one of the most frequently applied techniques for static pattern classification. They are based on the construction of a hyperplane in a potentially high dimensional feature space, which can be used for classification or regression. An SVM is built from a training set S consisting of I input-target pairs (x_i, l_i) where l_i is binary and represents one of two possible class labels: $l_i \in \{-1, +1\}$. To separate the two classes from each other, a hyperplane defined by all inputs satisfying

$$w^T x + b = 0 \tag{2.18}$$

is determined. The hyperplane is characterized by the normal vector w and the bias b and has to meet the conditions

$$\begin{aligned} l_i = +1 &\Rightarrow w^T x_i + b \geq +1, \\ l_i = -1 &\Rightarrow w^T x_i + b \leq -1. \end{aligned} \tag{2.19}$$

Provided that such a hyperplane exists, the boundary conditions can be normalized and the distance between an input and the hyperplane is

$$d(x) = \frac{w^T x + b}{||w||}. \tag{2.20}$$

Consequently, the minimum of the distances of all training inputs to the hyperplane can be interpreted as the so-called *margin of separation* and is computed as

$$\mu(w, b) = \min_{i=1,\ldots,I} |d(x_i)|. \tag{2.21}$$

Best class separation can be obtained for a hyperplane maximizing μ. Training instances having the minimum distance to the hyperplane are called *support vectors* x_i^{SV}. If the boundary conditions are normalized, their distance to the hyperplane is

$$d(x_i^{SV}) = \frac{1}{||w||}. \tag{2.22}$$

The margin of separation can be maximized using quadratic programming (also see [45]).

In most non-trivial pattern recognition problems, there exists no hyperplane that separates the classes in the training set without any errors. Thus, the equations 2.19 have to be extended by non-negative slack variables ξ_i which allow patterns to be placed on the wrong side of the hyperplane:

$$\begin{aligned} l_i = +1 &\Rightarrow w^T x_i + b \geq +1 - \xi_i, \\ l_i = -1 &\Rightarrow w^T x_i + b \leq -1 + \xi_i. \end{aligned} \tag{2.23}$$

To obtain the optimal hyperplane, the term

$$\frac{1}{2} w^T w + C \sum_{i=1}^{I} \xi_i$$

has to be minimized, where C can be freely chosen. This optimization is usually referred to as *primal problem* and is equivalent to the *dual problem*, consisting in the maximization of

$$\sum_{i=1}^{I} \alpha_i - \frac{1}{2} \sum_{i=1}^{I} \sum_{j=1}^{I} \alpha_i \alpha_j l_i l_j (x_i^T x_j)$$

with

$$0 \leq \alpha_i \leq C \tag{2.24}$$

and

$$\sum_{i=1}^{I} \alpha_i l_i = 0. \tag{2.25}$$

The normal vector of the resulting hyperplane can then be computed as a weighted sum of training samples with the coefficients α_i:

$$w = \sum_{i=1}^{I} \alpha_i l_i x_i. \tag{2.26}$$

With $i^\star$ being the index of the input vector with the largest coefficient α_i, the bias can be calculated as

$$b = l_{i^\star}(1 - \xi_{i^\star}) - x_{i^\star}^T w_{i^\star}. \tag{2.27}$$

Thus, all training samples with $\alpha_i > 0$ are support vectors. For the computation of $x_i^T x_j$ the Sequential Minimal Optimization (SMO) algorithm can be used [176]. Finally, classification is performed applying the function

$$h(x) = sgn(w^T x + b). \tag{2.28}$$

So far, we focused on linearly separable problems in which a hyperplane in the feature space can separate classes at an acceptable error. To extend the SVM principle to non-linear problems, the so-called *kernel trick* can be applied [205]. It is based on a non-linear transformation $\Phi(x_i)$ into a higher-dimensional space. This leads to a normal vector

$$w = \sum_{i:\alpha_i>0} \alpha_i l_i \Phi(x_i) \tag{2.29}$$

and a decision function

$$h(x) = sgn(w^T \Phi(x) + b). \tag{2.30}$$

For classification we need to compute

$$w^T \Phi(x) = \sum_{i:\alpha_i>0} \alpha_i l_i \Phi(x_i)^T \Phi(x) \tag{2.31}$$

which means that we do not explicitly require the transformation Φ but can apply a symmetric kernel function

$$k(x_i, x_j) = \Phi(x_i)^T \Phi(x_j). \tag{2.32}$$

Among the most frequently used kernel functions are the polynomial kernel

$$k(x_i, x_j) = (x_i^T x_j)^p \tag{2.33}$$

with polynomial order p and the Gaussian radial basis function (RBF)

$$k(x_i, x_j) = e^{\frac{||x_i - x_j||^2}{2\sigma^2}} \tag{2.34}$$

with standard deviation σ.

There exist various methods to extend the SVM principle to $K > 2$ classes, including for example the *one against all* approach with K binary decisions and the *one against one* approach with $\frac{1}{2} \cdot K \cdot (K-1)$ binary decisions.

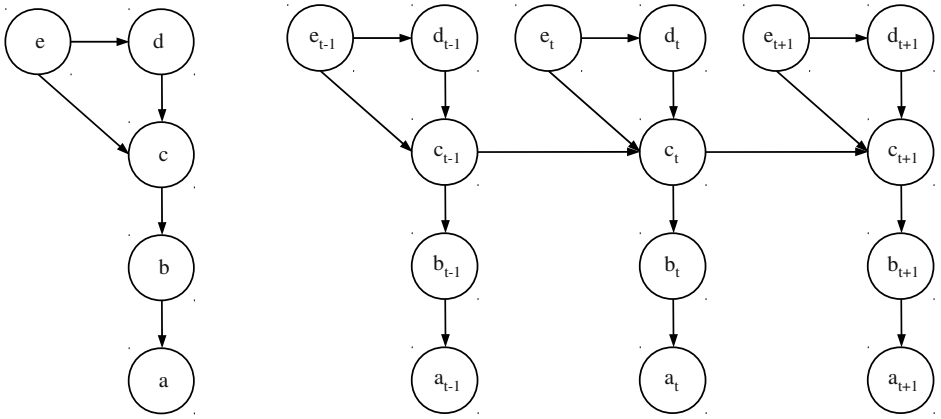
2.3.2 Dynamic Bayesian Networks

While pattern classification techniques such as SVMs estimate the class label from a given isolated pattern vector, dynamic classification approaches explicitly model the *temporal evolution* of periodically extracted feature vectors to perform sequence labeling or temporal classification (see Section 2.3). Since the speech signal is a function of time, dynamic classifiers are well-suited to model acoustic feature vector sequences, e. g., for speech recognition [278] or framewise emotion recognition [217]. Dynamic Bayesian Networks [165] offer a statistical modeling framework that is widely used in speech processing. They are part of the Graphical Model paradigm [115], which can be seen as a set of formalisms describing different types of probability distributions. GMs consist of a set of nodes and edges. Nodes represent random variables which can be either hidden or observed. If we speak of an observed variable, we mean that its value is known, i. e., there is some data or evidence available for that variable. An observed variable can for example be a feature vector that is extracted from a given signal. A hidden variable currently does not have a known value. All that is available for a hidden variable is its conditional distribution given the observed variables. Edges – or rather *missing* edges – within a Graphical Model encode conditional independence assumptions that are used to determine valid factorizations of the joint probability distribution. A Bayesian Network (BN) is a special kind of GM which has edges that are directed and acyclic. Edges point from parent nodes to child nodes. Figure 2.2(a) shows an example for a BN consisting of five nodes that represent random variables. Here, the variable a is a child of b, meaning that a is conditionally dependent on b.

BN graphs as depicted in Figure 2.2(a) implicitly reflect factorizations, being simplifications of the chain rule of probability [134]:

$$p(x_{1:N}) = \prod_i p(x_i | x_{1:i-1}) = \prod_i p(x_i | x_{\pi_i}). \tag{2.35}$$

The second equality holds for a particular BN of N random variables, where π_i denotes the set of parents of node x_i. This factorization implies that a BN can



(a) Bayesian Network (b) Dynamic Bayesian Network

Figure 2.2: Examples for a Bayesian Network and a Dynamic Bayesian Network with repeated template structure over time.

be characterized by a large number of conditional independence assumptions represented by missing edges in the graph. These assumptions can be exploited for efficient probabilistic inference. Generally, the term 'inference' refers to the computation of the probability of a subset of random variables given the values of some other subset of random variables. It can be used to make model-based predictions and to learn the model parameters, e. g., by applying the expectation maximization (EM) algorithm [57]. Exact inference tends to be computationally complex, however, the following example shows how inference can be performed more efficiently by making use of the conditional independence assumptions expressed via a BN: If we want to compute $p(a|e)$ from the joint distribution over five variables $p(a, b, c, d, e)$, we require both, $p(a, e)$ and $p(e)$. Hence, the variables b, c, and d have to be *marginalized* or integrated away in order to obtain $p(a, e)$. This can be done by the naive calculation of

$$p(a, e) = \sum_{b,c,d} p(a, b, c, d, e) \tag{2.36}$$

which, however, requires extensive computational effort. Yet, if we assume a graph as shown in Figure 2.2(a), the joint distribution can be factored as follows:

$$p(a, b, c, d, e) = p(a|b)p(b|c)p(c|d, e)p(d|e)p(e). \tag{2.37}$$

Hence, we can compute the sum as

$$p(a,e) = p(e) \sum_b p(a|b) \sum_c p(b|c) \sum_d p(c|d,e)p(d|e) \tag{2.38}$$

which is less computationally expensive since the sums are moved as far to the right as possible.

As speech is a temporal process, a Bayesian Network for speech representation must take this into account. Figure 2.2(b) shows an example for a *dynamic* Bayesian Network with a repeated 'template' structure over time [22]. Usually, a DBN is characterized by a 'rolled up' template specifying nodes and edges within one time slice as well as by the edges between successive slices. The DBN can then be 'unrolled' to any length T corresponding, e. g., to the length of the speech sequence modeled by the network.

2.3.3 Hidden Markov Models

The Hidden Markov Model is a special kind of DBN that has found many applications in temporal classification, and particularly in automatic speech recognition [183]. It is a generative model that is trained for each class and can be used to compute the conditional probability $p(x_{1:T}|C_k)$ of an observation $x_{1:T}$ given a class C_k. Usually, an observation is a sequence of feature vectors which is assumed to be generated by a Markov model. A Markov model can be regarded as a finite state machine that can change its state once every time unit. Depending on the current state s_t, a speech feature vector x_t is generated from the probability density $b_s(x_t)$ at each time step t. The probability of a transition from state i to state j is represented by the discrete probability a_{ij}. The name 'Hidden Markov Model' accounts for the fact that, unlike the observation sequence $x_{1:T}$ which is known, the underlying state sequence $s_{1:T}$ is hidden. Figure 2.3 depicts the DBN structure of an HMM with states s_t and observations x_t. Note that hidden variables are represented by circles while observed variables are denoted by squares and that straight lines refer to deterministic conditional probability functions (CPF) while zig-zagged lines represent random CPFs.

The required likelihood $p(x_{1:T}|C_k)$ can be computed by summing over all possible state sequences:

$$p(x_{1:T}|C_k) = \sum_{s_{1:T}} a_{s_0 s_1} \prod_{t=1}^{T} b_{s_t}(x_t) a_{s_t s_{t+1}} \tag{2.39}$$

For the sake of notation simplicity a non-emitting model entry state s_0 and a non-emitting model exit state s_{T+1} are introduced. As an alternative to summing over all state sequences an adequate approximation is to consider only the most likely state sequence:

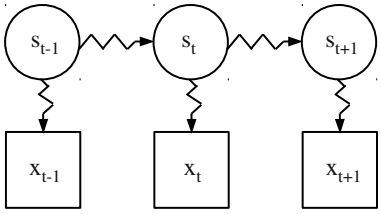


Figure 2.3: DBN structure of a Hidden Markov Model

$$\hat{p}(x_{1:T}|C_k) = \max_{s_{1:T}} \left\{ a_{s_0 s_1} \prod_{t=1}^{T} b_{s_t}(x_t) a_{s_t s_{t+1}} \right\} \tag{2.40}$$

The recognition problem is solved when the observation $x_{1:T}$ is assigned to the class C_k with the highest probability $p(x_{1:T}|C_k)$. We assume that all parameters a_{ij} and $b_s(x_t)$ are known for each model representing a class C_k. These parameters are the result of a re-estimation procedure that uses a number of training examples for each class to built the corresponding Hidden Markov Model.

In most applications the output probabilities $b_s(x_t)$ are represented by Gaussian mixture densities instead of discrete probabilities. With M being the number of mixture components and c_{sm} denoting the weight of the m^{th} component, the emission probabilities can be expressed as

$$b_s(x_t) = \sum_{m=1}^{M} c_{sm} \mathcal{N}(x_t; \mu_{sm}, \Sigma_{sm}) \tag{2.41}$$

$\mathcal{N}(\cdot; \mu, \Sigma)$ is a multivariate Gaussian with mean vector μ and covariance matrix Σ.

Baum-Welch Re-Estimation

The Baum-Welch re-estimation formula is a method to determine the parameters of an HMM [15]. If the maximum likelihood values of the means and covariances for a state s are to be calculated, it has to be considered that each observation vector x_t contributes to the parameter values for each state since the full likelihood of an observation sequence is based on the summation of all possible state sequences. Thus, the Baum-Welch re-estimation formula assigns each observation to every state in proportion to the probability of state occupancy when the vector is observed. If L_{st} is the likelihood of being in state s at time t the Baum-Welch re-estimation formula for means and covariances of a single component Gaussian distribution can be written as

$$\hat{\mu}_s = \frac{\sum_{t=1}^{T} L_{st} x_t}{\sum_{t=1}^{T} L_{st}} \tag{2.42}$$

and

$$\hat{\Sigma}_s = \frac{\sum_{t=1}^{T} L_{st}(x_t - \mu_s)(x_t - \mu_s)^T}{\sum_{t=1}^{T} L_{st}}. \tag{2.43}$$

The extension to Gaussian mixture densities is straightforward if the mixture components are considered as sub-states in which transition probabilities correspond to mixture weights. For the transition probabilities a similar formula can be derived [183].

To calculate the probabilities of state occupation L_{st}, the so-called *Forward-Backward* algorithm is used [15, 183]. The forward probability $\alpha_s(t)$ for a model representing the class C_k is defined as

$$\alpha_s(t) = p(x_{1:t}, s_t = s|C_k) \tag{2.44}$$

and can be considered as the joint probability of observing the first t feature vectors and being in state s at time t. The recursion

$$\alpha_s(t) = \left[\sum_{i=1}^{S} \alpha_i(t-1) a_{is}\right] b_s(x_t) \tag{2.45}$$

allows the efficient calculation of the forward probabilities with S denoting the number of emitting states. The backward probability $\beta_s(t)$ can be expressed as

$$\beta_s(t) = p(x_{t+1:T}|s_t = s, C_k) \tag{2.46}$$

and is calculated using the recursion

$$\beta_i(t) = \sum_{s=1}^{S} a_{is} b_s(x_{t+1}) \beta_s(t+1) \tag{2.47}$$

The probability of state occupation can be obtained by taking the product of forward and backward probability:

$$\alpha_s(t) \cdot \beta_s(t) = p(x_{1:T}, s_t = s|C_k) \tag{2.48}$$

Consequently L_{st} can be calculated as follows:

$$L_{st} = p(s_t = s|x_{1:T}, C_k) = \frac{p(x_{1:T}, s_t = s|C_k)}{p(x_{1:T}|C_k)} = \frac{1}{p(x_{1:T}|C_k)} \cdot \alpha_s(t) \cdot \beta_s(t) \tag{2.49}$$

If we assume that the last state S has to be occupied when the last observation x_T is made, the probability $p(x_{1:T}|C_k)$ is equal to $\alpha_S(T)$. Hence, the Baum-Welch re-estimation can now be performed as all information needed for the update formulas 2.42 and 2.43 is available.

Viterbi Decoding

The Viterbi algorithm [77], which is commonly used to perform recognition, is similar to the algorithm for the forward probability calculation except that the summation is replaced by a maximum operation. If for a specific model representing the class C_k the maximum likelihood of observing vectors $x_{1:t}$ while being in state s at time t is denoted by $\phi_s(t)$ the following recursion can be applied:

$$\phi_s(t) = \max_i\{\phi_i(t-1)a_{is}\}b_s(x_t) \tag{2.50}$$

Thus, the estimated maximum likelihood $\hat{p}(x_{1:T}|C_k)$ is equal to $\phi_S(T)$. The Viterbi algorithm corresponds to finding the best path through a matrix, the so-called 'trellis', in which the vertical dimension represents the states and the horizontal dimension shows the time steps. To each trellis coordinate, a probability of observing an input x_t at a time instant while being in a certain state can be assigned.

2.3.4 Asynchronous Hidden Markov Models

For some pattern recognition tasks it can be advantageous or even necessary to simultaneously model multiple input data streams coming from different modalities such as audio and video. If the data streams are perfectly synchronous and have the same frame rate (e. g., obtained via upsampling of the stream with the lower sampling rate), early fusion, i. e., feature level fusion, can be applied to model multimodal data. Compared to late fusion (decision level fusion), this has the advantage that mutual information can be used during training and decoding [270]. The concept of *hybrid fusion* unites the advantages of both, early and late fusion by exploiting mutual information and allowing the streams to be asynchronous. In [17], it has been shown that the Hidden Markov Model concept can be extended to a classification framework based on hybrid fusion by modeling the joint likelihood of two streams via so-called asynchronous Hidden Markov Models. The two streams, each coming from a different modality, do not necessarily have to be synchronous, so the AHMM can be applied to a wide range of problems like multimodal meeting analysis [307], person identification [18], audio-visual speech recognition [17], or bimodal speech and gesture interfaces [1].

An asynchronous Hidden Markov Model allows to model $p(x_{1:T}, y_{1:T'}|C_k)$ which is the joint likelihood of two observation streams $x_{1:T}$ and $y_{1:T'}$ with lengths T and T', respectively, given an AHMM representing the class C_k. Without loss of generality

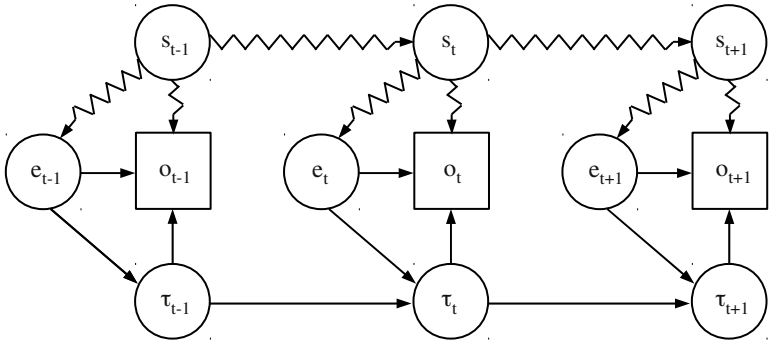


Figure 2.4: DBN structure of an asynchronous Hidden Markov Model

it is assumed that $T' \leq T$. Similar to a standard HMM, an AHMM has S different states s_t that are synchronized with stream $x_{1:T}$. At each time step t a state emits an observation from stream $x_{1:T}$. At the same time a state can (with the probability ϵ_s) also emit an observation from stream $y_{1:T'}$. Every time a y observation is emitted, the variable $\tau_t = 0...T'$ is incremented until the last y observation has been emitted. Therefore τ_t can be seen as a second hidden variable which models the alignment between $x_{1:T}$ and $y_{1:T'}$. The additional variable τ_t is included by adding a third dimension τ to the trellis. Figure 2.4 shows the DBN structure of an AHMM. Here, the multi-modal observation variable o_t subsumes both modalities, meaning that it can consist of either only x_t or of both, x_t and y_t. The variable e_t is binary and indicates whether an observation of stream $y_{1:T'}$ is emitted or not.

To calculate the likelihood $p(x_{1:T}, y_{1:T'}|C_k)$ of a bimodal observation given a certain AHMM representing C_k, we need a forward path variable $\alpha_{s,\tau}(t)$ [17] that, unlike the corresponding forward path variable for standard HMMs, depends on three indices which are state, alignment, and time:

$$\alpha_{s,\tau}(t) = p(s_t = s, \tau_t = \tau, x_t, y_\tau). \tag{2.51}$$

Provided that $\tau > 0$ (meaning that the model has already emitted a y observation), the induction step is

$$\begin{aligned}\alpha_{s,\tau+1}(t+1) = [1-\epsilon_s] \cdot p(x_{t+1}|s_{t+1} = s) \sum_{j=1}^{S} p(s_{t+1} = s|s_t = j) \cdot \alpha_{j,\tau+1}(t) \\ +\epsilon_s \cdot p(x_{t+1}, y_{\tau+1}|s_{t+1} = s) \sum_{j=1}^{S} p(s_{t+1} = s|s_t = j) \cdot \alpha_{j,\tau}(t).\end{aligned} \tag{2.52}$$

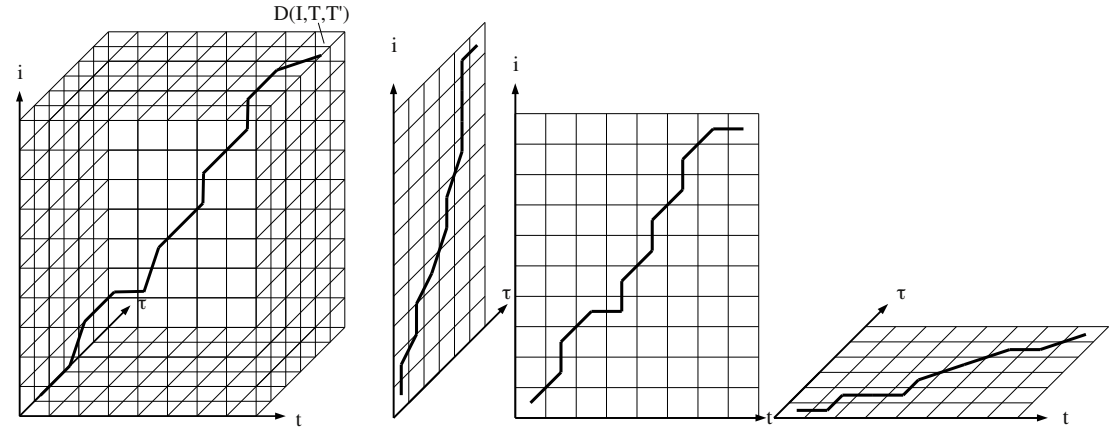
For the joint likelihood of the two observations the following termination equation holds:

$$p(x_{1:T}, y_{1:T'}|C_k) = \sum_{j=1}^{S} \alpha_{j,T'}(T). \tag{2.53}$$

The Viterbi decoding algorithm is similar to the forward path calculation. However, the sums have to be replaced by max operators. Via backtracking the best state-sequence and the most probable alignment of the two streams can be obtained. When calculating the forward path variable for all possible combinations of s, τ, and t, the complexity of the AHMM algorithm is $\mathcal{O}(S^2T'T)$ as each induction step approximately requires S summations. If the alignment between x and y is forced in a way that $|t - T/T'| < k$, with k being a constant indicating the maximum stretching between the streams, the complexity is reduced to $\mathcal{O}(S^2Tk)$ [17]. In [1], it was shown that the complexity is reduced to $\mathcal{O}(S^2[TT' - T'^2 + T])$ if α values that cannot be part of a valid path through the three-dimensional trellis are ignored. The path restriction is implied by the fact that all y observations have to be emitted until the last time step and the assumption that at every time step the number of emitted y observations cannot be larger than the number of emitted x observations and therefore $\tau \leq t$.

2.3.5 Multi-Dimensional Dynamic Time Warping

A major drawback of the AHMM is its comparably high computational complexity. Thus, in [270], a less complex hybrid fusion approach based on Dynamic Time Warping has been proposed. Generally, the DTW algorithm calculates the distance between an input sequence x_t and a reference sequence r_i which can be seen as the prototype of a certain class. As these two sequences may have different lengths or may differ in their temporal characteristics, the DTW algorithm performs a nonlinear distortion of the time axis so that the maximum correlation can be determined. Besides the distance, which can be seen as a similarity measure between an input pattern and a stored reference pattern, the DTW also delivers a warping function that maps each sample of the input to the corresponding sample of the reference sequence. In [270], it was shown how a three-dimensional DTW (3D-DTW) algorithm can model potentially asynchronous bimodal data, similar to the AHMM concept. The 3D-DTW algorithm searches for the best alignment between a synchronized reference sequence $r_{1:I}$, containing features of both modalities, an input sequence $x_{1:T}$, and a secondary input sequence $y_{1:T'}$. Their alignment can be visualized by a path through a three-dimensional distance matrix (see Figure 2.5(a)). The projection of the path to the i-t plane corresponds to the DTW-path that maps input stream $x_{1:T}$ to the features of the first modality of reference sequence $r_{1:I}$ (Figure 2.5(b), middle). Consequently, the nonlinear distortion of input stream $y_{1:T'}$, which is compared to



(a) Three-dimensional distance matrix.

(b) Projections of the path: i-τ, i-t, and t-τ plane

Figure 2.5: Warping function of the 3D-DTW.

the features of the second modality of $r_{1:I}$ can be seen in the path projection to the i-τ plane (Figure 2.5(b), left), whereas the path in the t-τ plane represents the best alignment between the two potentially asynchronous input streams $x_{1:T}$ and $y_{1:T'}$ (Figure 2.5(b), right).

For the three-dimensional DTW approach, a synchronized reference stream $r_{1:I}$, consisting of the reference features of both modalities $r^A_{1:I}$ and $r^B_{1:I}$, is used (see [270] for details on the corresponding synchronization algorithm). The elements of the distance matrix can be calculated as follows:

$$d(i,t,\tau) = \sum_{n=1}^{N} [r^A_{i,n} - x_{t,n}]^2 + g \cdot \sum_{m=1}^{M} [r^B_{i,m} - y_{\tau,m}]^2. \tag{2.54}$$

The variable $n = 1...N$ counts the features of the first input sequence $x_{1:T}$, while $m = 1...M$ counts the features of $y_{1:T'}$. With g, a factor to weight the distance coming from the individual modalities is introduced. Similar to the unimodal DTW, the best alignment can be visualized by a warping function that determines the path through the distance matrix (Figure 2.5(a)), going from cell $d(1,1,1)$ to cell $d(I,T,T')$. For the calculation of the best path, a three-dimensional accumulated distance matrix D is needed. Its endpoint $D(I,T,T')$ is equivalent to the total accumulated distance between the reference sequence and the two input streams. Considering a cell $D(i,t,\tau)$ with $i \geq 2$, $t \geq 2$, and $\tau \geq 2$, the accumulated distance can be determined by choosing the best of seven possible preceding cells [270]. If cell $D(i,t,\tau)$ is reached by a movement parallel to one of the axis, the distance $d(i,t,\tau)$ is added to the accumulated distance of the preceding cell. In case $D(i,t,\tau)$ is reached

by a movement parallel to one of the planes $i-\tau$, $i-t$, or $t-\tau$, the distance $d(i,t,\tau)$ is weighted by factor two because otherwise diagonal movements would be preferred. Consequently, $d(i,t,\tau)$ has to be weighted by factor three if cell $D(i-1,t-1,\tau-1)$ is considered as preceding cell as this movement could also be reached by three successive movements parallel to the three axes i, t, and τ. These considerations result in the equation

$$D(i,t,\tau) = \min \begin{cases} D(i-1,t,\tau) & + & d(i,t,\tau) \\ D(i,t-1,\tau) & + & d(i,t,\tau) \\ D(i,t,\tau-1) & + & d(i,t,\tau) \\ D(i-1,t-1,\tau) & + & 2 \cdot d(i,t,\tau) \\ D(i-1,t,\tau-1) & + & 2 \cdot d(i,t,\tau) \\ D(i,t-1,\tau-1) & + & 2 \cdot d(i,t,\tau) \\ D(i-1,t-1,\tau-1) & + & 3 \cdot d(i,t,\tau) \end{cases} \tag{2.55}$$

$$(i \geq 2, t \geq 2, \tau \geq 2).$$

For further equations, detailed computational complexity calculations, an extension to four-dimensional DTW for unsynchronized reference sequences, as well as for the derivation of a probability-based version of the 3D-DTW and experimental results, the reader is referred to [270].

2.3.6 Artificial Neural Networks

Artificial neural networks are widely used pattern classifiers and were originally built as computational models of the information processing paradigm of the human brain [197, 198]. An ANN can be interpreted as a network of nodes which are joined to each other by weighted connections. The nodes represent neurons while the weights of the connections correspond to the strength of the synapses between the neurons of the biological model. A frequently applied form of neural network is the multilayer perceptron [198] whose nodes are arranged in multiple layers. Connections in an MLP are 'feeding forward' from one layer to the next. An MLP for pattern recognition consists of an input layer whose activations correspond to the components of the feature vector, multiple hidden layers, and an output layer indicating the classification result (see Figure 2.6). The hidden layers usually have neurons with non-linear activation functions transforming the weighted sum of activations at the input of the node. The propagation of input activations through the hidden layers to the output is referred to as the *forward pass*.

For an MLP with I input nodes activated by the feature vector x, the activation α^h of a hidden unit h in the first hidden layer of the network can be computed as a weighted sum of the inputs

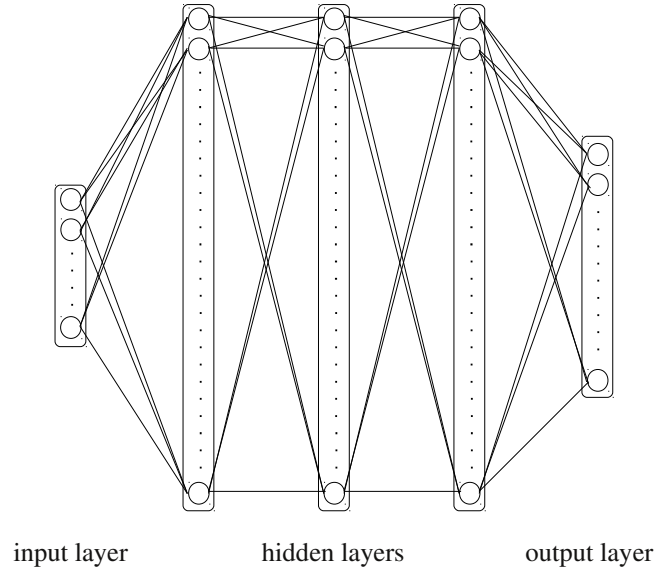


Figure 2.6: Architecture of a multilayer perceptron

$$\alpha^h = \sum_{i=1}^{I} \eta^{ih} x_i \tag{2.56}$$

with η^{ij} denoting the weight from unit i to unit j. The final activation β^h after applying the activation function f_h can be written as

$$\beta^h = f_h(\alpha^h). \tag{2.57}$$

Two frequently used activation functions are the hyperbolic tangent and the logistic sigmoid:

$$\tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{2.58}$$

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{2.59}$$

Note that both of these activation functions are non-linear which implies that the corresponding MLP is able to model non-linear classification boundaries, for example.

Since both functions are differentiable, the network can be trained applying gradient decent.

Once the activations of the first hidden layer are determined, the activation of a hidden unit h in a successive hidden layer q can be calculated as

$$\alpha^h = \sum_{h'=1}^{H^{q-1}} \eta^{h'h} \beta^{h'} \tag{2.60}$$

$$\beta^h = f_h(\alpha^h) \tag{2.61}$$

if H^q denotes the number of neurons in layer q. Similarly, the activation of an output unit k corresponds to a weighted sum of activations in the last of the L hidden layers, so that

$$\alpha^k = \sum_{h=1}^{H^L} \eta^{hk} \beta^h. \tag{2.62}$$

In order to use the output vector o for a classification task involving K possible classes, the common strategy is to build a network with K output units and to normalize the output activations α^k with the *softmax* function which results in estimates of the class probabilities

$$p(C_k|x) = o^k = \frac{e^{\alpha^k}}{\sum_{k'=1}^{K} e^{\alpha^{k'}}}. \tag{2.63}$$

The target class label l can be represented as a binary vector which consists of only zeros except for the entry l^k which is one and thus indicates that the correct class is C_k. Hence, the target probabilities can be expressed as

$$p(l|x) = \prod_{k=1}^{K} (o^k)^{l^k}. \tag{2.64}$$

In other words, the most active output units encodes the estimated class label and can thus be used as the pattern classification result.

If we substitute Equation 2.64 into 2.17, we obtain the maximum likelihood objective functions

$$O = -\sum_{(x,l)\in S} \sum_{k=1}^{K} l^k \ln o^k \tag{2.65}$$

(for more details, see [24]). Applying gradient decent, MLPs can be trained to minimize any differentiable objective function as MLPs themselves are also differentiable operators. To this aim, the derivative of the objective function with respect to the network parameters (i. e., with respect to all the network weights) has to be found,

so that the parameters can be adjusted in the direction of the negative slope. In Equation 2.65, the objective function is defined as a sum over the complete training set. However, to simplify the following equations, we will focus on the derivatives of an objective function for one particular labeled training example. The computation of the derivatives for the whole training set is straightforward, as we simply have to sum over all training instances.

The gradient can be determined via a method called *backpropagation* or the *backward pass* [198], which repeatedly applies the chain rule for partial derivatives. Unlike the forward pass, which refers to the propagation of the input activations to the network output, we now start with the output layer by calculating the derivatives of the objective function with respect to the output nodes. If we differentiate 2.65, we get

$$\frac{\partial O}{\partial o^k} = -\frac{l^k}{o^k}. \tag{2.66}$$

Note that, according to Equation 2.63, the activation of each node in the (softmax) output layer depends on the input to *each* node in the output layer. Thus, when applying the chain rule, we obtain

$$\frac{\partial O}{\partial \alpha^k} = \sum_{k'=1}^{K} \frac{\partial O}{\partial o^{k'}} \frac{\partial o^{k'}}{\partial \alpha^k} \tag{2.67}$$

as the derivative of the objective function with respect to the output activations *before* application of the softmax normalization. The differentiation of Equation 2.63 with respect to α^k gives

$$\frac{\partial o^{k'}}{\partial \alpha^k} = o^k \delta_{kk'} - o^k o^{k'} \tag{2.68}$$

where δ_{ij} denotes the Kronecker delta, i. e., $\delta_{ij} = 1$ if $i = j$ and zero otherwise. Finally, we substitute both, Equation 2.66 and 2.68 into Equation 2.67 and get

$$\frac{\partial O}{\partial \alpha^k} = o^k - l^k \tag{2.69}$$

since $\sum_{k=1}^{K} l^k = 1$.

The next step is to go backwards through the hidden layers of the network, continuing to apply the chain rule. The derivative with respect to units in the last hidden layer is

$$\frac{\partial O}{\partial \beta^h} \frac{\partial \beta^h}{\partial \alpha^h} = \frac{\partial \beta^h}{\partial \alpha^h} \sum_{k=1}^{K} \frac{\partial O}{\partial \alpha^k} \frac{\partial \alpha^k}{\partial \beta^h}. \tag{2.70}$$

Thus, we have to differentiate Equations 2.57 and 2.62 and obtain

$$\frac{\partial O}{\partial \alpha^h} = f'_h(\alpha^h) \sum_{k=1}^{K} \frac{\partial O}{\partial \alpha^k} \eta^{hk}. \tag{2.71}$$

For the remaining hidden layers, we can use the recursive equation

$$\frac{\partial O}{\partial \alpha^h} = f'_h(\alpha^h) \sum_{h'=1}^{H^{q+1}} \frac{\partial O}{\partial \alpha^{h'}} \eta^{hh'}. \tag{2.72}$$

Provided that we have determined the derivatives of the objective function with respect to the activations of all hidden cells we finally are able to calculate the derivatives with respect to all the network weights by using Equation 2.56:

$$\frac{\partial O}{\partial \eta^{ij}} = \frac{\partial O}{\partial \alpha^j} \frac{\partial \alpha^j}{\partial \eta^{ij}} = \frac{\partial O}{\partial \alpha^j} \beta^i. \tag{2.73}$$

Now, we can update the network weights by applying the gradient decent algorithm, i. e., by repeatedly taking fixed-size steps in the direction of the negative error gradient. If $w(n)$ corresponds to a vector of weights after the nth update, we calculate

$$\Delta w(n) = -r \frac{\partial O}{\partial w(n)} \tag{2.74}$$

with r being the so-called *learning rate* that takes values between 0 and 1. To prevent the algorithm from converging towards local minima, we add a *momentum* term

$$\Delta w(n) = m \Delta w(n-1) - r \frac{\partial O}{\partial w(n)} \tag{2.75}$$

where the momentum parameter m is in the range from 0 to 1 (also see [24]).

2.3.7 Recurrent Neural Networks

When applying artificial neural networks for sequence labeling tasks in which context in the form of inputs from past time steps can be exploited for enhancing the estimation of the target label in the current time step, it is beneficial to employ *recurrent* neural networks, i. e., ANNs that have cyclic connections in the hidden layer. Self connected hidden cells used within RNNs collect (weighted) activations not only from the input nodes but also from hidden nodes in the previous time step. This implicitly allows a 'memory' of previous inputs that can be modeled in the internal state of the network. Unlike MLPs, which can only map from input vectors to output vectors, an RNN can theoretically map from the 'history' of previous input vectors to an output vector. Figure 2.7 shows the structure of an RNN with one

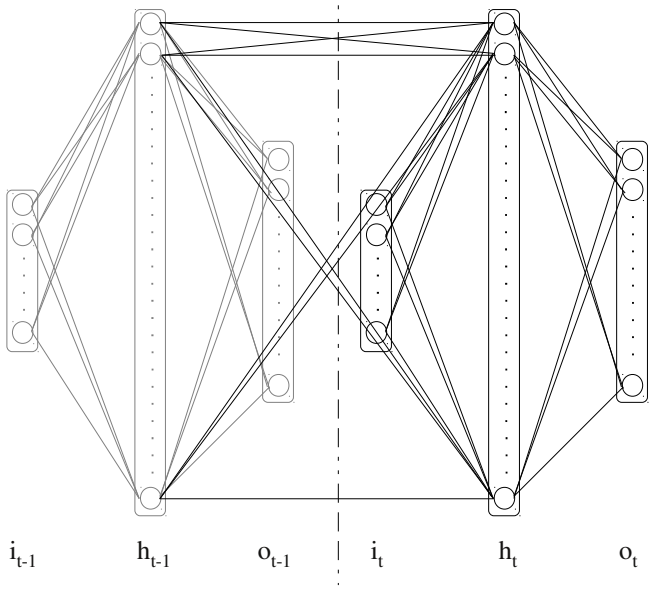


Figure 2.7: Architecture of a recurrent neural network with one hidden layer. i_t refers to the vector of input activations at time t, h_t denotes the vector of activations in the hidden layer, and o_t is the vector of output activations.

hidden layer where the hidden nodes are connected to themselves or, in other words, to the hidden nodes of the previous time step.

For the sake of simplicity, we will focus on RNNs consisting of only one hidden layer when deriving the forward and backward pass of the network. Instead of static, isolated pattern vectors x representing the input of an MLP, we now consider a *sequence* of input vectors $x_{1:T}$ with length T. Again, we denote the number of input, hidden, and output nodes by I, H, and K. For the calculation of the activations in the hidden layer we now have to consider both, the inputs in the current time step and the hidden units in the previous time step. This results in

$$\alpha_t^h = \sum_{i=1}^{I} \eta^{ih} x_t^i + \sum_{h'=1}^{H} \eta^{hh'} \beta_{t-1}^{h'}. \tag{2.76}$$

Again, an activation function f_h is applied to obtain the final activation of hidden unit h at time t:

$$\beta_t^h = f_h(\alpha_t^h). \tag{2.77}$$

To calculate the whole sequence of activations, we have to start at $t = 1$ and recursively apply the above equations while incrementing t at every time step. The activations β_0^h are defined to be equal to zero. Similar to Equation 2.62 for MLPs, the inputs to the output units can be calculated by summing over the weighted hidden activations:

$$\alpha_t^k = \sum_{h=1}^{H} \eta^{hk} \beta_t^h. \tag{2.78}$$

For the backward pass we require the partial derivatives of the objective function with respect to the weights. These derivatives can be determined via backpropagation through time [266] which, similar to the backward pass for MLPs, is a repeated application of the chain rule. Now we have to consider that the objective function depends on the activations in the hidden layer not only via its influence on the output layer, but additionally via its influence on the hidden layer in the next time step. Thus, we obtain

$$\frac{\partial O}{\partial \alpha_t^h} = f_h'(\alpha_t^h) \left(\sum_{k=1}^{K} \frac{\partial O}{\partial \alpha_t^k} \eta^{hk} + \sum_{h'=1}^{H} \frac{\partial O}{\partial \alpha_{t+1}^{h'}} \eta^{hh'} \right). \tag{2.79}$$

As we have to calculate the complete *sequence* of partial derivatives, we have to start at $t = T$ and recursively apply the above equation while decrementing t in each round. Similar to the forward step we assume

$$\frac{\partial O}{\partial \alpha_{T+1}^h} = 0. \tag{2.80}$$

Since the weights do not change for different time steps, we have to sum over the complete sequence to obtain the partial derivatives of the objective function with respect to the network weights:

$$\frac{\partial O}{\partial \eta^{ij}} = \sum_{t=1}^{T} \frac{\partial O}{\partial \alpha_t^j} \frac{\partial \alpha_t^j}{\partial \eta^{ij}} = \sum_{t=1}^{T} \frac{\partial O}{\partial \alpha_t^j} \beta_t^i. \tag{2.81}$$

2.3.8 Bidirectional Recurrent Neural Networks

Conventional RNNs are restricted in a way that they exclusively model past, but not future context. However, for many tasks such as framewise labeling of phonemes in speech signals, it is beneficial to have access to both, past and future context information. There exists a variety of straightforward approaches to incorporate future context into RNN-based classification. One possibility is to define a time window of future context and add the corresponding number of future frames to the network input. Yet, an important drawback of this method is that the range

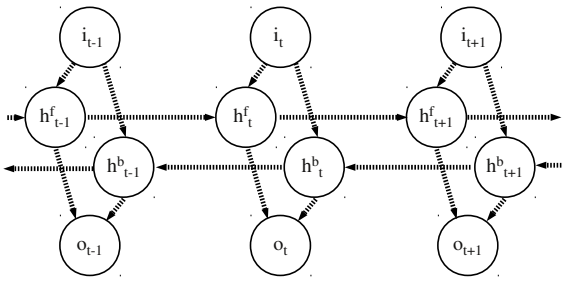


Figure 2.8: Structure of a bidirectional network with input i, output o, and two hidden layers (h^f and h^b) for forward and backward processing. All input, hidden, and output cells in one time step are summarized by one node denoted as i_t, h_t^f / h_t^b, and o_t, respectively.

of context is fixed in that case, i. e., the range of future context has to be specified by hand. The same holds for approaches that introduce a defined delay between inputs and classification targets. An additional disadvantage of networks with time delay is the asymmetry between past and future context modeling as past context is modeled via cyclic connections in the RNN while future context is modeled through the time delay. Further, the network might have difficulties in 'remembering' the original input and the past observations throughout the delay.

Attempting to overcome these problems, a solution based on *bidirectional* modeling was introduced in [229]. Bidirectional recurrent neural networks (BRNN) consist of two separate recurrent hidden layers, one that processes the input sequence forwards (from $t = 1$ to $t = T$) and one that processes the input in backward direction (from $t = T$ to $t = 1$). Both of these hidden layers are connected to the same output layer (see Figure 2.8). The effect is that the network has access to the entire past and future context while preserving the temporal synchrony between inputs and targets.

During the BRNN forward pass the input sequence is processed in opposite directions by the two hidden layers. Only after both hidden layers have processed the whole sequence of inputs, the output layer is updated. Hence, for each time step from $t = 1$ to $t = T$, the activations have to be stored while the forward pass for the forward hidden layer is performed. Then, starting at $t = T$ until $t = 1$, the forward pass of the backward hidden layer has to be carried out, again storing the activations for each t. Finally, using the stored activations from both hidden layers, the forward pass for the output layer can be performed.

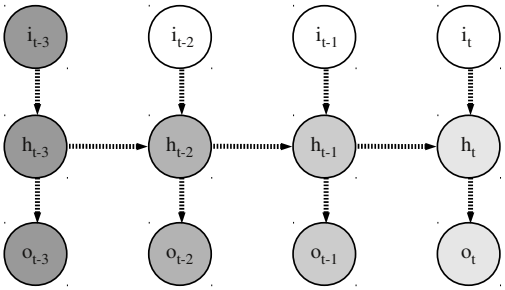
For the backward pass, the partial derivatives of the objective function with respect to the output layer activations have to be computed before they are used in opposite directions within the two hidden layers (i. e., this time we start at $t = T$ in the forward hidden layer and at $t = 1$ in the backward hidden layer).

Note that BRNNs cannot be used in every causal on-line recognition task as future context might not be available if fully incremental processing is required. However, bidirectional networks can nevertheless be applied for a variety of causal and temporal tasks such as speech recognition as soon as we drop the requirement of fully incremental classification. For example, in speech recognition it might be acceptable to wait until the end of a sentence, or until a pause in speech, before the speech sequence is processed.

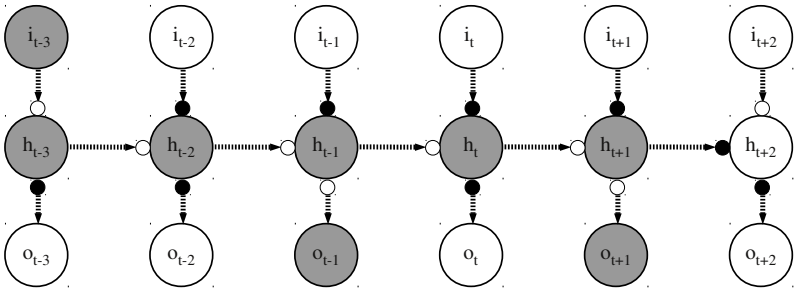
2.3.9 Long Short-Term Memory Networks

Even though the recurrent connections in RNNs allow to model contextual information, which makes them a more effective sequence labeling framework than, for example MLPs or SVMs, it is known that the range of context that standard RNNs can access is limited [110]. The reason for this is that the influence of a certain input on the hidden and output layer of the network either blows up or decays exponentially over time while cycling around the recurrent connections of the network. In literature, this problem is referred to as the so-called *vanishing gradient problem*. The effect of this decaying sensitivity is that RNNs have difficulties in learning temporal dependencies for which relevant inputs and targets are separated by more than ten time steps [110], i. e., the network cannot *remember* previous inputs over longer time spans so that it is hardly possible to model input-target dependencies that are not synchronous. This led to various attempts to address the problem of vanishing gradients for RNN, including non-gradient-based training [19], time-delay networks [132, 145, 202], hierarchical sequence compression [203], and echo state networks [114]. One of the most effective techniques is the Long Short-Term Memory architecture [111], which is able to store information in linear memory cells over a longer period of time. LSTM is able to overcome the vanishing gradient problem and can learn the optimal amount of contextual information relevant for the classification task.

An LSTM hidden layer is composed of multiple recurrently connected subnets which will be referred to as *memory blocks* in the following. Every memory block consists of self-connected *memory cells* and three multiplicative *gate* units (input, output, and forget gates). Since these gates allow for write, read, and reset operations within a memory block, an LSTM block can be interpreted as (differentiable) memory chip in a digital computer. An illustration of the vanishing gradient problem and its solution via LSTM can be seen in Figures 2.9(a) and 2.9(b), respectively. In this example, the shading of the nodes indicates the sensitivity to the input at time $t - 3$ (the darker the shading, the greater the sensitivity). In conventional RNNs (Figure 2.9(a)) the sensitivity decays over time since new inputs overwrite the activation of the hidden cells. Note that, for the sake of simplicity, all input, hidden, and output cells in one time step are summarized by one node denoted as i_t, h_t, and o_t, respectively. Figure 2.9(b) shows a simplified architecture of an LSTM network,



(a) Recurrent neural network



(b) Long Short-Term Memory network

Figure 2.9: The vanishing gradient problem in RNNs: The shading of the nodes corresponds to the hidden cell's sensitivity to the input at $t-3$. LSTM networks are able to 'remember' the input at $t-3$ by additional gate units controlling the hidden cells.

where each hidden node is equipped with three different gate units, indicated by small circles. Here, we assume that gates are either entirely open or entirely closed. A white circle corresponds to an open gate while a black circle indicates a closed gate. As long as the forget gate is open and the input gate is closed, the hidden cell activation cannot be overwritten by new inputs and the input information from time $t-3$ can be accessed at arbitrary time steps by opening the output gate. Figure 2.10 contains a more detailed illustration of the architecture of a memory block comprising one memory cell.

If α_t^{in} denotes the activation of the input gate at time t *before* the activation function f_g has been applied and β_t^{in} represents the activation *after* application of the activation function, the input gate activations (forward pass) can be written as

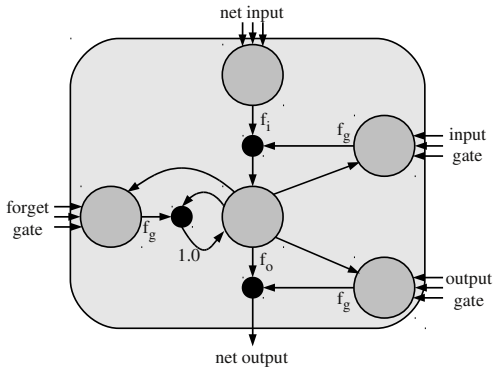


Figure 2.10: LSTM memory block consisting of one memory cell: the input, output, and forget gates collect activations from inside and outside the block which control the cell through multiplicative units (depicted as small circles); input, output, and forget gate scale input, output, and internal state respectively; f_i, f_g, and f_o denote activation functions; the recurrent connection of fixed weight 1.0 maintains the internal state.

$$\alpha_t^{\text{in}} = \sum_{i=1}^{I} \eta^{i,\text{in}} x_t^i + \sum_{h=1}^{H} \eta^{h,\text{in}} \beta_{t-1}^h + \sum_{c=1}^{C} \eta^{c,\text{in}} s_{t-1}^c \tag{2.82}$$

and

$$\beta_t^{\text{in}} = f_g(\alpha_t^{\text{in}}), \tag{2.83}$$

respectively. Again, the variable η^{ij} corresponds to the weight of the connection from unit i to unit j while 'in', 'for', and 'out' refer to input gate, forget gate, and output gate, respectively. Indices i, h, and c count the inputs x_t^i, the cell outputs from other blocks in the hidden layer, and the memory cells, while I, H, and C are the number of inputs, the number of cells in the hidden layer, and the number of memory cells. Finally, s_t^c corresponds to the *state* of a cell c at time t, meaning the activation of the linear cell unit.

Similarly, the activation of the forget gates before and after applying f_g can be calculated as follows:

$$\alpha_t^{\text{for}} = \sum_{i=1}^{I} \eta^{i,\text{for}} x_t^i + \sum_{h=1}^{H} \eta^{h,\text{for}} \beta_{t-1}^h + \sum_{c=1}^{C} \eta^{c,\text{for}} s_{t-1}^c \tag{2.84}$$

$$\beta_t^{\text{for}} = f_g(\alpha_t^{\text{for}}). \tag{2.85}$$

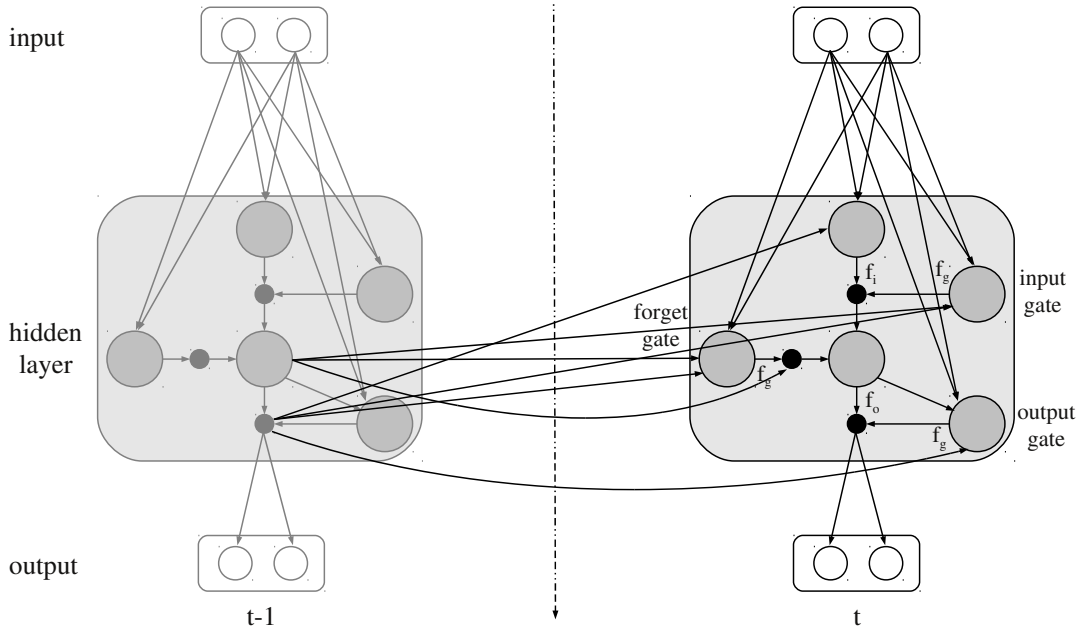


Figure 2.11: Connections in an LSTM network consisting of two input nodes, one memory cell with one memory block, and two output nodes.

The memory cell value α_t^{c} is a weighted sum of inputs at time t and hidden unit activations at time $t-1$:

$$\alpha_t^{\mathrm{c}} = \sum_{i=1}^{I} \eta^{i,\mathrm{c}} x_t^i + \sum_{h=1}^{H} \eta^{h,\mathrm{c}} \beta_{t-1}^h. \tag{2.86}$$

To determine the current state of a cell c, we scale the previous state by the activation of the forget gate and the input $f_i(\alpha_t^{\mathrm{c}})$ by the activation of the input gate:

$$s_t^{\mathrm{c}} = \beta_t^{\mathrm{for}} s_{t-1}^{\mathrm{c}} + \beta_t^{\mathrm{in}} f_i(\alpha_t^{\mathrm{c}}). \tag{2.87}$$

The computation of the output gate activations follows the same principle as the calculation of the input and forget gate activations, however, this time we consider the *current* state s_t^c, rather than the state from the previous time step:

$$\alpha_t^{\mathrm{out}} = \sum_{i=1}^{I} \eta^{i,\mathrm{out}} x_t^i + \sum_{h=1}^{H} \eta^{h,\mathrm{out}} \beta_{t-1}^h + \sum_{c=1}^{C} \eta^{c,\mathrm{out}} s_t^c \tag{2.88}$$

$$\beta_t^{\mathrm{out}} = f_g(\alpha_t^{\mathrm{out}}). \tag{2.89}$$

Finally, the memory cell output is determined as

$$\beta_t^{\mathrm{c}} = \beta_t^{\mathrm{out}} f_o(s_t^{\mathrm{c}}). \tag{2.90}$$

Figure 2.11 provides an overview over the connections in an 'unrolled' LSTM network for time steps $t-1$ and t. For the sake of simplicity, this network only contains small input and output layers (two nodes each) and just one memory block with one cell. Note that the initial version of the LSTM architecture contained only input and output gates. Forget gates were added later [84] in order to allow the memory cells to reset themselves whenever the network needs to *forget* past inputs. In this thesis, we exclusively consider the enhanced LSTM version including forget gates.

Similar to standard ANNs, LSTM networks can be interpreted as differentiable 'function approximators' and can be trained using backpropagation through time in combination with gradient descent [93]. For equations defining the LSTM backward pass, the reader is referred to [89]. As shown in [93], it is possible to combine the principles of bidirectional networks (see Section 2.3.8) and Long Short-Term Memory, which results in bidirectional Long Short-Term Memory (BLSTM) and allows to model long-range context in both input directions.

2.3.10 Connectionist Temporal Classification

A major limitation of the standard objective functions for RNNs is that they require individual targets for each point in the data sequence, which in turn requires the boundaries between segments with different labels (e. g., the phoneme boundaries in speech) to be pre-determined. The Connectionist Temporal Classification output layer [90] solves this problem by allowing the network to choose the location as well as the class of each label. By summing up over all sets of label locations that yield the same label sequence, CTC determines a probability distribution over possible labelings, conditioned on the input sequence.

A CTC layer has as many output units as there are distinct labels for a task, plus an extra *blank* unit for no label. The activations of the outputs at each time step are normalized and interpreted as the probability of observing the corresponding label (or no label) at that point in the sequence. Because these probabilities are conditionally independent given the input sequence $x_{1:T}$, the total probability of a given (framewise) sequence $z_{1:T}$ of blanks and labels is

$$p(z_{1:T}|x_{1:T}) = \prod_{t=1}^{T} o_t^{z_t}, \tag{2.91}$$

where o_t^k is the activation of output unit k at time t. In order to sum over all the output sequences corresponding to a particular labeling (regardless of the *location* of the labels) we define an operator $\mathcal{B}(\cdot)$ that removes first the repeated labels and then

the blanks from the output sequence, so that e. g., $\mathcal{B}(AA--BBB-B) = ABB$. The total probability of the length V labeling $l_{1:V}$, where $V \leq T$, is then

$$p(l_{1:V}|x_{1:T}) = \sum_{z_{1:T}:\mathcal{B}(z_{1:T})=l_{1:V}} p(z_{1:T}|x_{1:T}). \tag{2.92}$$

A naive calculation of Equation 2.92 is unfeasible, because the number of $z_{1:T}$ terms corresponding to each labeling increases exponentially with the sequence length. However, $p(l_{1:V}|x_{1:T})$ can be efficiently calculated with a dynamic programming algorithm similar to the forward-backward algorithm for HMMs. If we consider a modified label sequence $l'_{1:V'}$ with the blank label added to the beginning and end, and between each pair of labels (giving $l'_{1:V'}$ a total length of $V' = 2V + 1$), then for segment v of $l'_{1:V'}$ and time t we define a forward variable $\alpha_t(v)$ and a backward variable $\beta_t(v)$:

$$\alpha_t(v) = \sum_{z_{1:t}:\mathcal{B}(z_{1:t})=l_{1:v/2}} \prod_{t'=1}^{t} o_{t'}^{z_{t'}} \tag{2.93}$$

$$\beta_t(v) = \sum_{z_{t:T}:\mathcal{B}(z_{t:T})=l_{v/2:V}} \prod_{t'=t+1}^{T} o_{t'}^{z_{t'}}. \tag{2.94}$$

With these definitions it can be shown that

$$p(l_{1:V}|x_{1:T}) = \sum_{v=1}^{V'} \alpha_t(v)\beta_t(v). \tag{2.95}$$

The CTC objective function O^{CTC} is defined as the negative log likelihood of the training set S

$$O^{CTC} = -\sum_{(x_{1:T},l_{1:V})\in S} \ln p(l_{1:V}|x_{1:T}) \tag{2.96}$$

which can be calculated directly from Equation 2.95. An RNN (or LSTM) with a CTC output layer can be trained with gradient descent by backpropagating through time the following partial derivatives of O^{CTC} with respect to the output activations:

$$\frac{\partial O^{CTC}}{\partial o_t^k} = \frac{-1}{p(l_{1:V}|x_{1:T})o_t^k} \sum_{v\in lab(l_{1:V},k)} \alpha_t(v)\beta_t(v), \tag{2.97}$$

where $lab(l_{1:V}, k)$ is the set of positions in $l_{1:V}$ where the label k occurs (see [90] for a detailed derivation).

When a new input sequence is presented to a network trained with CTC, the output activations (corresponding to the label probabilities) tend to form single

frame *spikes* separated by long intervals where the blank label is emitted. The location of the spikes corresponds to the portion of the input sequence where the label is detected.

3

Verbal Behavior Analysis

In this chapter, we will focus on verbal behavior analysis as needed in human-machine interfaces supporting speech-based interaction. Applying suitable speech features such as those introduced in Section 2.2, this chapter proposes novel, context-sensitive, and robust machine learning methods for the extraction of the spoken content in a user's utterance. First, in Section 3.1, we will concentrate on the detection of keywords which is necessary for many dialogue systems using a specific inventory of important words to infer the intention or the state of the user. Since the SEMAINE system (see Section 2.1) represents the major use case for the keyword spotting techniques introduced in this chapter, the evaluations of the proposed systems will mostly consider spontaneous, conversational, emotional, and partly disfluent and noisy speech which is typical for the SEMAINE scenario. Next, in Section 3.2, we will investigate advanced techniques for continuous recognition of conversational speech. Four different recognition frameworks exploiting long-range temporal context modeling via bidirectional Long Short-Term Memory will be introduced and evaluated. Section 3.3 is devoted to strategies for enhancing the noise robustness of automatic speech recognition. We will review popular techniques, such as feature enhancement based on Histogram Equalization (HEQ) or Switching Linear Dynamic Models (SLDM) and multi-condition training, before we draw our attention to novel robust recognition engines applying Non-Negative Matrix Factorization (NMF) as well as Long Short-Term Memory.

3.1 Vocabulary Independent Keyword Detection

Speech interfaces allowing for hands-free and natural human-machine communication have become an integral part of modern human-computer interaction [4, 265]. Yet, since full natural language understanding is far beyond the capabilities of today's conversational agents, speech interpretation modules of dialogue systems tend to evaluate certain relevant *keywords* rather than using the full automatic speech

recognition output in order to generate responses or take actions [37, 168, 180, 206, 207, 255]. Thus, for many applications it is more important to reliably detect keywords than to process the full transcript of the spoken utterance. For example the 'Sensitive Artificial Listeners' implemented in the SEMAINE system [206] (see Section 2.1) aim to infer the emotional state of the user from keywords detected in the user's speech. Other examples of speech-based human-machine interaction exploiting recognized keywords include conversational agents for food-ordering [255], systems for multimodal interaction in virtual environments [168], interactive storytelling frameworks [37], and systems for tracking conversation topics and fostering group conversations [180].

The aim of keyword spotting is to detect a set of predefined keywords from continuous speech signals [195]. If keyword detectors need to be flexible with respect to changes in the keyword vocabulary or if applications require the detection of certain terms or names that are not part of our everyday speech, it is often not adequate or not possible to apply standard large vocabulary continuous speech recognition (LVCSR) systems employing language models to capture the keywords. For example, if certain proper nouns (names of persons, cities, etc.) are to be detected in continuous speech signals, often only the phonemizations of the names are known to the system, while LVCSR language model likelihoods are not available. This makes *vocabulary independent* systems very popular [151, 230, 278] – i. e., systems that differ from conventional large vocabulary continuous speech recognition systems in that they do not rely on a language model but exclusively on acoustic evidence in combination with the known keyword phonemization. In this section we focus on techniques that (unlike classical spoken term detection systems [248]) do not require word lattices, which in turn are generated considering language model scores, but exclusively apply acoustic models together with feature-level contextual information.

At present, most keyword detection systems apply Hidden Markov Models and capture both, keywords and arbitrary speech segments (i. e., *garbage* speech) either via whole-word models for keywords and garbage speech, or by using connected phoneme models [124, 196]. Systems applying whole-word models are inherently not vocabulary independent since they presume that the modeled keywords frequently occur in the training database. Designing appropriate garbage models that capture arbitrary non-keyword speech is challenging since a model that is flexible enough to model any possible phoneme sequence can potentially also model phoneme sequences that correspond to keywords.

A popular approach towards improving acoustic modeling within ASR and word spotting systems is to combine conventional hidden Markov modeling with neural networks [192, 247]. Such techniques can be categorized into *hybrid* model architectures which apply neural networks to estimate the state-posterior probabilities of HMMs, and *Tandem* systems which use state- or phoneme posterior probabilities generated by a neural network as features observed by an HMM [310]. Both approaches offer a number of advantages when compared to conventional Gaussian

mixture modeling of low-level features such as Mel-Frequency Cepstral Coefficients: Neural networks do not make assumptions about the statistical distribution that has to be modeled which leads to more accurate acoustic models, given sufficient learning material. Furthermore, they provide an easy method for discriminative training and they use shared parameters to model all probability distributions (see Section 2.3.6).

Due to co-articulation effects in human speech, the integration of context modeling into hybrid and Tandem ASR systems is an active area of research [38, 94, 108]. Especially if high-level contextual knowledge in the form of a language model is not available, it is important to capture lower-level context in order to enable robust phoneme recognition. While the integration of delta features and the application of triphone acoustic models is a common strategy to consider low-level context, some recent studies on Tandem systems account for a fixed amount of context by stacking a predefined number of past and future feature vectors before processing them via Multilayer Perceptrons [94]. Other systems apply *recurrent* neural networks (see Section 2.3.7) to consider neighboring feature frames for prediction [171]. However, as detailed in Section 2.3.9, the context an RNN can model is known to be limited to about ten frames due to the *vanishing gradient problem* [110]. An elegant way to overcome the vanishing gradient problem was introduced in [111] and in Section 2.3.9: Long Short-Term Memory networks are able to model a self-learned amount of (long-range) temporal context. Thus, LSTM or bidirectional LSTM networks are a promising technique for improving context modeling within LVCSR and keyword spotting systems.

The first attempt towards BLSTM-based keyword spotting using whole-word modeling was presented in [75]. This section introduces several novel vocabulary independent keyword spotting techniques that apply the principle of Graphical Models, Long Short-Term Memory, and Connectionist Temporal Classification [90] (see Sections 2.3.2, 2.3.9, and 2.3.10). First, in Section 3.1.1, we focus on a discriminative approach towards keyword detection [123] and investigate how its performance can be enhanced via BLSTM modeling [275]. Next, in Section 3.1.2, a Graphical Model framework for vocabulary independent keyword spotting is introduced [278]. Section 3.1.3 shows, how this GM architecture can be extended to a Tandem model for improved context-sensitive keyword detection [273]. Finally, two CTC-based keyword spotting approaches are outlined in Sections 3.1.4 and 3.1.5 [280, 297]. By employing BLSTM networks with CTC output layers, these keyword detection frameworks do not need (potentially error-prone) phoneme-level forced alignments of speech data for training, but can be trained on unsegmented data. Phoneme detection spikes generated by the CTC network are processed by a flexible Graphical Model architecture building on recently introduced GM decoders [272, 278]. Thus, the system combines the high level flexibility of Graphical Models and Dynamic Bayesian Networks with the low-level signal processing power of BLSTM-CTC networks.

Section 3.1.6 compares the performance of the different keyword spotting ap-

proaches on both, read and spontaneous speech. In conformance with experiments shown in [123], the well-known TIMIT corpus [80] is used for evaluations on read speech. As our main motivation behind the design of flexible keyword detection approaches is their application in conversational agent scenarios such as the SEMAINE system, we also consider the SEMAINE database [155] for evaluations on spontaneous and emotional speech. This scenario is considerably more challenging, since it involves disfluent, conversational, and affective speaking styles which are known to be difficult to recognize.

3.1.1 Discriminative Keyword Spotting Exploiting BLSTM

As argued in [123], the common approach of using Hidden Markov Models for keyword spotting involves several drawbacks such as the suboptimal convergence of the expectation maximization algorithm to local maxima, the assumption of conditional independence of the observations, and the fact that HMMs do not directly maximize the keyword detection rate. For these reasons the keyword detector outlined in this section follows [122] in using a supervised, discriminative approach to keyword spotting, that does not require the use of HMMs. In general, discriminative learning algorithms are likely to outperform generative models such as HMMs since the objective function used during training more closely reflects the actual decision task. The discriminative method described in [122] uses feature functions to nonlinearly map the speech utterance, along with the target keyword, into an abstract vector space. It was shown to prevail over HMM modeling. However, in contrast to state-of-the-art HMM recognizers which use triphones to incorporate information from past and future speech frames, the discriminative system does not explicitly consider contextual knowledge. This section shows how context information can be built into a discriminative keyword spotter by including the outputs of a bidirectional Long Short-Term Memory RNN in the feature functions. In contrast to [75], this keyword spotting approach uses BLSTM for phoneme discrimination and not for the recognition of whole keywords. As well as reducing the complexity of the network, the use of phonemes makes the technique applicable to any vocabulary independent keyword spotting task.

Discriminative Keyword Spotting

The goal of the discriminative keyword spotter investigated in this section is to determine the likelihood that a specific keyword is uttered in a given speech sequence. It is assumed that each keyword k consists of a phoneme sequence $q^k_{1:L}$ with L being the length of the sequence and q denoting a phoneme out of the domain $\mathcal{P}$ of possible phoneme symbols. The speech signal is represented by a sequence of feature vectors $x_{1:T}$ where T is the length of the utterance. $\mathcal{X}$ and $\mathcal{K}$ mark the domain of all possible feature vectors and the lexicon of keywords respectively. Using a phoneme counter

variable τ, the alignment of the keyword phonemes is defined by the start times κ_τ of the phonemes as well as by the end time of the last phoneme ϵ_L: $\kappa^k_{1:L} = (\kappa_1, ..., \kappa_L, \epsilon_L)$. We assume that the start time of phoneme $q_{\tau+1}$ corresponds to the end time of phoneme q_τ, so that $\epsilon_\tau = \kappa_{\tau+1}$. The keyword spotter f takes as input a feature vector sequence $x_{1:T}$ as well as a keyword phoneme sequence $q^k_{1:L}$ and outputs a real valued confidence that the keyword k is uttered in $x_{1:T}$. In order to make the final decision whether k is contained in $x_{1:T}$, the confidence score is compared to a threshold δ. The confidence calculation is based on a set of n non-linear feature functions $\{\phi_j\}^n_{j=1}$ which take a sequence of feature vectors $x_{1:T}$, a keyword phoneme sequence $q^k_{1:L}$, and a suggested alignment $\kappa^k_{1:L}$ to compute a confidence measure for the candidate keyword alignment.

The keyword spotting algorithm searches for the best alignment $\kappa_{1:L}$ producing the highest possible confidence for the phoneme sequence of keyword k in $x_{1:T}$. Merging the feature functions ϕ_j to an n-dimensional vector function $\boldsymbol{\phi}$ and introducing a weight vector ω, the keyword spotter is given as

$$f(x_{1:T}, q^k_{1:L}) = \max_{\kappa_{1:L}} \omega \cdot \boldsymbol{\phi}(x_{1:T}, q^k_{1:L}, \kappa_{1:L}). \tag{3.1}$$

Consequently, f outputs a weighted sum of feature function scores maximized over all possible keyword alignments. This output then corresponds to the confidence that the keyword k is uttered in the speech feature sequence $x_{1:T}$. Since the number of possible alignments is exponentially large, the maximization is calculated using dynamic programming.

In order to evaluate the performance of a keyword spotter, it is common to compute the Receiver Operating Characteristics (ROC) curve [16, 124] which shows the true positive rate as a function of the false positive rate. The operating point on this curve can be adjusted by changing the keyword rejection threshold δ. If a high true positive rate shall be obtained at a preferably low false positive rate, the area under the ROC curve (AUC) has to be maximized. With $\mathcal{X}^+_k$ denoting a set of utterances that contains the keyword k and $\mathcal{X}^-_k$ a set that does not contain the keyword respectively, the AUC for keyword k is calculated as according to [44] as

$$A_k = \frac{1}{|\mathcal{X}^+_k||\mathcal{X}^-_k|} \sum_{\substack{x^+_{1:T} \in \mathcal{X}^+_k \\ x^-_{1:T} \in \mathcal{X}^-_k}} \mathbb{I}_{\{f(x^+_{1:T}, q^k_{1:L}) > f(x^-_{1:T}, q^k_{1:L})\}} \tag{3.2}$$

and can be thought of as the probability that an utterance containing keyword k ($x^+_{1:T}$) produces a higher confidence than a sequence in which k is not uttered ($x^-_{1:T}$). Here, $\mathbb{I}_{\{\cdot\}}$ denotes the indicator function. When speaking of the average AUC, we refer to

$$A = \frac{1}{\mathcal{K}} \sum_{k \in \mathcal{K}} A_k. \tag{3.3}$$

In [122] an algorithm for the computation of the weight vector ω in Equation 3.1 is presented. The algorithm aims at training the weights ω in a way that they maximize the average AUC on unseen data. One training example $\{q_{1:L}^{k_i}, x_{1:T,i}^{+}, x_{1:T,i}^{-}, \kappa_{1:L,i}^{k_i}\}$ consists of an utterance in which keyword k_i is uttered, one sequence in which the keyword is not uttered, the phoneme sequence of the keyword, and the correct alignment of k_i. With

$$\kappa'_{1:L} = \arg\max_{\kappa_{1:L}} \omega_{i-1} \cdot \boldsymbol{\phi}(x_{1:T,i}^{-}, q_{1:L}^{k_i}, \kappa_{1:L}) \tag{3.4}$$

representing the most probable alignment of k_i in $x_{1:T,i}^{-}$ according to the weights ω_{i-1} of the previous training iteration $i-1$, a term

$$\Delta\boldsymbol{\phi}_i = \frac{1}{|\mathcal{X}_{k_i}^{+}||\mathcal{X}_{k_i}^{-}|}\big(\boldsymbol{\phi}(x_{1:T,i}^{+}, q_{1:L}^{k_i}, \kappa_{1:L}^{k_i}) - \boldsymbol{\phi}(x_{1:T,i}^{-}, q_{1:L}^{k_i}, \kappa'_{1:L})\big) \tag{3.5}$$

is computed which is the difference of feature functions for $x_{1:T,i}^{+}$ and $x_{1:T,i}^{-}$. For the update rule of ω the Passive-Aggressive algorithm for binary classification (PA-I) outlined in [51] is applied. Consequently, ω is updated according to

$$\omega_i = \omega_{i-1} + \alpha_i \Delta\boldsymbol{\phi}_i, \tag{3.6}$$

where α_i can be calculated as

$$\alpha_i = \min\left\{C^u, \frac{[1 - \omega_{i-1} \cdot \Delta\boldsymbol{\phi}_i]_+}{||\Delta\boldsymbol{\phi}_i||^2}\right\}. \tag{3.7}$$

The parameter C^u controls the *aggressiveness* of the update rule and $[1-\omega_{i-1}\cdot\Delta\boldsymbol{\phi}_i]_+$ can be interpreted as the *loss* suffered on iteration i. After every training step the AUC on a validation set is computed whereas the vector ω which achieves the best AUC on the validation set is the final output of the algorithm.

Feature Functions

As can be seen in Equation 3.1, the keyword spotter is based on a set of non-linear feature functions $\{\phi_j\}_{j=1}^{n}$ that map a speech utterance, together with a candidate alignment, into an abstract vector space. In the following, $n = 7$ feature functions which proved successful for the keyword spotter introduced in [121] are used. Yet, in order to enhance the framewise phoneme estimates used in the first feature function ϕ_1, the output activations of a BLSTM network for phoneme prediction are included into ϕ_1. One variant is to extend ϕ_1 to a two-dimensional function, giving an overall feature function dimension of $n = 8$. In what follows five versions of the first feature function, denoted ϕ_{1A} - ϕ_{1E}, are described (also see [275]).

Feature function ϕ_{1A} is the same as used in [122] and is based on the hierarchical phoneme classifier described in [55]. The classifier outputs a confidence $h_q(x_{1:T})$ that

phoneme q is pronounced in $x_{1:T}$ which is then summed over the whole phoneme sequence to give

$$\phi_{1A}(x_{1:T}, q_{1:L}, \kappa_{1:L}) = \sum_{\tau=1}^{L} \sum_{t=\kappa_\tau}^{\kappa_{\tau+1}-1} h_{q_\tau}(x_{1:T}). \tag{3.8}$$

Unlike ϕ_{1A}, the feature function ϕ_{1B} incorporates contextual information for the computation of the phoneme probabilities by replacing the confidences $h_q(x_{1:T})$ by the BLSTM output activations $o_q(x_{1:T})$, thus

$$\phi_{1B}(x_{1:T}, q_{1:L}, \kappa_{1:L}) = \sum_{\tau=1}^{L} \sum_{t=\kappa_\tau}^{\kappa_{\tau+1}-1} o_{q_\tau}(x_{1:T}). \tag{3.9}$$

Since the BLSTM outputs tend to produce high-confidence phoneme probability distribution spikes for the recognized phoneme of a frame while all other activations are close to zero, it is beneficial to also include the probability distribution $h_q(x_{1:T})$ (which – due to the hierarchical structure of the classifier – consists of multiple rather low-confidence spikes) in the first feature function. Therefore, ϕ_{1C} expands the first feature function to a two-dimensional function which can be written as

$$\phi_{1C}(x_{1:T}, q_{1:L}, \kappa_{1:L}) = \begin{pmatrix} \sum_{\tau=1}^{L} \sum_{t=\kappa_\tau}^{\kappa_{\tau+1}-1} h_{q_\tau}(x_{1:T}) \\ \sum_{\tau=1}^{L} \sum_{t=\kappa_\tau}^{\kappa_{\tau+1}-1} o_{q_\tau}(x_{1:T}) \end{pmatrix}. \tag{3.10}$$

Alternatively, ϕ_{1D} consists of a linear combination of the distributions $h_q(x_{1:T})$ and $o_q(x_{1:T})$ so that

$$\phi_{1D}(x_{1:T}, q_{1:L}, \kappa_{1:L}) = \sum_{\tau=1}^{L} \sum_{t=\kappa_\tau}^{\kappa_{\tau+1}-1} \lambda_h \cdot h_{q_\tau}(x_{1:T}) + \lambda_o \cdot o_{q_\tau}(x_{1:T}), \tag{3.11}$$

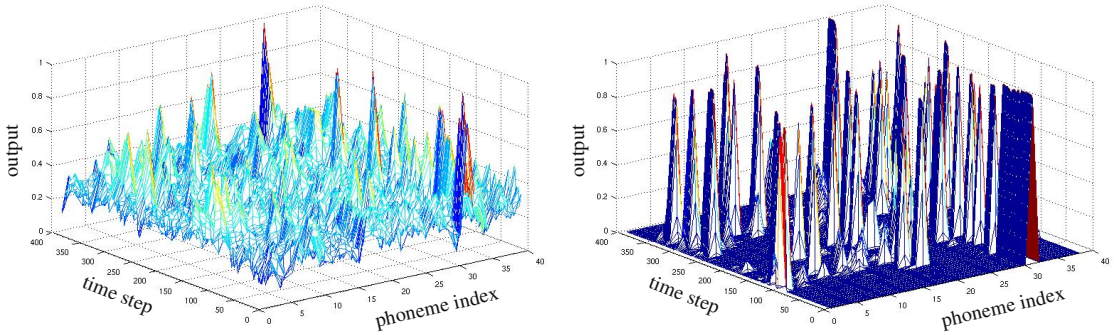
with λ_h denoting the weight of the hierarchical classifier and λ_o corresponding to the weight of the BLSTM output.

The function ϕ_{1E} takes the maximum of the distributions $h_q(x_{1:T})$ and $o_q(x_{1:T})$. This maintains the high-confidence BLSTM output activations as well as the multiple rather low-confidence hypotheses of $h_q(x_{1:T})$ for q-t coordinates where $o_q(x_t)$ is close to zero:

$$\phi_{1E}(x_{1:T}, q_{1:L}, \kappa_{1:L}) = \sum_{\tau=1}^{L} \sum_{t=\kappa_\tau}^{\kappa_{\tau+1}-1} \max\left(h_{q_\tau}(x_{1:T}), o_{q_\tau}(x_{1:T})\right). \tag{3.12}$$

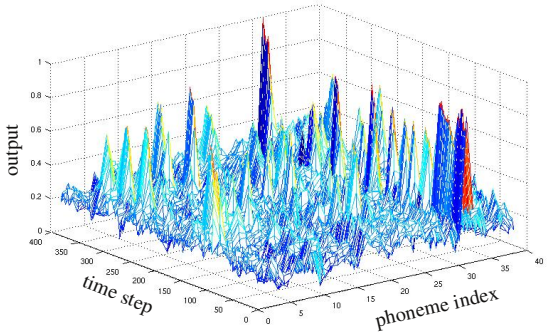
Figures 3.1(a) to 3.1(d) show the outputs of feature functions ϕ_{1A}, ϕ_{1B}, ϕ_{1D}, and ϕ_{1E} over time for an example utterance and a phoneme inventory of size 39.

The remaining feature functions ϕ_2 - ϕ_7 used in this section are the same as in [122]. ϕ_2 - ϕ_5 measure the Euclidean distance between feature vectors at both

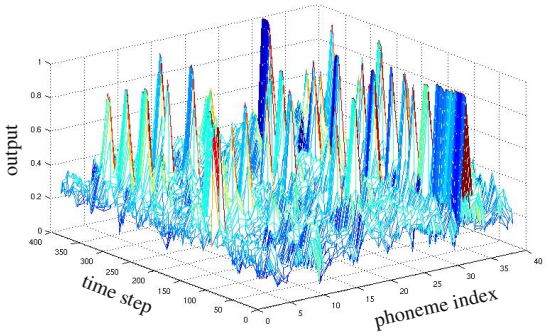


(a) Feature function ϕ_{1A}

(b) Feature function ϕ_{1B}



(c) Feature function ϕ_{1D}



(d) Feature function ϕ_{1E}

Figure 3.1: Phoneme confidences over time for an example utterance when using different feature functions based on a hierarchical phoneme classifier and/or a BLSTM phoneme classifier. ϕ_{1D} is normalized and uses weights $\lambda_h = 1.0$ and $\lambda_o = 1.5$.

sides of the suggested phoneme boundaries, assuming that the correct alignment will produce a large sum of distances, since the distances at the phoneme boundaries are likely to be high compared to those within a phoneme. Function ϕ_6 scores the timing sequences based on typical phoneme durations and ϕ_7 considers the speaking rate implied with the candidate phoneme alignment, presuming that the speaking rate changes only slowly over time (see [122] for formulas).

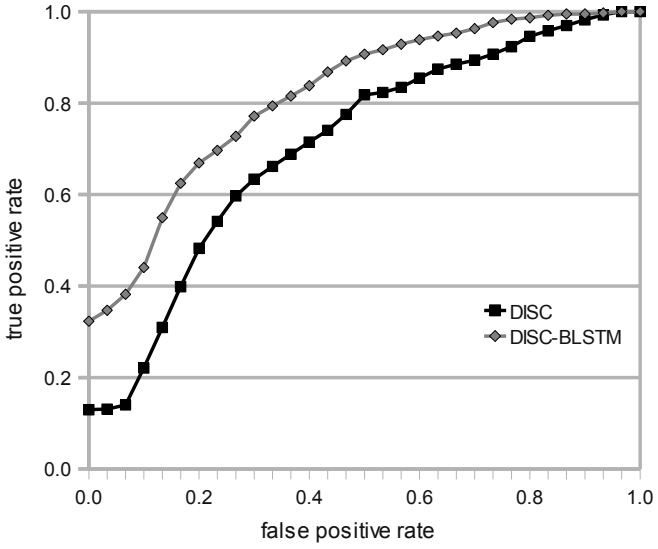


Figure 3.2: ROC curve for the discriminative keyword spotter (DISC) based on feature function ϕ_{1A} as introduced in [123] and the keyword spotter enhanced via BLSTM modeling based on ϕ_{1D} (DISC-BLSTM). Evaluation on the SAL database.

Experiments and Results

To compare the performance of the discriminative keyword spotter proposed in [122] and [123] with a keyword spotter enhanced via BLSTM modeling of phonemes, the TIMIT corpus and its framewise phoneme annotations was used as a training database. As preliminary experiments in [275] revealed that the most effective way to incorporate BLSTM context modeling into the first feature function is to use a linear combination of the phoneme estimation scores produced by the hierarchical classifier $h_q(\cdot)$ and the BLSTM phoneme predictor $o_q(\cdot)$ with weights $\lambda_h = 1.0$ and $\lambda_o = 1.5$, we focus on feature function ϕ_{1D} (see Equation 3.11) and compare it with feature function ϕ_{1A} which uses no contextual information in the form of BLSTM phoneme estimates. The TIMIT training set was divided into five parts. 1 500 utterances were used to train the frame-based phoneme recognizer needed for the first feature function. 150 utterances served as training set for the forced alignment algorithm which was applied to initialize the weight vector ω (for details see [121]). 100 sequences formed the validation set of the forced aligner, and from the remaining 1 946 utterances two times 200 samples (200 positive and 200 negative utterances) were selected for training and two times 200 utterances for validation of the discriminative keyword spotter. The feature vectors consisted of cepstral mean

normalized MFCC features 0 to 12 with first and second order delta coefficients. As aggressiveness parameter C^u for the update algorithm (see Equation 3.7) $C^u = 1$ was used. For the training of the BLSTM used for feature function ϕ_{1D}, the same 1 500 utterances as for the phoneme recognizer of ϕ_{1A} were chosen, however, they were split into 1 400 sequences for training and 100 for validation. The BLSTM input layer had a size of 39 (one for each MFCC feature) and the size of the output layer was also 39 since the reduced set of 39 TIMIT phonemes was used. Both hidden LSTM layers contained 100 memory blocks of one cell each. To improve generalization, zero mean Gaussian noise with standard deviation 0.6 was added to the inputs during training. The applied learning rate was 10^{-5} and the momentum was 0.9.

The effect of replacing feature function ϕ_{1A} with ϕ_{1D} was evaluated on the Belfast Sensitive Artificial Listener (SAL) database [64] containing spontaneous and emotionally colored speech. For a more detailed description of the SAL database, see [64] or [276]. 24 keywords were randomly chosen. For each keyword 20 utterances in which the keyword is not uttered and up to 20 utterances (depending on how often the keyword occurs in the whole corpus) which include the keyword were selected. On average, a keyword consisted of 5.4 phonemes. As can be seen in Figure 3.2, the BLSTM approach (using ϕ_{1D}) is able to outperform the keyword spotter which does not use long-range dependencies via BLSTM output activations. The average AUC is 0.80 for the BLSTM experiment and 0.68 for the experiment using the original feature function ϕ_{1A}, respectively.

This result can be interpreted as a first indication that bidirectional Long Short-Term Memory modeling can enhance the performance of keyword detection in spontaneous emotional speech. A detailed comparison of the two discriminative approaches and the (partly generative) techniques outlined in Sections 3.1.2 to 3.1.5 can be found in Section 3.1.6.

3.1.2 Graphical Models for Keyword Detection

Hidden Markov Model based keyword spotting systems [124, 196] usually require keyword HMMs and a *filler* or *garbage* HMM to model both, keywords and non-keyword parts of the speech sequence. Using whole word HMMs for the keywords and the garbage model presumes that there are enough occurrences of the keywords in the training corpus and suffers from low flexibility since new keywords cannot be added to the system without having to re-train it. Modeling sub-units of words, such as phonemes, offers the possibility to design a garbage HMM that connects all phoneme models [196]. However, the inherent drawback of this approach is that the garbage HMM can potentially model any phoneme sequence, including the keyword itself. Better garbage models can be trained when modeling non-keyword speech with a large vocabulary ASR system where the lexicon excludes the keyword [259]. Disadvantages of this method are its higher decoding complexity and the large amount

of required training data to obtain a reasonable language model. Moreover, such LVSCR systems presume that all keywords are contained in the language model, which makes them less flexible than vocabulary independent systems where no information about the set of keywords is required while training the models.

In this section, a new Graphical Model design which can be used for robust keyword spotting and overcomes most of the drawbacks of other approaches is introduced. Graphical Models offer a flexible statistical framework that is increasingly applied for speech recognition tasks [21, 22] since it allows for conceptual deviations from the conventional HMM architecture. As outlined in Section 2.3.2, a GM – or, more specifically, a DBN – makes use of the graph theory in order to describe the time evolution of speech as a statistical process and defines conditional independence properties of the observed and hidden variables that are involved in the process of speech decoding. Apart from common HMM approaches, there exist only a small number of methods which try to address the task of keyword spotting using the general Graphical Model paradigm. In [144], a Graphical Model is applied for spoken keyword spotting based on performing a joint alignment between the phone lattices generated from a spoken query and a long stored utterance. This concept, however, is optimized for offline phone lattice generation and bears no similarity to the technique proposed in this section. The same holds for approaches towards GM based out-of-vocabulary (OOV) detection [143] where a Graphical Model indicates possible OOV regions in continuous speech.

In the following, the explicit graph representation of a GM based keyword spotter is introduced. The GM does not need a trained garbage model and is robust with respect to phoneme recognition errors. The approach is conceptually more simple than a large vocabulary ASR system since it does not require a language model but only the keyword phonemizations. By adding a further hierarchy level to a Dynamic Bayesian Network for phoneme recognition, we derive a framework for reliably detecting keywords in continuous speech. The method uses a hidden *garbage variable* and the concept of *switching parents* [21] to model either a keyword or arbitrary speech (also see [278]). DBNs are the Graphical Models of choice for speech recognition tasks, since they consist of repeated template structures over time, modeling the temporal evolution of a speech sequence. Conventional HMM approaches can be interpreted as *implicit* graph representations using a single Markov chain together with an integer state to represent all contextual and control information determining the allowable sequencing. In this section, however, we focus on the *explicit* approach, where information such as the current phoneme, the indication of a phoneme transition, or the position within a word is expressed by random variables. As shown in [22], explicit graph representations are advantageous whenever the set of hidden variables has factorization constraints or consists of multiple hierarchies.

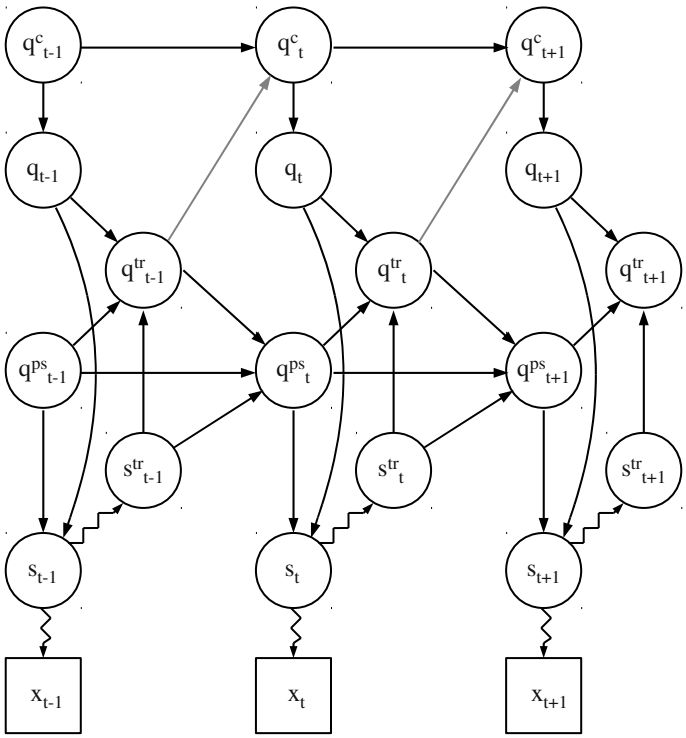


Figure 3.3: DBN structure of the Graphical Model used to train the keyword spotter.

Training

The DBN used to train the keyword spotter is depicted in Figure 3.3. Compared to the DBN that will be applied for decoding, the DBN for the training of the keyword spotter is less complex, since so far, only phonemes are modeled. The training procedure is split up into two stages: In the first stage, phonemes are trained framewise, whereas during the second stage, the segmentation constraints are relaxed using a forced alignment (embedded training).

In conformance with Figure 3.3, the following random variables are defined for every time step t: q_t^c is a count variable determining the current position in the phoneme sequence, q_t denotes the phoneme identity, q_t^{ps} represents the position within the phoneme, q_t^{tr} indicates a phoneme transition, s_t is the current state with s_t^{tr} indicating a state transition, and x_t denotes the observed acoustic features. Following the notation introduced in Sections 2.3.2 and 2.3.3, Figure 3.3 displays hidden variables as circles and observed variables as squares. Deterministic conditional

probability functions are represented by straight lines, whereas zig-zagged lines correspond to random CPFs. The grey-shaded arrow in Figure 3.3, pointing from q_{t-1}^{tr} to q_t^c is only valid during the second training cycle when there are no segmentation constraints, and will be ignored in Equations 3.14 and 3.15. Assuming a speech sequence of length T, the DBN structure specifies the factorization

$$\begin{aligned} &p(q_{1:T}^c, q_{1:T}, q_{1:T}^{tr}, q_{1:T}^{ps}, s_{1:T}^{tr}, s_{1:T}, x_{1:T}) = \\ &\quad p(x_1|s_1) f(s_1|q_1^{ps}, q_1) p(s_1^{tr}|s_1) f(q_1^{tr}|q_1^{ps}, q_1, s_1^{tr}) f(q_1|q_1^c) f(q_1^{ps}) f(q_1^c) \\ &\quad \times \prod_{t=2}^{T} p(x_t|s_t) f(s_t|q_t^{ps}, q_t) p(s_t^{tr}|s_t) f(q_t^{tr}|q_t^{ps}, q_t, s_t^{tr}) f(q_t|q_t^c) f(q_t^{ps}|s_{t-1}^{tr}, q_{t-1}^{ps}, q_{t-1}^{tr}) \\ &\quad \times f(q_t^c|q_{t-1}^c). \end{aligned} \tag{3.13}$$

Equation 3.13 can be simplified, yielding a more compact representation

$$\begin{aligned} &p(q_{1:T}^c, q_{1:T}, q_{1:T}^{tr}, q_{1:T}^{ps}, s_{1:T}^{tr}, s_{1:T}, x_{1:T}) = \\ &\quad f(q_1^{ps}) f(q_1^c) \prod_{t=1}^{T} p(x_t|s_t) f(s_t|q_t^{ps}, q_t) p(s_t^{tr}|s_t) f(q_t^{tr}|q_t^{ps}, q_t, s_t^{tr}) f(q_t|q_t^c) \\ &\quad \times \prod_{t=2}^{T} f(q_t^{ps}|s_{t-1}^{tr}, q_{t-1}^{ps}, q_{t-1}^{tr}) f(q_t^c|q_{t-1}^c) \end{aligned} \tag{3.14}$$

with $p(\cdot)$ denoting random conditional probability functions and $f(\cdot)$ describing deterministic CPFs. The probability of the observed sequence can then be computed as

$$p(x_{1:T}) = \sum_{q_{1:T}^c, q_{1:T}, q_{1:T}^{tr}, q_{1:T}^{ps}, s_{1:T}^{tr}, s_{1:T}} p(q_{1:T}^c, q_{1:T}, q_{1:T}^{tr}, q_{1:T}^{ps}, s_{1:T}^{tr}, s_{1:T}, x_{1:T}). \tag{3.15}$$

The factorization property given in Equation 3.14 is exploited in order to optimally distribute the sums over the hidden variables into the products (see Section 2.3.2). To this end, we apply the junction tree algorithm [115] to move the sums as far to the right as possible which reduces computational complexity. The CPFs $p(x_t|s_t)$ are described by Gaussian mixtures as common in an HMM system. Both, $p(x_t|s_t)$ and $p(s_t^{tr}|s_t)$ are learnt via EM training. s_t^{tr} is a binary variable, indicating whether a state transition takes place or not. Since the current state is known with certainty, given the phoneme and the phoneme position, $f(s_t|q_t^{ps}, q_t)$ is purely deterministic. A phoneme transition occurs whenever $s_t^{tr} = 1$ and $q_t^{ps} = S$ provided that S denotes the number of states of a phoneme. This is expressed by the function $f(q_t^{tr}|q_t^{ps}, q_t, s_t^{tr})$.

During training, the current phoneme q_t is known, given the position q_t^c in the training utterance, which implies a deterministic mapping $f(q_t|q_t^c)$. In the first training cycle q_t^c is incremented in every time frame, whereas in the second cycle q_t^c is only incremented if $q_{t-1}^{tr} = 1$. The phoneme position q_t^{ps} is known with certainty if s_{t-1}^{tr}, q_{t-1}^{ps}, and q_{t-1}^{tr} are given.

Decoding

Once the distributions $p(x_t|s_t)$ and $p(s_t^{tr}|s_t)$ are trained, a more complex GM is used for keyword spotting (see Figure 3.4): In the decoding phase, the hidden variables w_t, w_t^{ps}, and w_t^{tr} are included in order to model whole words. Further, a hidden *garbage variable* g_t indicates whether the current word is a keyword or not. In Figure 3.4, dotted lines correspond to so-called *switching parents* [21], which allow a variable's parents to change conditioned on the current value of the switching parent. A switching parent cannot only change the set of parents but also the implementation (i. e., the CPF) of a parent. Considering all statistical independence assumptions, the DBN can be factorized as follows:

$$\begin{aligned}
&p(g_{1:T}, w_{1:T}, w_{1:T}^{tr}, w_{1:T}^{ps}, q_{1:T}, q_{1:T}^{tr}, q_{1:T}^{ps}, s_{1:T}^{tr}, s_{1:T}, x_{1:T}) = \\
&\quad f(q_1^{ps})p(q_1|w_1^{ps}, w_1, g_1)f(w_1^{ps})p(w_1) \\
&\quad \times \prod_{t=1}^{T} p(x_t|s_t)f(s_t|q_t^{ps}, q_t)p(s_t^{tr}|s_t)f(q_t^{tr}|q_t^{ps}, q_t, s_t^{tr})f(g_t|w_t)f(w_t^{tr}|q_t^{tr}, w_t^{ps}, w_t) \\
&\quad \times \prod_{t=2}^{T} f(q_t^{ps}|s_{t-1}^{tr}, q_{t-1}^{ps}, q_{t-1}^{tr})p(q_t|q_{t-1}^{tr}, q_{t-1}, w_t^{ps}, w_t, g_t)f(w_t^{ps}|q_{t-1}^{tr}, w_{t-1}^{ps}, w_{t-1}^{tr}) \\
&\quad \times p(w_t|w_{t-1}^{tr}, w_{t-1}).
\end{aligned} \tag{3.16}$$

The hidden variable w_t can take values in the range $w_t = 0...K$ with K being the number of different keywords in the vocabulary. In case $w_t = 0$ the model is in the *garbage state* which means that no keyword is uttered at that time. The variable g_t is then equal to one. w_{t-1}^{tr} is a switching parent of w_t: If no word transition is indicated, w_t is equal to w_{t-1}. Otherwise, a simple word bigram specifies the CPF $p(w_t|w_{t-1}^{tr} = 1, w_{t-1})$. In our experiments, the word bigram is simplified to a unigram which makes each keyword equally likely. However, differing a priori likelihoods for keywords and garbage phonemes are introduced:

$$p(w_t = 1 : K|w_{t-1}^{tr} = 1) = \frac{K \cdot 10^a}{K \cdot 10^a + 1} \tag{3.17}$$

and

$$p(w_t = 0|w_{t-1}^{tr} = 1) = \frac{1}{K \cdot 10^a + 1}. \tag{3.18}$$

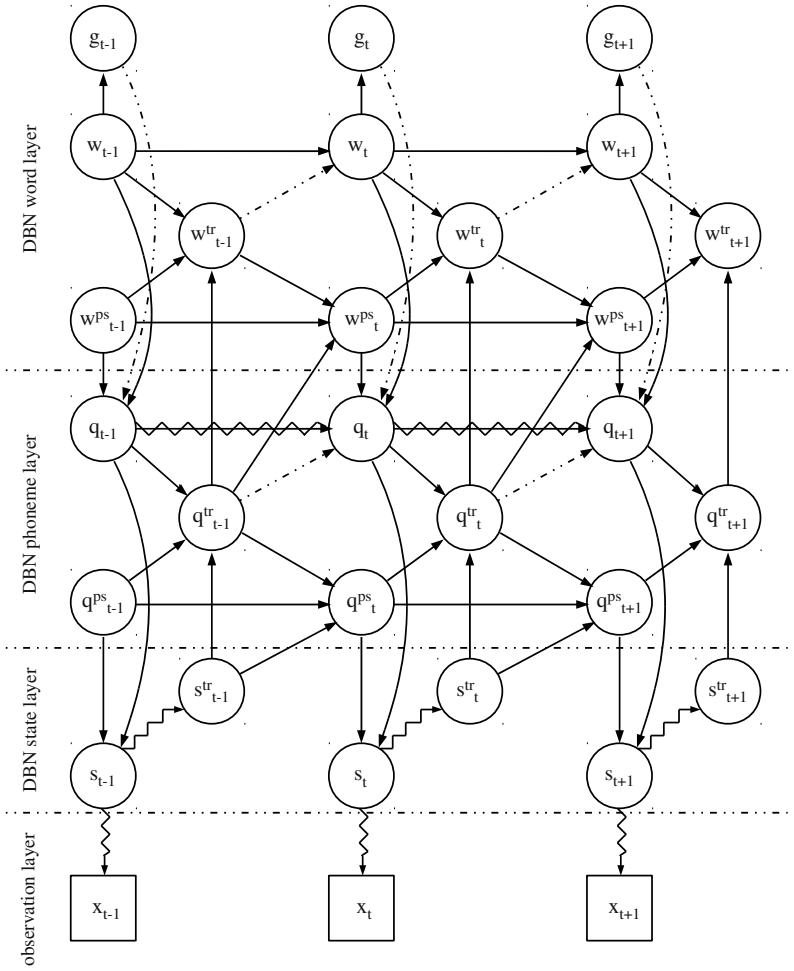


Figure 3.4: DBN structure of the Graphical Model for keyword spotting.

The parameter a can be used to adjust the trade-off between true positives and false positives. Setting $a = 0$ means that the a priori probability of a keyword and the probability that the current phoneme does not belong to a keyword are equal. Adjusting $a > 0$ implies a more aggressive search for keywords, leading to higher true positive and false positive rates. The CPFs $f(w_t^{tr}|q_t^{tr}, w_t^{ps}, w_t)$ and $f(w_t^{ps}|q_{t-1}^{tr}, w_{t-1}^{ps}, w_{t-1}^{tr})$ are similar to the phoneme layer of the GM (i. e., the CPFs for q_t^{tr} and q_t^{ps}). However, we assume that 'garbage words' always consist of only one phoneme, meaning that if $g_t = 1$, a word transition occurs as soon as $q_t^{tr} = 1$. Consequently, w_t^{ps} is always zero if the model is in the garbage state. The variable q_t has two switching parents: q_{t-1}^{tr} and g_t. Similar to the word layer, q_t is equal to q_{t-1} if $q_{t-1}^{tr} = 0$. Otherwise, the switching parent g_t determines the parents of q_t. In case

$g_t = 0$ – meaning that the current word is a keyword – q_t is a deterministic function of the current keyword w_t and the position within the keyword w_t^{ps}. If the model is in the garbage state, q_t only depends on q_{t-1} using a trained phoneme bigram P. This phoneme bigram matrix is used to model arbitrary speech and is learnt by simply counting phoneme transitions that occur in a training corpus:

$$P = N - f \cdot I. \tag{3.19}$$

The bigram matrix P contains the probabilities

$$P_{ij} = p(q_t = j | q_{t-1}^{tr} = 1, g_t = 1, q_{t-1} = i) \tag{3.20}$$

that the phoneme j occurs after phoneme i. N includes the number of phoneme transitions n_{ij}, normalized by the number N_i of occurrences of the phoneme i in the training corpus. All entries n_{ij} are floored to ξ:

$$N_{ij} = \max\left(\frac{n_{ij}}{N_i}, \frac{\xi}{N_i}\right). \tag{3.21}$$

Since Equation 3.21 introduces a probability floor value for all possible transitions, the subtraction of the identity matrix I weighted by ξ ensures that transitions from phoneme i to phoneme i occur with zero probability.

Note that the design of the CPF $p(q_t | q_{t-1}^{tr}, q_{t-1}, w_t^{ps}, w_t, g_t)$ entails that the GM will strongly tend to choose $g_t = 0$ (i. e., it will detect a keyword) once a phoneme sequence that corresponds to a keyword is observed. Decoding such an observation while being in the garbage state $g_t = 1$ would lead to 'phoneme transition penalties' since P contains probabilities less than one. By contrast, $p(q_t | q_{t-1}^{tr} = 1, w_t^{ps}, w_t, g_t = 0)$ is deterministic, introducing no likelihood penalties at phoneme borders.

Experiments and Results

The DBN for keyword spotting was trained and evaluated on the TIMIT corpus in order to enable a first insight into its performance compared to a standard HMM keyword spotter. All feature vectors consisted of cepstral mean normalized MFCC coefficients 1 to 12, log. energy, as well as first and second order regression coefficients. The phoneme models were composed of three hidden states each. During the first training cycle of the GM, phonemes were trained framewisely using the training portion of the TIMIT corpus. All Gaussian mixtures were split once 0.02% convergence was reached until the number of mixtures per state increased to 16. In the second training cycle segmentation constraints were relaxed and no further mixture splitting was conducted (embedded training). 60 keywords were randomly chosen from the TIMIT corpus to evaluate the keyword spotter DBN. The floor value ξ (see Equation 3.21) was set to 10 and the trade-off parameter a (see Equation 3.17) was varied between 0 and 10.

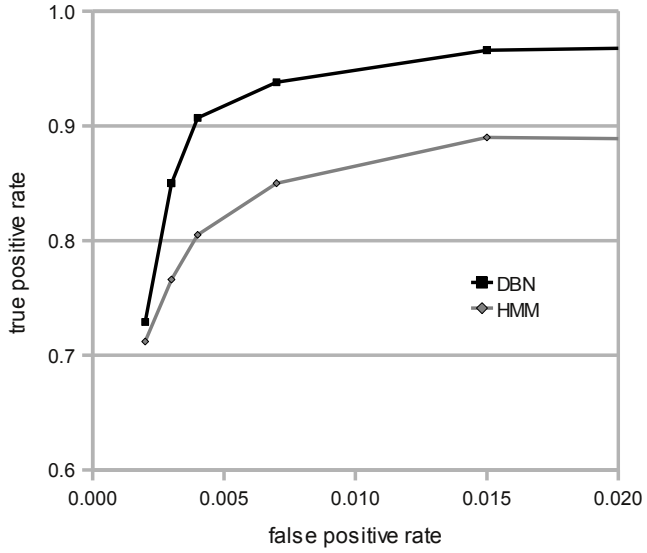


Figure 3.5: Part of the ROC curve for the DBN and for the HMM keyword spotter. Evaluation on the TIMIT database.

For comparison, a phoneme based keyword spotter using conventional HMM modeling was trained and evaluated on the same task. Analogous to the DBN experiment, each phoneme was represented by three states (left-to-right HMMs) with either 16 Gaussian mixtures. Cross-word triphone models were applied in order to account for contextual information. Like the DBN, all phoneme HMMs were re-trained using embedded training. For keyword detection a set of keyword models and a garbage model was defined. The keyword models estimate the likelihood of a feature vector sequence, given that it corresponds to the keyword phoneme sequence. The garbage model is composed of phoneme HMMs that are fully connected to each others, meaning that it can model any phoneme sequence. Via Viterbi decoding the best path through all models is found and a keyword is detected as soon as the path passes through the corresponding keyword HMM. In order to be able to adjust the operating point on the ROC curve, different a priori likelihoods are introduced for keyword and garbage HMMs, identical to the word unigram used for the Graphical Model.

Figure 3.5 shows a part of the ROC curve for the DBN keyword spotter and the HMM-based keyword spotter, displaying the true positive rate (tpr) as a function of the false positive rate (fpr). Note that due to the design of the decoder, the full ROC curve – ending at an operating point tpr=1 and fpr=1 – cannot be determined, since the model does not include a confidence threshold that can be set to an arbitrarily

low value. Due to the inherent robustness with respect to phoneme recognition errors, the DBN architecture achieves significantly higher true positive rates at equal false positive rates, compared to the standard HMM approach. One can observe a performance difference of up to 10 %. Conducting the McNemar's test reveals that the performance difference between the DBN keyword spotter and the HMM approach is statistically significant at a common significance level of 0.01.

3.1.3 Tandem BLSTM-DBN

This section shows how the Graphical Model structure presented in Section 3.1.2 can be extended to a Tandem approach that is not only based on Gaussian mixture modeling but additionally applies a recurrent neural network to provide improved phoneme predictions, which can then be incorporated into the DBN [274]. As in Section 3.1.1, the RNN uses the bidirectional Long Short-Term Memory architecture to access long-range context information along both input directions. The aim is to improve the keyword spotting accuracy of the DBN introduced in Section 3.1.2 by an additional modeling of contextual information, such as co-articulation effects, via BLSTM networks. In addition to evaluations on the TIMIT and the SAL database, a part of the experiments in this section will deal with keyword detection in children's speech. Recognition of children's speech is known to be a challenge for state-of-the-art ASR systems since acoustic and linguistic properties strongly differ from adult speech [85]. Typical differences in pitch, formant positions, and co-articulation led to the development of techniques like voice transformations and frequency warping [100, 179].

In what follows, we will apply BLSTM modeling in order to generate phoneme predictions that are decoded together with conventional speech features in a Dynamic Bayesian Network and use this principle for keyword detection in a child-robot interaction scenario (also see [293]). As the characteristics of co-articulation in children's speech strongly differ from co-articulation effects in adult speech [83], BLSTM networks are applied as an efficient method of context modeling. Children develop co-articulation skills with increasing age which leads to strong variations in the amount of temporal context that needs to be considered to capture co-articulation for context-sensitive speech feature generation and acoustic modeling [153, 190]. Thus, it seems inappropriate to manually define an inflexible, fixed amount of context, as it is commonly done when stacking multiple low-level feature frames for neural network based feature generation [94]. By contrast, modeling contextual information in children's speech via BLSTM networks allows us to *learn* the proper amount of relevant context.

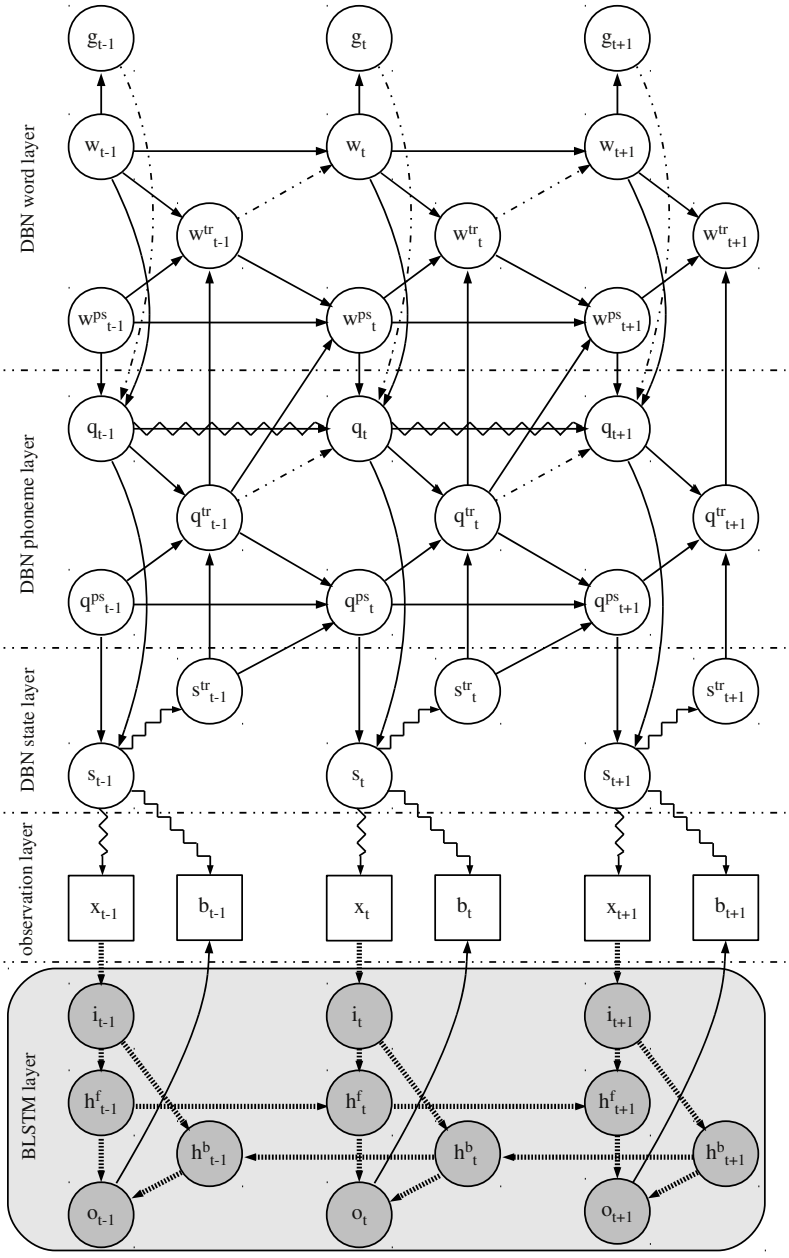


Figure 3.6: Structure of the Tandem BLSTM-DBN keyword spotter.

Model Architecture

The proposed Tandem BLSTM-DBN architecture for keyword spotting is depicted in Figure 3.6. The network is composed of five different layers and hierarchy levels

respectively: a word layer, a phoneme layer, a state layer, the observed features, and the BLSTM layer (nodes inside the grey shaded box). As can be seen in Figure 3.6, the DBN jointly processes speech features and BLSTM phoneme predictions. The BLSTM layer consists of an input layer i_t, two hidden layers h_t^f and h_t^b (one for forward and one for backward processing), and an output layer o_t. The random variables g_t, w_t, w_t^{ps}, w_t^{tr}, q_t, q_t^{ps}, q_t^{tr}, s_t, s_t^{tr}, and x_t are identical to the random variables specified for the DBN presented in Section 3.1.2. A second observed variable b_t contains the (framewise) phoneme prediction of the BLSTM. Note that the bold dashed lines in the BLSTM layer of Figure 3.6 do not represent statistical relations but simple data streams. Again, we assume a speech sequence of length T, so that the DBN structure specifies the factorization

$$
\begin{aligned}
&p(g_{1:T}, w_{1:T}, w_{1:T}^{tr}, w_{1:T}^{ps}, q_{1:T}, q_{1:T}^{tr}, q_{1:T}^{ps}, s_{1:T}^{tr}, s_{1:T}, x_{1:T}, b_{1:T}) = \\
&\quad f(q_1^{ps})p(q_1|w_1^{ps}, w_1, g_1)f(w_1^{ps})p(w_1) \\
&\quad \times \prod_{t=1}^{T} p(x_t|s_t)p(b_t|s_t)f(s_t|q_t^{ps}, q_t)p(s_t^{tr}|s_t)f(q_t^{tr}|q_t^{ps}, q_t, s_t^{tr})f(w_t^{tr}|q_t^{tr}, w_t^{ps}, w_t)f(g_t|w_t) \\
&\quad \times \prod_{t=2}^{T} f(q_t^{ps}|s_{t-1}^{tr}, q_{t-1}^{ps}, q_{t-1}^{tr})p(w_t|w_{t-1}^{tr}, w_{t-1})p(q_t|q_{t-1}^{tr}, q_{t-1}, w_t^{ps}, w_t, g_t) \\
&\quad \times f(w_t^{ps}|q_{t-1}^{tr}, w_{t-1}^{ps}, w_{t-1}^{tr}).
\end{aligned}
\tag{3.22}
$$

The size of the BLSTM input layer i_t corresponds to the dimensionality of the acoustic feature vector x_t and the vector o_t contains one probability score for each of the P different phonemes at each time step. b_t is the index of the most likely phoneme:

$$
b_t = \underset{j}{\operatorname{argmax}}(o_t^1, ..., o_t^j, ..., o_t^P). \tag{3.23}
$$

Together with $p(x_t|s_t)$ and $p(s_t^{tr}|s_t)$, the CPF $p(b_t|s_t)$ is learned using expectation maximization (see [273] for details). All other CPFs are the same as in Section 3.1.2. Again the DBN can be trained by replacing the word layer random variables with a phoneme counter variable q_t^c pointing to the phoneme ground truth in an phonetically aligned training corpus (see Figure 3.3). The factorization of the corresponding DBN for training the Tandem system can be derived similarly to Equation 3.14 as

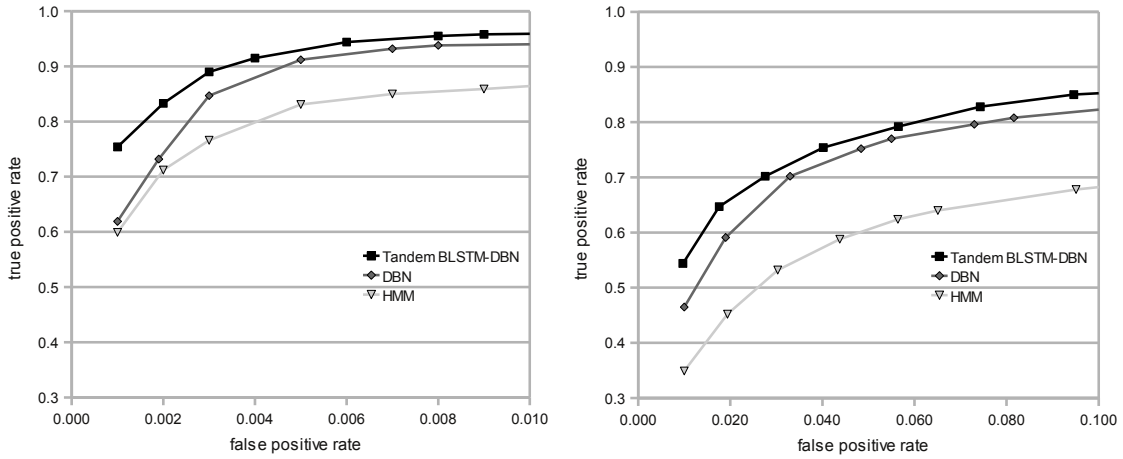
$$
\begin{aligned}
&p(q^c_{1:T}, q_{1:T}, q^{tr}_{1:T}, q^{ps}_{1:T}, s^{tr}_{1:T}, s_{1:T}, x_{1:T}, b_{1:T}) = \\
&\quad f(q^{ps}_1) f(q^c_1) \prod_{t=1}^{T} p(x_t|s_t) p(b_t|s_t) f(s_t|q^{ps}_t, q_t) p(s^{tr}_t|s_t) f(q^{tr}_t|q^{ps}_t, q_t, s^{tr}_t) f(q_t|q^c_t) \\
&\quad \times \prod_{t=2}^{T} f(q^{ps}_t|s^{tr}_{t-1}, q^{ps}_{t-1}, q^{tr}_{t-1}) f(q^c_t|q^c_{t-1}).
\end{aligned}
\tag{3.24}
$$

The BLSTM network is trained independently with standard backpropagation through time [266] using the exact error gradient as in [91].

Initial Experiments and Results

For initial evaluations of the proposed Tandem BLSTM-DBN keyword spotter on read and spontaneous adult speech, the TIMIT corpus and the SAL database were used. As in Section 3.1.2, the acoustic feature vectors consisted of cepstral mean normalized MFCC coefficients 1 to 12, log. energy, as well as first and second order delta coefficients. For the training of the BLSTM, 200 utterances of the TIMIT training split were used as validation set while the net was trained on the remaining training sequences. The BLSTM network was configured as in Section 3.1.1 and the two-stage DBN training procedure was identical to the technique detailed in Section 3.1.2, yet, prior to evaluation on the SAL corpus, all means, variances, and weights of the Gaussian mixture probability distributions $p(x_t|s_t)$, as well as the state transition probabilities $p(s^{tr}_t|s_t)$ were re-estimated using the training split of the SAL corpus. Again, re-estimation was stopped once the change of the overall log likelihood of the SAL training set fell below a threshold of 0.02 %. Details regarding the investigated keyword spotting task on the TIMIT corpus and on the SAL database as well as the configuration of the baseline HMM system can be found in Sections 3.1.1 and 3.1.2.

Figure 3.7(a) shows a part of the ROC curves for the baseline HMM, the DBN introduced in Section 3.1.2, and the Tandem BLSTM-DBN keyword spotter for the TIMIT experiment. The most significant performance gain of context modeling via BLSTM predictions occurs at an operating point with a false positive rate of 0.1 %: There, the Tandem approach can increase the true positive rate by 13.5 %, when compared to the DBN without BLSTM layer. For higher values of the trade-off parameter a (see Section 3.1.2), implying a more aggressive search for keywords, the performance gap between the DBN and the Tandem keyword spotter becomes smaller, as more phoneme confusions are tolerated when seeking for keywords. Furthermore, both DBN architectures significantly outperform the baseline HMM approach. The ROC performance for the SAL experiment can be seen in Figure 3.7(b). Obviously, the task of keyword detection in emotional speech is considerably harder, implying lower true positive rates and higher false positive rates, respectively. As



(a) Evaluation on TIMIT

(b) Evaluation on SAL

Figure 3.7: Part of the ROC curve for the baseline HMM system, the DBN keyword spotter (without BLSTM phoneme predictions) and the Tandem BLSTM-DBN approach.

for the TIMIT experiment, the Tandem BLSTM-DBN approach prevails over the DBN and the HMM baseline system with a performance gain of up to 8 % when compared to the DBN.

The FAU AIBO Emotion Corpus

To collect further evidence for achievable keyword spotting performance gains when exploiting BLSTM for context-sensitive phoneme modeling and to examine whether the improvements generalize to other challenging keyword detection tasks involving children's speech, further experiments were conducted on the FAU AIBO Emotion Corpus, a corpus of German spontaneous speech with recordings of children at the age of 10 to 13 years communicating with a pet robot [236]. The general framework for this children's speech database is child-robot communication and the elicitation of emotion-related speaker states. The robot is Sony's (dog-like) robot Aibo. The basic idea has been to combine children's speech and naturally occurring emotional speech within a Wizard-of-Oz task. The speech is spontaneous, because the children were not told to use specific instructions but to talk to Aibo like they would talk to a friend. In this experimental design, the child is led to believe that Aibo is responding to his or her commands, but the robot is actually being remote-controlled by a human operator. The wizard causes Aibo to perform a fixed, predetermined sequence of actions, which takes no account of what the child is actually saying. This obedient and disobedient behavior provokes the children in order to elicit emotional behavior.

The data was collected from 51 children (21 male, 30 female) aged 10 to 13 years from two different schools (*Mont* and *Ohm*); the recordings took place in the respective class-rooms. The total vocabulary size is 1.1 k. Each recording session took around 30 minutes; in total there are 27.5 hours of data. The recordings contain large amounts of silence, which are due to the reaction time of Aibo. After removing longer pauses, the total amount of speech is equal to 8.9 hours. All recordings were split into turns using a pause threshold of ≥ 1 s. For the (speaker-independent) keyword spotting experiments, all speech recorded at the *Ohm* school are used for training (6 370 turns), apart from two randomly selected *Ohm*-sessions which are used for validation (619 turns). The sessions recorded at the *Mont* school are used for testing (6 653 turns, see also Table 3.1).

Keywords

The keyword vocabulary consists of three different categories: words expressing positive valence, words expressing negative valence, and command words (see Table 3.2). Keywords indicating positive or negative valence were included to allow the Aibo robot to be sensitive to positive or negative feedback from the child. Such keywords can also be used as linguistic features for automatic emotion recognition [13, 14, 210, 236]. Examples are (German) words like *fein*, *gut*, *böse*, etc. (Engl.: *fine*, *good*, *bad*). Command words like *links*, *rechts*, *hinsetzen*, etc. (Engl.: *left*, *right*, *sit down*) were included so that the children are able to control the Aibo robot via speech. The dictionary contains multiple pronunciation variants as well as multiple forms of the (lemmatized) keywords listed in Table 3.2. For example the word *umdrehen* (Engl.: *turn around*) can also be pronounced as *umdrehn* and verbs do not necessarily have to be uttered in the infinitive form (e. g., *gehen* (Engl.: *go*) can also be *geh*, *gehst*, or *geht*). In order to allow a fair comparison between techniques that depend on frequent keyword occurrences in the training set (such as the CTC method introduced in [75] which will be included in our experimental study) and the vocabulary independent approaches, only those command words or emotionally relevant words that occurred at least 50 times (incl. variants) in the FAU Aibo Emotion Corpus were included in the vocabulary. In total, there are 82 different entries in the dictionary which are mapped onto exactly 25 keywords as listed in Table 3.2.

Table 3.1: Size of the training, validation, and test set: school in which the children were recorded, number of turns, number of words, and duration.

set	school	turns	words	duration
training	Ohm	6 370	22 244	4.5 h
validation	Ohm	619	2 516	0.5 h
testing	Mont	6 653	23 641	3.9 h

Table 3.2: Keywords

category	German keywords	translation
positive valence	brav, fein, gut, schön	well-behaved/good, fine, good, nice
negative valence	böse, nein, nicht	bad, no, not
commands	aufstehen, bleiben, drehen, gehen, geradeaus, hinsetzen, kommen, laufen, links, rechts, setzen, stehen, stehenbleiben, stellen, stopp, tanzen, umdrehen, weiterlaufen	stand up, keep, turn, go, straight, sit down, come, run, left, right, sit, stand, stand still, put, stop, dance, turn around, keep running

In the test set, 85.6 % of the turns contain at least one keyword; 40.6 % of the turns contain two or more keywords. The average number of keywords contained in a turn is 1.4 and the average number of words per turn is 3.6.

System Parametrization and Training

Five different keyword spotting techniques will be evaluated in the following: the Tandem BLSTM-DBN approach introduced in this section [273], the CTC method as proposed in [75], the DBN outlined in Section 3.1.2 [278], a conventional phoneme-based HMM system, and a multi-stream HMM approach that incorporates BLSTM phoneme predictions as an additional discrete stream of observations (for further details on the implementation of the multi-stream approach, see Section 3.2.2). Using a set of 25 keywords, we will investigate the performance of the respective techniques focusing on the task of keyword detection in a child-robot interaction scenario.

The acoustic feature vectors used for all keyword detectors consisted of cepstral mean normalized MFCC coefficients 1 to 12, log. energy, as well as first and second order delta coefficients. The BLSTM network was trained on the *framewise* phoneme segmentations of the training set. Since the corpus is only transcribed at the word level, an HMM system was applied in order to obtain the phoneme-level forced alignments. The BLSTM input layer had a size of 39 (one for each feature) and the size of the output layer was 65 since a set of 54 German phonemes is modeled, with additional targets for *silence*, *short pause*, *breathing*, *coughing*, *laughing*, *unidentifiable phonemes*, *noise*, *human noise*, *nasal hesitation*, *vocal hesitation*, and *nasal+vocal hesitation*. Both hidden layers (for forward and backward processing) consisted of one backpropagation layer with 65 hidden cells and two LSTM layers with 130 and 65 memory blocks, respectively. Each memory block consisted of one memory cell. Input and output gates used hyperbolic tangent (tanh) activation functions, while the forget gates had logistic activation functions.

The BLSTM network was trained with standard backpropagation through time. Again, a learning rate of 10^{-5} was used. To improve generalization, zero mean Gaussian noise with standard deviation 0.6 was added to the inputs during training. Before training, all weights of the BLSTM network were randomly initialized in the

range from -0.1 to 0.1. Training was aborted as soon as no improvement on the validation set (two *Ohm*-sessions) could be observed for at least 50 epochs. Finally, the network that achieved the best framewise phoneme error rate on the validation set was chosen. The resulting frame error rate on the test set is 15.1 %. Note that for the BLSTM-DBN system, the validation set was exclusively used to determine a stop criterion for BLSTM training and not to tune parameters such as the number of memory blocks.

The DBN was trained applying the two-stage approach explained in Section 3.1.2: During the first training cycle of the DBN, models for phonemes and non-linguistic vocalizations were trained framewisely using the *Ohm*-sessions of the FAU Aibo Emotion Corpus. All Gaussian mixtures were split once the change of the overall log likelihood of the training set became less than 0.02 %. The number of mixtures per state was increased to eight. All models were composed of three hidden states.

In order to compare the performance of the Tandem model to a CTC keyword spotter based on whole-word modeling as proposed in [75], a BLSTM network with CTC output layer was trained. The output layer consisted of one output node per keyword and an additional output unit for the *non-keyword* event (see Section 2.3.10). As for the Tandem model, the BLSTM-CTC network consisted of one backpropagation layer and two LSTM layers for each input direction (size 65, 130, and 65, respectively). Network training was conducted exactly in the same way as for the Tandem approach. The only difference is that the CTC network uses keywords rather than phonemes as targets. Note that this leads to *empty* target sequences for training turns which contain no keywords.

As a baseline experiment, the performance of a phoneme-based keyword spotter using conventional HMM modeling was evaluated. Analogous to the DBN, each of the 54 phonemes was represented by three states (left-to-right HMMs) with eight Gaussian mixtures. Increasing the number of mixture components to more than eight did not result in better recognition accuracies. HMMs for non-linguistic events consisted of nine states. Cross-word triphone models were applied in order to model co-articulation. Details on the keyword detection technique used within the baseline HMM can be found in the experimental part of Section 3.1.2.

To investigate the performance gain when including the discrete BLSTM phone prediction b_t (see Equation 3.23) as an additional feature into the baseline HMM framework, the HMM-based system was extended to a multi-stream recognizer modeling MFCC and BLSTM observations in independent feature streams. As for the Tandem BLSTM-DBN approach, MFCC observations are modeled via Gaussian mixtures while the BLSTM feature is modeled using the discrete emission probability distribution $p(b_t|s_t)$. Thus, the BLSTM-HMM system can be interpreted as a combined continuous-discrete multi-stream HMM (also see Section 3.2.2).

Results

All five keyword spotting approaches were evaluated on children's speech as contained in the *Mont*-sessions of the FAU Aibo Emotion Corpus. Since only the *Ohm*-sessions are used during training, the experiments are completely speaker-independent. Figure 3.8 shows a part of the ROC curves for the baseline HMM, the multi-stream BLSTM-HMM, the DBN as introduced in Section 3.1.2, the CTC method proposed in [75], as well as for the Tandem BLSTM-DBN. Since the CTC framework offers no possibility to adjust the trade-off between a high true positive rate and a low false positive rate, we only get one operating point in the ROC space, corresponding to a true positive rate of 85.2 % at a false positive rate of 0.23 %. This operating point lies almost exactly on the ROC curve of the Tandem BLSTM-DBN so that both techniques can be characterized as equally suited for detecting keywords in the given child-robot interaction scenario. Note, however, that unlike the CTC method, the Tandem approach is more flexible as far as changes in the keyword vocabulary are concerned: As both, the BLSTM network and the DBN are phoneme-based, the Tandem model is vocabulary independent. By contrast, the CTC network is trained on whole words, which implies that the whole network would have to be re-trained if a vocabulary entry is to be changed. If a higher false positive rate can be tolerated, the Tandem approach achieves a keyword detection rate of up to 95.9 %. As can be seen in Figure 3.8, the Tandem model prevails over the baseline HMM system. The performance difference is most significant at lower false positive rates: When evaluating the ROC curve at a false positive rate of 0.4 %, the absolute difference in true positive rates is larger than 12 %. This indicates that for our children's speech scenario, modeling context via Long Short-Term Memory leads to better results than conventional triphone modeling. In general, for the investigated children's speech scenario, considering contextual information during decoding seems to be essential, since the DBN approach which models only monophones leads to a lower ROC performance when compared to the triphone HMM system and to systems applying LSTM. At lower false positive rates, modeling the co-articulation properties of children's speech by applying the principle of Long Short-Term Memory also boosts the performance of the HMM approach which can be seen in the ROC curve for the multi-stream BLSTM-HMM. Yet, the overall performance is slightly better for the Tandem system.

Figures 3.9(a) to 3.9(d) show the performance of the five different keyword detection approaches when tested on different fractions of the FAU Aibo Emotion Corpus. Figure 3.9(a) considers exclusively the 17 female speakers of the *Mont* school while Figure 3.9(b) shows the word spotting performance for the eight male speakers. For female speakers we can observe a significantly larger performance gap between the multi-stream BLSTM-HMM technique and the Tandem BLSTM-DBN than when considering male speakers, for which both BLSTM-based methods perform almost equally well. Generally, the Tandem approach as proposed in this section prevails

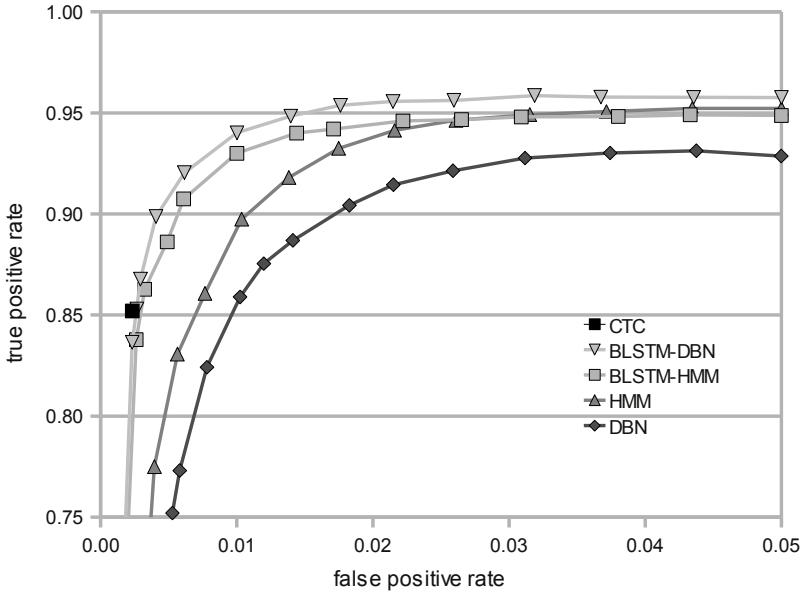
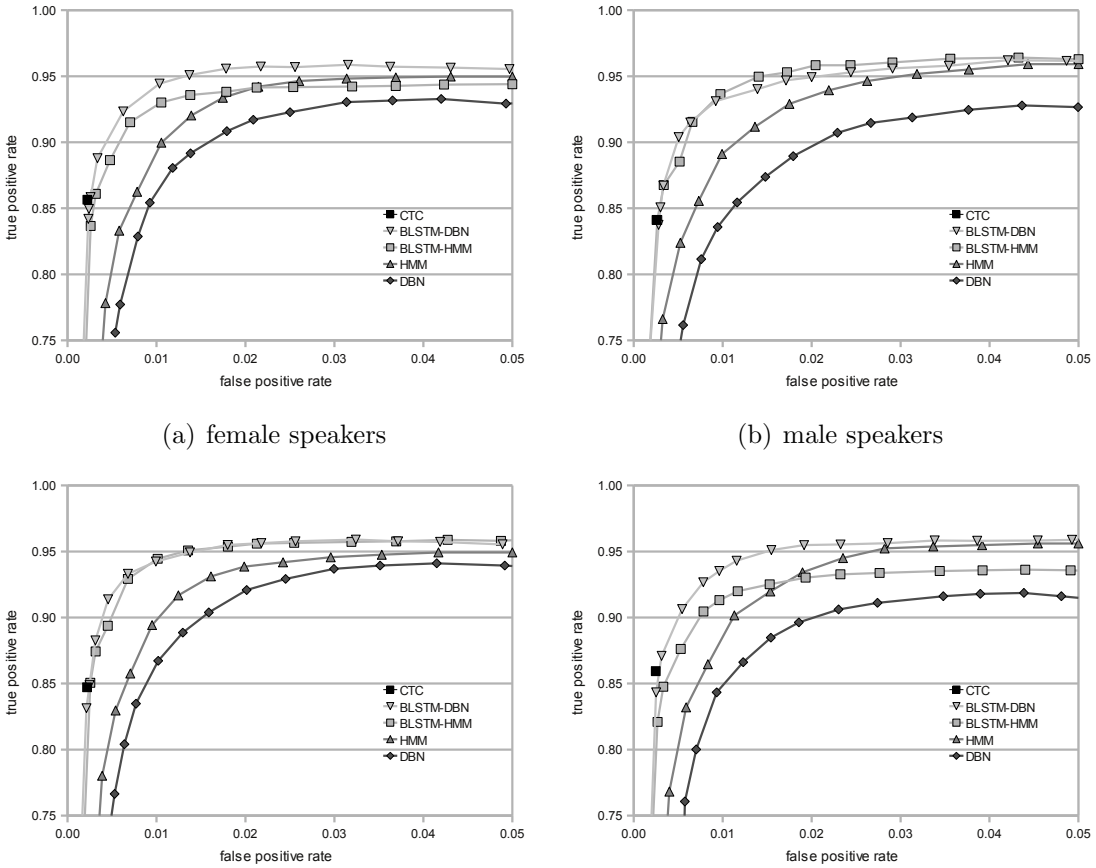


Figure 3.8: Evaluation on the FAU Aibo Emotion Corpus (25 keywords): part of the ROC curve for the baseline HMM system, the multi-stream BLSTM-HMM, the DBN keyword spotter (without BLSTM phoneme predictions), the CTC approach, and the Tandem BLSTM-DBN technique. The operating points correspond to $a = 0, 1, 2, 3$, etc. (linear interpolation).

over the baseline HMM system for both, female and male speakers – especially at lower false positive rates. Figures 3.9(c) and 3.9(d) contain the results for younger (age between 10 and 11 years) and older children (age between 12 and 13 years), respectively. The baseline HMM leads to almost equal performance for both, younger and older children, however, the multi-stream HMM performs significantly better for the younger age group. Again, the Tandem BLSTM-DBN consistently leads to better results when compared to the HMM system, indicating that the Tandem system is suitable for both genders and different age groups. Generally we can observe that the performance of techniques such as the DBN system, the (multi-stream) HMM approach, and the CTC method shows a higher dependency on the childrens' age and gender than the proposed Tandem BLSTM-DBN.

Table 3.3 shows the average true positive rates for individual keywords at a false positive rate of 1 %. Keywords are grouped into words expressing positive valence, words expressing negative valence, and command words, according to Table 3.2. For all keyword spotting systems, we observe the same trend: Command words seem to be easier to detect than words related to valence. Besides differences in phonetic composition and lengths of keywords, a plausible reason for this phenomenon is



(a) female speakers

(b) male speakers

(c) younger children (age 10-11)

(d) older children (age 12-13)

Figure 3.9: ROC curves for the different keyword spotting systems evaluated on female speakers, male speakers, younger children (age between 10 and 11 years), and older children (age between 12 and 13 years).

that pronunciations of 'positive' or 'negative' words tend to be emotionally colored while command words are rather pronounced in a neutral or emphatic way. Furthermore, for most recognition engines, words expressing negative valence lead to higher error rates than words associated with positive valence. Since the FAU Aibo Emotion Corpus contains emotion annotations at the word-level, it is possible to analyze which emotions are typically assigned to which keyword. Table 3.4 shows the emotion class distributions for each word category: A considerable percentage of 'positive' and 'negative' keywords are pronounced in a *motherese* (positive valence) and *angry* (negative valence) way, respectively, whereas most of the command words are annotated as *neutral* or *emphatic*. Similar results were observed by [210], where emotional children's speech led to higher error rates.

Table 3.3: True positive rates (tpr) for the DBN, HMM, BLSTM-HMM, and BLSTM-DBN keyword spotter at a false positive rate of 0.01: mean and standard deviation (std.) of the true positive rates for individual keywords expressing positive/negative valence or command words; weighted (WAv) and unweighted average (UAv) true positive rate for the complete set of keywords; 'unweighted' refers to the true positive rate averaged over all keywords while 'weighted' means the average of the true positive rates weighted by the number of occurrences of the individual keywords.

	DBN		**HMM**		**BLSTM-HMM**		**BLSTM-DBN**	
tpr	mean	std.	mean	std.	mean	std.	mean	std.
pos. valence	0.716	0.281	0.724	0.223	0.595	0.317	0.741	0.280
neg. valence	0.535	0.264	0.576	0.272	0.702	0.244	0.662	0.213
commands	0.817	0.182	0.858	0.118	0.929	0.051	0.926	0.070
UAv	0.767	0.222	0.803	0.178	0.848	0.194	0.865	0.166
WAv	0.859		0.897		0.930		0.940	

Table 3.4: Emotions assigned to the keyword categories in %: angry, motherese, emphatic, and neutral.

[%]	**angry**	**motherese**	**emphatic**	**neutral**
positive valence	0	23	0	77
negative valence	15	0	16	69
commands	4	1	9	86
all	4	2	9	85

3.1.4 Hybrid CTC-DBN

A further technique applying BLSTM for phoneme-based keyword detection was introduced in [280]. In contrast to the discriminative approach (Section 3.1.1) and the Tandem method (Section 3.1.3), this technique includes a CTC output layer and thus can be trained on unsegmented data. Similar to the Tandem model, the hybrid CTC-DBN approach makes use of a DBN layer to decode the phoneme string detected by the CTC network. The DBN is trained to explicitly learn and model typical phoneme confusions, deletions, and insertions that occur in the CTC layer which allows the network to detect keywords even if the pronunciation differs from the keyword phonemizations in the dictionary. A major difference between the hybrid CTC-DBN outlined in this section and the Tandem model described in Section 3.1.3 is that the DBN used in the hybrid CTC-DBN model exclusively decodes the *phonemewise* CTC predictions and not the *framewise* BLSTM output in combination with the MFCC features. Further, there is an important difference between the hybrid CTC-DBN and the CTC keyword spotter proposed in [75], since the CTC-DBN is trained on phonemes rather than on whole keywords. This implies

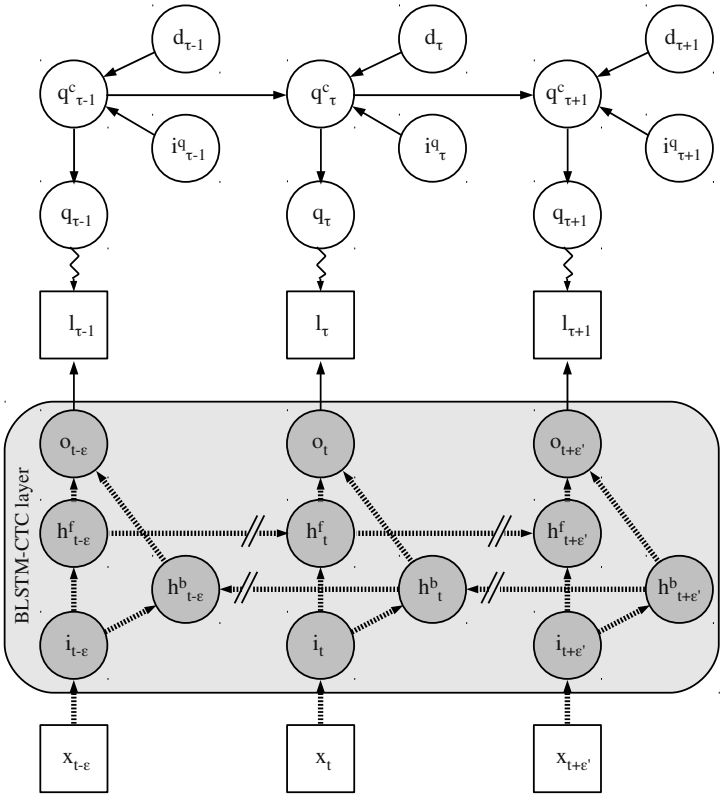


Figure 3.10: Hybrid CTC-DBN architecture for training.

that, similar to the other approaches outlined in Sections 3.1.1 to 3.1.3, the hybrid CTC-DBN can be applied for vocabulary independent keyword detection and the keyword inventory does not have to be considered during system training.

Again, the keyword spotting system consists of two major components: a bidirectional Long Short-Term Memory recurrent neural net and a Dynamic Bayesian Network. The BLSTM network can access long-range context information along both input directions and uses a Connectionist Temporal Classification output layer [90] to localize and classify the phonemes, while the DBN is applied for keyword detection.

Training

Figure 3.10 shows the DBN model architecture that is used for training the hybrid CTC-DBN keyword detector. The grey-shaded box represents the BLSTM-CTC layer comprising an input layer i_t, two hidden layers h_t^f and h_t^b (forward and back-

ward direction), and an output layer o_t. Note that even though the BLSTM network produces an output activation for every feature frame index, only the non-blank labels are forwarded to the DBN. To simplify the notation in this section, we use a counter variable τ which is synchronized with the CTC label predictions rather than with the feature frame index. Time index t is synchronized with the feature frames. In order to indicate that not every feature frame x_t triggers a CTC label prediction l_τ, Figure 3.10 uses the variable ϵ denoting the number of feature frames that lie between the CTC outputs $l_{\tau-1}$ and l_τ. Similarly, ϵ' represents the number of frames between l_τ and $l_{\tau+1}$. Within the DBN layer the following random variables are defined for every τ: q_τ is the current phoneme index corresponding to the phoneme annotation of the training sequence, q_τ^c is a simple count variable containing the current position within the ground truth phoneme string, and the binary variables d_τ and i_τ^q indicate deletions and insertions, respectively. With L being the length of the CTC output phoneme sequence, the DBN structure in Figure 3.10 corresponds to the factorization

$$p(l_{1:L}, q_{1:L}, q_{1:L}^c, d_{1:L}, i_{1:L}^q) = \prod_{\tau=1}^{L} p(l_\tau|q_\tau) f(q_\tau|q_\tau^c) p(d_\tau) p(i_\tau^q) f(q_1^c|d_1, i_1^q) \prod_{\tau=2}^{L} f(q_\tau^c|q_{\tau-1}^c, d_\tau, i_\tau^q). \tag{3.25}$$

The probability of the observed label sequence $l_{1:L}$ can then be computed by summing over all hidden variables. The CPF $f(q_\tau^c|q_{\tau-1}^c, d_\tau, i_\tau^q)$ defines that the count variable q_τ^c is incremented by one at every step τ in case d_τ and i_τ are equal to zero. Otherwise, if there is a deletion ($d_\tau = 1$), q_τ^c is incremented by two, whereas an insertion implies that $q_\tau^c = q_{\tau-1}^c$. Thus, apart from training the CTC network, the goal of the training phase is to learn the CPFs $p(l_\tau|q_\tau)$, $p(d_\tau)$, and $p(i_\tau^q)$ (i. e., to learn substitution, deletion, and insertion probabilities).

Decoding

Figure 3.11 shows the DBN decoding architecture for keyword spotting based on hybrid CTC-DBN modeling. Recall that dotted lines within the DBN layer represent so-called *switching parent* dependencies which allow a variable's parents (and CPFs) to change conditioned on the current value of the switching parent. The DBN for decoding contains five additional hidden variables: w_τ denotes the identity of the current word, w_τ^{ps} is the position within the word, w_τ^{tr} indicates a word transition, c_τ represents a 'cut' variable that is equal to one as soon as there is a deletion at the *end* of a keyword, and a hidden *garbage variable* g_τ indicates whether the current word is a keyword or not. According to Figure 3.11, we get the following factorization:

$$
\begin{aligned}
&p(l_{1:L}, q_{1:L}, w_{1:L}, w^{ps}_{1:L}, w^{tr}_{1:L}, d_{1:L}, i^q_{1:L}, c_{1:L}, g_{1:L}) = \\
&\quad p(w_1) f(w^{ps}_1 | d_1, i^q_1) p(q_1 | w_1, w^{ps}_1, g_1, c_1, i^q_1) \\
&\quad \times \prod_{\tau=1}^{L} p(l_\tau | q_\tau) f(w^{tr}_\tau | w_\tau, w^{ps}_\tau) f(c_\tau | w_\tau, w^{ps}_\tau) f(g_\tau | w_\tau) p(d_\tau | g_\tau) p(i^q_\tau | g_\tau) \\
&\quad \times \prod_{\tau=2}^{L} p(w_\tau | w_{\tau-1}, w^{tr}_{\tau-1}) f(w^{ps}_\tau | w^{ps}_{\tau-1}, w^{tr}_{\tau-1}, d_\tau, i^q_\tau) p(q_\tau | q_{\tau-1}, w_\tau, w^{ps}_\tau, g_\tau, c_\tau, i^q_\tau).
\end{aligned}
\tag{3.26}
$$

As in Section 3.1.2, the hidden variable w_τ can take values between 0 and K, with K being the number of different keywords. The CPFs for g_τ, w^{tr}_τ, w_τ are the same as for the DBN outlined in Section 3.1.2. A word transition occurs whenever $w^{ps} = P$, if P is the number of phonemes contained in w_τ. If a keyword is detected, q_τ is known, given w_τ and w^{ps}_τ. Otherwise, for garbage speech, a phoneme bigram defines $p(q_\tau | q_{\tau-1})$. The same holds for the case when an insertion occurs while a keyword is decoded ($i^q_\tau = 1$), or when the last phoneme of a keyword is deleted ($d_\tau = 1$ and $c_\tau = 1$). Similar to the variable q^c_τ in the DBN for training, the increment of w^{ps}_τ is controlled by the insertion and the deletion variable. The 'cut' variable c_τ is equal to one if w^{ps}_τ exceeds P, meaning that the last phoneme of a keyword has been deleted.

Experiments and Results

The hybrid CTC-DBN keyword spotter was trained and evaluated on the TIMIT corpus. As in Section 3.1.3, 200 utterances of the TIMIT training split were used as validation set for determining when to abort training, and the remaining utterances as training set. The size of the CTC output layer was 40, representing 39 phonemes plus one blank label. The network consisted of three hidden layers per input direction: a backpropagation layer composed of 78 hidden cells and two hidden LSTM layers containing 128 and 80 memory blocks respectively. Each memory block consisted of one cell. A learning rate of 10^{-4} was used and the keyword spotting task was the same as in Section 3.1.2. Again, the performance of the baseline HMM is given for comparison.

Moreover, the benefit of the DBN decoder in comparison to a trivial phoneme string search on the raw CTC output was evaluated. Figure 3.12 shows a part of the ROC curve for the CTC-DBN keyword spotter, the HMM based keyword spotter, as well as for a simple string matching approach, tolerating a Levenshtein distance of 1 and 2, respectively. It can be seen that the hybrid CTC-DBN decoder not only prevails over CTC string matching but also outperforms the HMM approach by up to 7 % (at a false positive rate of 0.4 %). For higher a priori keyword likelihoods the performance gap becomes smaller as more phoneme confusions are tolerated during the keyword search.

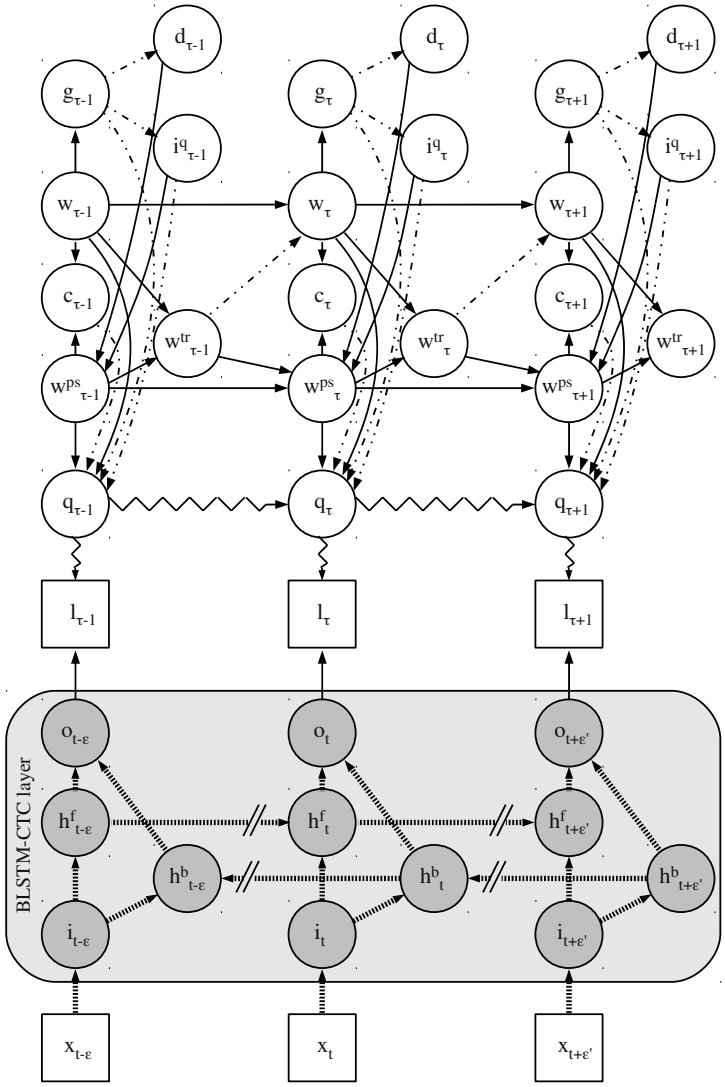


Figure 3.11: Hybrid CTC-DBN architecture for keyword spotting.

3.1.5 Tandem CTC-DBN

In this section, a novel Tandem CTC-DBN keyword spotting technique that combines the advantages of the methods proposed in [280] (see Section 3.1.4) and [273] (see Section 3.1.3) is introduced. I. e., the model can be trained on unsegmented data and – unlike the hybrid approach in [280] – applies *framewise* modeling within the DBN layer.

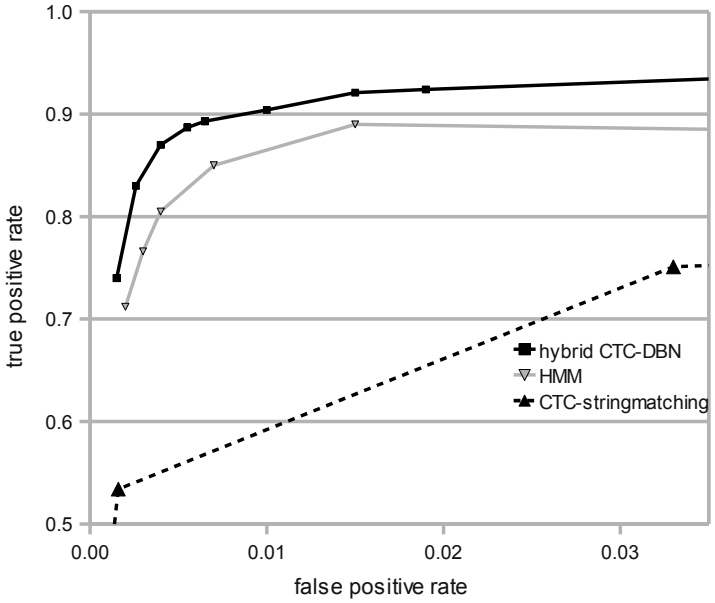


Figure 3.12: Part of the ROC curve using the hybrid CTC-DBN, the baseline HMM, and a simple string search on the CTC phoneme output. Evaluation on the TIMIT database.

Decoding

The corresponding CTC-DBN architecture of the decoder can be seen in Figure 3.13. Again, the model consists of multiple hierarchy levels (word, phoneme, state, observation, and BLSTM-CTC layer). Unlike the BLSTM network used in Section 3.1.3, the BLSTM-CTC network does not generate phoneme estimates for every frame but rather outputs *spikes* as explained in Section 2.3.10. Thus, the observed variable z_t contains the phoneme prediction of the BLSTM and l_t^{sp} indicates whether a spike corresponding to a phoneme prediction is produced by the BLSTM in the current time step. The DBN therefore interprets z_t and l_t^{sp} as observed 'higher level features'. Analogous to Equation 3.23, z_t is the index of the most likely phoneme:

$$z_t = \underset{j}{\operatorname{argmax}}(o_t^1, ..., o_t^j, ..., o_t^P, o_t^{blank}). \tag{3.27}$$

If the BLSTM output layer displays a blank label, the binary variable l_t^{sp} is equal to zero, otherwise a phoneme is detected and l_t^{sp} equals one (see Figure 3.14 for an example).

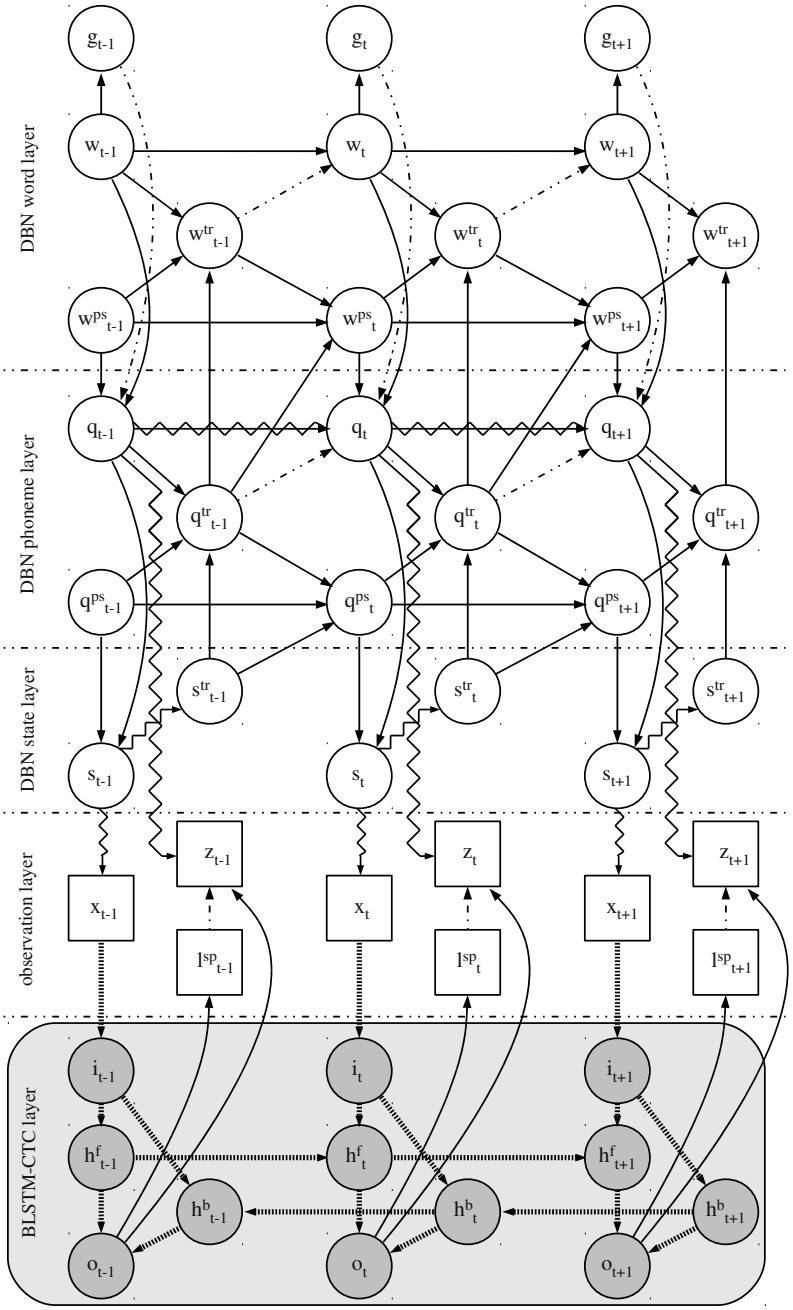


Figure 3.13: Structure of the Tandem CTC-DBN keyword spotter.

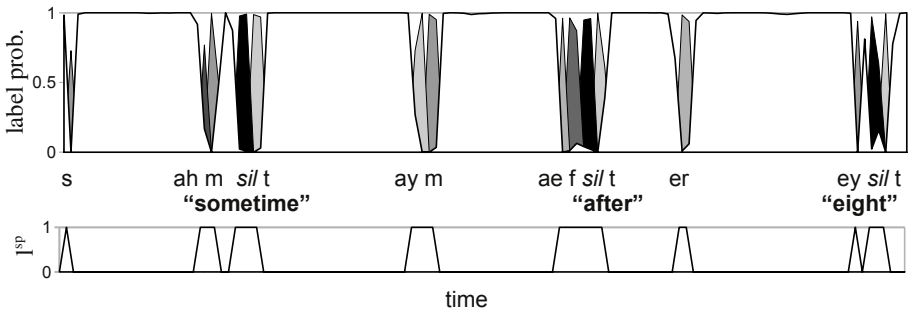


Figure 3.14: top: CTC output activations corresponding to probabilities of observing a phoneme (grey shaded spikes) or the *blank* label (white areas); different shadings represent different phoneme identities; bottom: values of the spike indicator variable l_t^{sp}.

The DBN structure depicted in Figure 3.13 corresponds to the factorization

$$
\begin{aligned}
&p(g_{1:T}, w_{1:T}, w_{1:T}^{tr}, w_{1:T}^{ps}, q_{1:T}, q_{1:T}^{tr}, q_{1:T}^{ps}, s_{1:T}^{tr}, s_{1:T}, x_{1:T}, z_{1:T}, l_{1:T}^{sp}) = \\
&\quad f(q_1^{ps}) p(q_1 | w_1^{ps}, w_1, g_1) f(w_1^{ps}) p(w_1) \\
&\quad \times \prod_{t=1}^{T} p(x_t|s_t) p(z_t|q_t, l_t^{sp}) f(s_t|q_t^{ps}, q_t) f(q_t^{tr}|q_t^{ps}, q_t, s_t^{tr}) p(s_t^{tr}|s_t) f(g_t|w_t) \\
&\quad \times f(w_t^{tr}|q_t^{tr}, w_t^{ps}, w_t) \\
&\quad \times \prod_{t=2}^{T} f(q_t^{ps}|s_{t-1}^{tr}, q_{t-1}^{ps}, q_{t-1}^{tr}) p(w_t|w_{t-1}^{tr}, w_{t-1}) p(q_t|q_{t-1}^{tr}, q_{t-1}, w_t^{ps}, w_t, g_t) \\
&\quad \times f(w_t^{ps}|q_{t-1}^{tr}, w_{t-1}^{ps}, w_{t-1}^{tr}).
\end{aligned}
\tag{3.28}
$$

As can be seen in Figure 3.13, the CTC spike indicator variable l_t^{sp} serves as a switching parent of the CTC observation z_t: If a spike (indicating the detection of a phoneme) is observed, l_t^{sp} triggers a statistical dependency $p(z_t|q_t, l_t^{sp} = 1)$ that tells the DBN to use the CTC output during decoding. Otherwise, if $l_t^{sp} = 0$, the DBN ignores the CTC observation z_t. Since the conditional dependency $p(z_t|q_t, l_t^{sp} = 1)$ is learned during training, the DBN also *learns* typical CTC phoneme confusions which introduces a certain robustness with respect to errors in the BLSTM-CTC prediction and implicitly controls whether, for a given phoneme, x_t or z_t has more influence on inference. All other statistical relations between random variables in the word, phoneme, and state layer of the CTC-DBN depicted in Figure 3.13 are identical to the CPFs outlined in Section 3.1.2.

The Graphical Model applied for learning the random CPFs $p(x_t|s_t)$, $p(s_t^{tr}|s_t)$, and $p(z_t|q_t, l_t^{sp} = 1)$ can be derived similarly to the GM in Section 3.1.2 by omitting

the word layer and introducing a count variable q^c. Again, the training procedure is split up into two stages: In the first stage phonemes are trained framewise, while during the second stage, a forced alignment is used. For a training sequence of length T, we get the factorization

$$\begin{aligned} p(q^c_{1:T}, q_{1:T}, q^{tr}_{1:T}, q^{ps}_{1:T}, s^{tr}_{1:T}, s_{1:T}, x_{1:T}, z_{1:T}, l^{sp}_{1:T}) = \\ f(q^{ps}_1) f(q^c_1) \prod_{t=1}^{T} p(x_t|s_t) p(z_t|q_t, l^{sp}_t) f(s_t|q^{ps}_t, q_t) p(s^{tr}_t|s_t) f(q^{tr}_t|q^{ps}_t, q_t, s^{tr}_t) f(q_t|q^c_t) \\ \times \prod_{t=2}^{T} f(q^{ps}_t|s^{tr}_{t-1}, q^{ps}_{t-1}, q^{tr}_{t-1}) f(q^c_t|q^c_{t-1}). \end{aligned} \tag{3.29}$$

In what follows, a detailed evaluation of the Tandem CTC-DBN keyword detector is shown.

3.1.6 Evaluation and Discussion

To get an impression of the keyword spotting accuracies obtained when applying the techniques outlined in Sections 3.1.1 to 3.1.5, all methods were evaluated on two different keyword detection tasks, aiming to consider a variety of different speaking styles and using the same training and test conditions for all systems. Both tasks focus on *vocabulary independent* keyword spotting, i. e., the keyword vocabulary is not known during the training phase of the models.

Databases

The first task was to detect a set of 60 randomly selected keywords in the TIMIT test set. The TIMIT corpus consists of read speech and features speaker-independent test and training sets. Its total vocabulary size is 4.9 k. As training set for the TIMIT experiment all 3 696 utterances contained in the official TIMIT training partition were used. In conformance with [278], [273], and [280], only those utterances that contain at least one keyword were considered as test set (321 out of 1 680 TIMIT test utterances). The average length of a TIMIT training utterance is 3.0 s and the average length of a test utterance is 3.2 s. 21 out of the 60 randomly chosen keywords did not occur in the training partition. Note that the keyword inventory was randomly chosen, *independent* of whether the keywords occur in the training database or not. In total, there are 305 keyword occurrences in the TIMIT training set and 354 in the test set.

As a considerably more challenging scenario, all keyword detection techniques were also trained and evaluated on the freely available SEMAINE database [155]

(www.semaine-db.eu), which contains emotionally colored, spontaneous, and conversational speech recorded during interactions between a user and a Wizard-of-Oz conversational agent (or *operator*). The audiovisual SEMAINE corpus was originally recorded to study natural social signals that occur in conversations between humans and artificially intelligent agents. It has been used as training material for the development of the SEMAINE system [206] (see Section 2.1). During the creation of the database, the Sensitive Artificial Listener scenario as explained in Section 2.1.1 was used. It involves a user interacting with emotionally stereotyped characters whose responses are stock phrases keyed to the user's emotional state rather than the content of what he/she says. For the recordings, the participants are asked to talk in turn to the four SAL characters introduced in Section 2.1.1. The data used in the following experiments is based on the 'Solid-SAL' part of the SEMAINE database, i. e., the users do not speak with artificial agents but instead with human operators who pretend to be the agents (Wizard-of-Oz setting). Further details on the interaction scenario can be found in [155] and [221]. Because we assume that the SAL agent has no language understanding, a few rules necessarily govern this type of interaction. The most important of these is that the agent (operator) cannot answer questions. However, the operators are instructed that the most important aspect of their task is to create a conversation that has a natural style; strict adherence to the rules of a SAL engagement was secondary to this so that the interactions would produce a rich set of conversation-related behaviors.

As for the TIMIT experiment, evaluations on the SEMAINE corpus are based on audio data with 16 kHz and 16 bits per sample. Recordings for the SEMAINE database were captured with a close-talking microphone (AKG HC 577 L). The task was to detect 40 different randomly selected keywords. The speaker-independent SEMAINE test set consists of recording sessions 3, 5, 12, 14, and 20 (640 utterances from the user) and the remaining 14 recording sessions were used for training (4898 utterances from both, user and operator). Note that for this task, all 640 test utterances are considered, i. e., also utterances that contain no keywords. In the SEMAINE training partition, the average length of an utterance is 4.0 s and the average length of a test utterance is 3.9 s. In total, 20 different speakers are contained in the SEMAINE database. Training and test splits of the SEMAINE task are speaker-independent, meaning that the five speakers in the test sessions do not occur in the training partition. In contrast to the TIMIT keyword spotting task, all of the randomly chosen keywords for the SEMAINE task occur in the training database. The total number of keyword occurrences in the SEMAINE training partition is 2 669 and the number of keyword occurrences in the SEMAINE test set is 394. Due to the challenging speech characteristics in the SEMAINE corpus (disfluent, spontaneous, emotional speech spoken in different accents) the word error rate (WER) obtained with a standard ASR system is extremely high, e. g., with a conventional tied-state cross-word triphone HMM system trained on the SEMAINE training data, the WER is as high as 64 %. The total vocabulary size of the SEMAINE corpus is 3.6 k.

To learn the weight vector ω for the discriminative keyword spotting approach (see Sections 3.1.1), two times 200 disjunct keywords were randomly selected – one time for learning ω and one time for validating the current weight vector ω. Note that for each of the two times 200 keywords one utterance containing, and one utterance not containing the respective keyword were selected (as described in Section 3.1.1). Those utterances were randomly chosen from the training set of the TIMIT corpus and the SEMAINE database, respectively.

All keywords consisted of at least three phonemes. Keyword pronunciations were obtained from the CMU dictionary and alternative pronunciation variants as contained in the CMU dictionary were also included. By focusing on small keyword vocabularies we consider conditions which are typical for human-machine communication applications (for example conversational agent scenarios such as the SEMAINE system [206]) and many command detection tasks. Still, all investigated approaches do not use whole-word modeling and thus are applicable for larger vocabularies as well.

Parametrization

The feature vectors x_t applied for all tasks and keyword spotting systems consisted of cepstral mean normalized MFCC coefficients 1 to 12, log. energy, as well as first and second order delta coefficients. As phoneme inventory $\mathcal{P}$ the CMU set of 39 phonemes together with *short pause* and *silence* was used. For the SEMAINE task, additional models trained on the non-linguistic vocalizations *breathing*, *laughing*, and *sighing* were included.

As a baseline experiment, the performance of a phoneme-based keyword spotter using conventional HMM modeling in combination with a phoneme bigram was evaluated. Each phoneme was represented by three states (left-to-right HMMs) with 16 Gaussian mixtures. HMMs for non-linguistic events consisted of nine states. Cross-word triphone models were applied to model context in the HMM system.

As in Section 3.1.1, $\lambda_h = 1$ and $\lambda_o = 1.5$ was chosen for the discriminative approaches (see Equation 3.11). According to past experience [280], the BLSTM networks were configured to have three hidden layers: one backpropagation layer of size 78 and two LSTM layers consisting of 128, and 80 memory blocks, respectively. Each memory block consisted of one memory cell. For BLSTM training a learning rate of 10^{-5} was used while for CTC training a learning rate of 10^{-4} was chosen. Training was aborted as soon as no improvement on a validation set (200 randomly selected utterances from the training sets of the respective tasks) could be observed for at least 50 epochs, and the network that achieved the best phoneme error rate on the validation set was selected. The DBNs were trained applying the two-stage technique outlined in Section 3.1.2. All DBN phoneme models consisted of three states with 16 Gaussian mixtures.

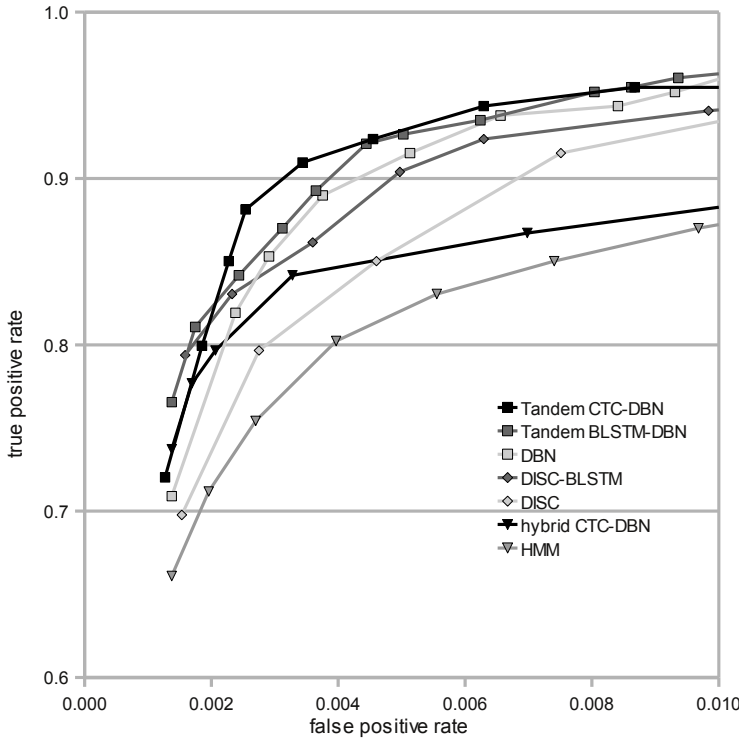


Figure 3.15: TIMIT keyword spotting task: part of the ROC curve for different keyword detection approaches.

Results

Figure 3.15 shows a part of the ROC curve for the TIMIT experiment, i. e., the true positive rate (tpr) is shown as a function of the false positive rate (fpr). Focusing on rather low fpr values seems reasonable when considering for example conversational agents as target application: Missing a few keywords is less critical than wrongly detecting a large number of keywords that are not uttered by the user and will cause false decisions by the dialogue management processing the keyword spotter output.

For false positive rates between 0.2 and 0.4 %, the best vocabulary independent approach is the Tandem CTC-DBN proposed in Section 3.1.5. For higher and lower fpr-values the BLSTM-DBN (Section 3.1.3) achieves comparable true positive rates. Both BLSTM-based Tandem techniques are able to outperform the DBN-method (see Section 3.1.2) which indicates that long-range context modeling leads to improved keyword spotting performance. The hybrid CTC-DBN method (Section 3.1.4) prevails over the baseline HMM but cannot compete with the Tandem models. For the two discriminative approaches ('DISC' and 'DISC-BLSTM'), the benefit of integrating BLSTM phoneme predictions is also clearly visible which confirms pre-

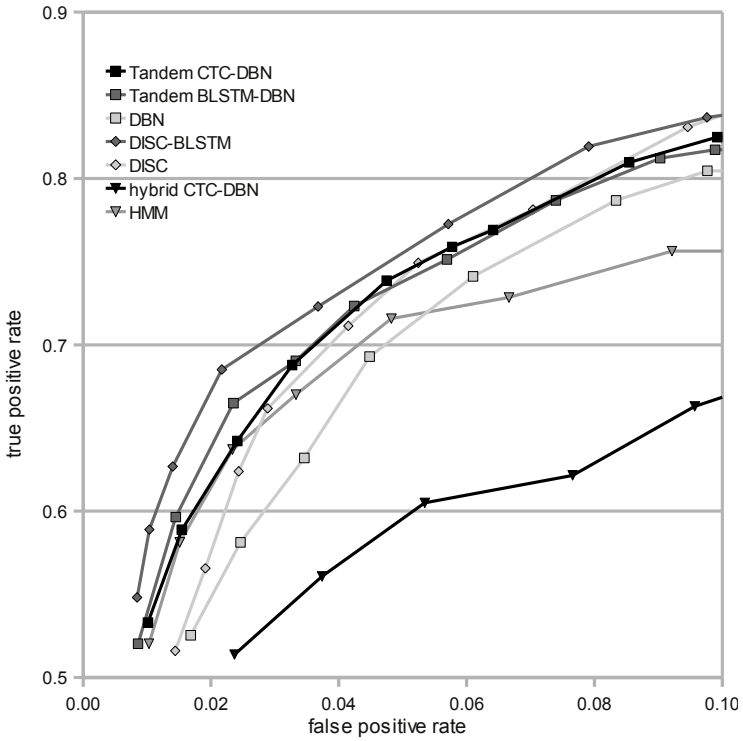


Figure 3.16: SEMAINE keyword spotting task: part of the ROC curve for different keyword detection approaches.

vious studies on discriminative keyword spotting [275]. The AUC obtained for the discriminative technique is 0.99, corresponding to results reported for the TIMIT task in [123].

The results for the SEMAINE keyword spotting task can be seen in Figure 3.16. For such challenging tasks involving highly spontaneous and emotionally colored, disfluent speech, context modeling seems to be even more important: For lower false positive rates, the approaches applying only monophone modeling (i. e., the discriminative keyword spotter and the DBN) lead to lower true positive rates than the triphone HMM-baseline. The AUC for the 'DISC'-approach is 0.95. The hybrid CTC-DBN exclusively relying on the CTC predictions is obviously not suited for spontaneous speech while the Tandem BLSTM-DBN and CTC-DBN perform comparably well and prevail over the HMM approach. Best performance on the SEMAINE task can be obtained with the discriminative BLSTM keyword spotter outlined in Section 3.1.1 [275].

In order to compare keyword detection accuracies for a defined *range* of acceptable false positive rates, let us introduce a 'local AUC' ($lAUC$) which corresponds to

Table 3.5: Local AUC ($lAUC$) obtained for the TIMIT and the SEMAINE task when using different keyword detection approaches.

	TIMIT	**SEMAINE**
model architecture	$lAUC^{0.01}_{0.001}$	$lAUC^{0.1}_{0.01}$
Tandem CTC-DBN	**0.9089**	0.7317
Tandem BLSTM-DBN	0.9060	0.7341
DBN	0.8889	0.6961
DISC-BLSTM	0.8886	**0.7558**
DISC	0.8477	0.7239
hybrid CTC-DBN	0.8461	0.5799
HMM	0.8036	0.7020

the AUC between a lower fpr boundary ε_l and an upper fpr boundary ε_u, normalized by the maximum AUC (ε_u-ε_l) in that range. In other words, we define

$$lAUC^{\varepsilon_u}_{\varepsilon_l} = \frac{AUC^{\varepsilon_u}_{\varepsilon_l}}{\varepsilon_u - \varepsilon_l} \tag{3.30}$$

which is equal to 1 for perfect keyword detection. For the TIMIT task we analyze $lAUC^{0.01}_{0.001}$, while for the more challenging SEMAINE task, we allow higher false positive rates up to 0.1. Table 3.5 shows the $lAUC$-values obtained for the different model architectures, confirming the results illustrated in Figures 3.15 and 3.16.

A large absolute performance difference between the TIMIT and the SEMAINE task is observed for all approaches and is comprehensible given the different speaking styles contained in the two corpora. Even though certain relative differences can be seen (e. g., purely discriminative modeling ('DISC-BLSTM') is best suited for emotional speech while combined discriminative-generative modeling (CTC-DBN) prevails for read speech), we can consistently see an improvement via BLSTM phoneme predictions – independent of speech characteristics and recognition frameworks.

The aim of this section was to investigate how long-range context modeling via Long Short-Term Memory recurrent neural networks can improve the performance of vocabulary independent keyword detection and to provide an overview over advanced discriminative and generative keyword spotting techniques that exclusively rely on acoustic evidence and do not require an in-domain language model [123, 273, 275, 278, 280, 297]. In order to combine the advantages of framewise Tandem BLSTM-DBN modeling and Connectionist Temporal Classification, a novel Tandem CTC-DBN keyword spotter that exploits the principle of Long Short-Term Memory and does not presume presegmented data for training was introduced in Section 3.1.5. The experiments in this section aimed to evaluate the keyword spotting accuracies of seven different approaches on the TIMIT and the SEMAINE database and demonstrated that the best vocabulary independent keyword spotting performance on read speech can be obtained with the proposed Tandem CTC-DBN

approach. For spontaneous speech, purely discriminative modeling in combination with BLSTM prevails over all other investigated methods.

3.2 Conversational Speech Recognition

The accuracy of systems for automatic speech recognition heavily depends on the quality of the features extracted from the speech signal. Thus, during the last decades, a variety of methods were proposed to enhance commonly used MFCC or PLP features, especially in noisy conditions. As indicated in Section 3.1, a popular technique that has become state-of-the-art in modern ASR systems, is to apply a neural network to generate phoneme or phoneme state posteriors which in turn can be used as 'Tandem' features [107].

While first experiments on Tandem ASR systems concentrated on using the logarithmized and decorrelated activations of the *output* layer of recurrent neural networks or multi-layer perceptrons as probabilistic features, recent studies report performance gains when extracting the activations of a narrow *hidden* layer within the network as so-called 'bottleneck' features [95]. This implies the advantage that the size of the feature space can be chosen by defining the size of the network's bottleneck layer. Consequently, the dimension of the feature vectors is independent of the number of network training targets. The linear outputs of the bottleneck layer are usually well decorrelated and do not have to be logarithmized.

Since human speech is highly context-sensitive, both, the ASR front- and back-end need to account for contextual information in order to produce acceptable recognition results. Standard recognizer back-ends consider context by applying triphones, using language models, and via the Markov assumption in Hidden Markov Models or general Graphical Models. Feature-level context is usually modeled by appending derivatives of low-level features and by presenting a number of successive stacked feature frames to the neural network for Tandem feature extraction. Furthermore, the extraction of long-term features is an active area of research [245]. In Tandem systems, context can also be modeled *within* the neural network, e. g., by using recurrent connections. Motivated by the promising results obtained for vocabulary independent keyword spotting exploiting LSTM (see Section 3.1.6), we now focus on continuous recognition of conversational speech and investigate how traditional ASR systems can be improved via phoneme estimates or features produced by LSTM and BLSTM networks. First, in Section 3.2.1, a Tandem BLSTM-HMM system [279] modeling BLSTM phoneme estimates as additional feature is proposed. Section 3.2.2 introduces a multi-stream HMM architecture [281] in which both, continuous MFCC and discrete BLSTM features are decoded as independent data streams. Next, in Section 3.2.3, we examine a BLSTM front-end [291] integrating continuous, logarithmized, and decorrelated BLSTM features into an ASR system. Finally, in Section 3.2.4, a novel Bottleneck-BLSTM feature extractor [292, 296] uniting the principles

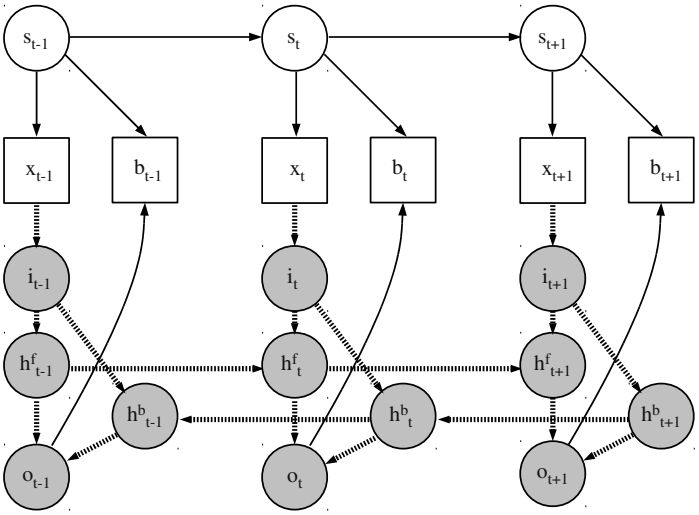


Figure 3.17: Architecture of the Tandem BLSTM-HMM decoder.

of bottleneck features, LSTM, and bidirectional speech modeling is outlined. All of the proposed context-sensitive ASR systems are compared and evaluated on spontaneous conversational speech in Section 3.2.5.

3.2.1 Tandem BLSTM-HMM

In this section, we want to investigate the potential of BLSTM phoneme modeling for continuous speech recognition in a challenging conversational ASR scenario by applying a Tandem BLSTM-HMM system similar to the Tandem BLSTM-DBN keyword spotter presented in Section 3.1.3. As in Section 3.1.3, a Tandem system generating BLSTM phoneme predictions which are incorporated into an HMM framework is created [279]. This allows us to combine Long Short-Term Memory and triphone modeling and leads to higher word accuracies when using the system for decoding continuous, noisy, and spontaneous speech as contained in the COSINE corpus [240, 241].

System Architecture

The structure of the Tandem decoder can be seen in Figure 3.17: s_t and x_t represent the HMM state and the acoustic (MFCC) feature vector, respectively, while b_t corresponds to the discrete phoneme prediction of the BLSTM network (shaded nodes). The HMM uses b_t as observation, in addition to the MFCC features. x_t also serves

as input for the BLSTM and the size of the BLSTM input layer i_t corresponds to the dimensionality of the acoustic feature vector. At each time step, the vector of output activations o_t produced by the framewise BLSTM phoneme predictor contains one probability score for each of the P different phonemes. Analogous to Equation 3.23, b_t is the index of the most likely phoneme:

$$b_t = \underset{j}{\operatorname{argmax}}(o_t^1, ..., o_t^j, ..., o_t^P). \tag{3.31}$$

Thus, at every time step t, the BLSTM generates a phoneme prediction according to Equation 3.31 and the HMM observes both, x_t and b_t using learned emission probabilities $p(x_t, b_t|s_t)$.

Recall that the usage of bidirectional context implies a short look-ahead buffer, meaning that recognition cannot be performed truly on-line. Yet, in many ASR scenarios it is sufficient to obtain an output, e. g., at the end of an utterance, so that both, forward and backward context can be used during decoding.

Experiments and Results

For the experiments presented in this section, the COnversational Speech In Noisy Environments (COSINE) corpus [241] was used. The COSINE corpus is a relatively new database which contains multi-party conversations recorded in real world environments. The recordings were captured on a wearable recording system so that the speakers were able to walk around during recording. Since the participants were asked to speak about anything they liked and to walk to various noisy locations, the corpus consists of natural, spontaneous, and highly disfluent speaking styles partly masked by indoor and outdoor noise sources such as crowds, vehicles, and wind. The recordings were captured using multiple microphones simultaneously, however, to match most application scenarios, we exclusively consider speech recorded by a close-talking microphone (Sennheiser ME-3).

All ten transcribed sessions, containing 11.40 hours of pairwise conversations and group discussions were used. The 37 speakers are fluent, but not necessarily native English speakers. Each speaker participated in only one session and the speakers' ages range from 18 to 71 years (median 21 years).

For the experiments, the recommended test set (sessions 3 and 10) comprising 1.81 hours of speech was applied. Sessions 1 and 8 were used as validation set and the remaining six sessions made up the training set. The vocabulary size is 4.8 k and the out-of-vocabulary rate in the test set is 3.4 %.

All experiments are speaker-independent, meaning that training and testing were performed on data by different speakers. The feature vectors x_t consisted of MFCC coefficients 1 to 12, log. energy, and first and second order regression coefficients. To compensate for stationary noise effects, cepstral mean normalization was applied.

Table 3.6: Framewise phoneme error rate using the COSINE corpus and different network architectures: BLSTM, LSTM, BRNN, and RNN consisting of one and three hidden layers per input direction.

network type	hidden layers	frame error rates [%] train	validation	test
BLSTM	3	23.64	35.76	**33.59**
LSTM	3	30.28	42.89	41.09
BRNN	3	48.74	50.60	49.49
RNN	3	52.37	53.11	51.09
BLSTM	1	26.79	38.16	37.02
LSTM	1	37.69	44.46	42.21
BRNN	1	51.10	51.80	50.09
RNN	1	53.17	54.64	52.85

In order to train and evaluate the quality of phoneme prediction, various network architectures were investigated. As the networks were trained on framewise phoneme targets, an HMM system was applied to obtain phoneme borders via forced alignment. Four different network architectures were evaluated: conventional recurrent neural networks, bidirectional neural networks, unidirectional LSTM networks, and bidirectional LSTM networks. Furthermore, two different variants of the respective architectures were investigated. The first one used a single hidden layer (per input direction) composed of 128 hidden cells and memory blocks, respectively. Each memory block consisted of one memory cell. The second one used the network topology also applied for evaluations in Section 3.1.6, i. e., three hidden layers of size 78, 128, and 80, respectively. The LSTM and BLSTM using three hidden layers per input direction consisted of one backpropagation layer (size 78) and two LSTM layers (size 128 and 80).

For training the common learning rate of 10^{-5} and a momentum of 0.9 was used. Zero mean Gaussian noise with standard deviation 0.6 was added to the inputs during training in order to enhance the generalization capabilities of the networks. The networks were trained on the standard (CMU) set of 41 different English phonemes, including targets for *silence* and *short pause*. Training was aborted as soon as no improvement on the validation set (sessions 1 and 8) could be observed for at least 50 epochs, and the network that achieved the best framewise phoneme error rate on the validation set was chosen as final network.

Table 3.6 shows the framewise error rates on the test, validation, and training set of the COSINE corpus obtained with the different network architectures. Generally, bidirectional context prevails over unidirectional context, LSTM context modeling outperforms conventional RNN architectures, and using three hidden layers leads to better performance than using only one hidden layer. The best error rate can be achieved with a BLSTM network consisting of three hidden layers (35.76 % on the validation set and 33.59 % on the test set).

Table 3.7: Word accuracies (WA) on the COSINE test set for different Tandem models and the baseline HMM recognizer.

network type	layers	WA [%]
BLSTM	3	45.04
LSTM	3	44.46
BRNN	3	42.59
RNN	3	43.79
BLSTM	1	44.27
LSTM	1	43.82
BRNN	1	42.95
RNN	1	43.02
baseline	-	43.36

For continuous speech recognition, the BLSTM phoneme prediction feature is incorporated into an HMM framework for LVCSR where each phoneme is represented by three emitting states (left-to-right HMMs) with 16 Gaussian mixtures. The initial monophone models consisted of one Gaussian mixture per state. All initial means and variances were set to the global means and variances of all feature vector components (flat start initialization). The monophone models were then trained using four iterations of embedded Baum-Welch re-estimation. After that, the monophones were mapped to tied-state cross-word triphone models with shared state transition probabilities. Two Baum-Welch iterations were performed for re-estimation of the triphone models. Finally, the number of mixture components of the triphone models was increased to 16 in four successive rounds of mixture doubling and re-estimation (four iterations in every round). In each round the newly created mixture components were copied from the existing ones, mixture weights were divided by two, and the means were shifted by plus and minus 0.2 times the standard deviation. Both, acoustic models and a bigram language model were trained on the training set of the COSINE corpus.

For the sake of simplicity, the BLSTM phoneme prediction feature was modeled using the same Gaussian mixture framework as for the continuous MFCC features. Since the prediction feature can be interpreted as a discrete index whose absolute value is not correlated to any intensity but rather encodes the most likely phoneme at a given time step, the weights of the Gaussians are used to represent the likelihood of a certain phoneme prediction while being in a given HMM state. By training the weights of the Gaussians, the HMM learns typical phoneme confusions of the BLSTM network that are visible as (lower weighted) Gaussian components in the respective distributions. Generally, the trained Gaussian distributions tend to form single Gaussians of low variance and high weight ('spikes') corresponding to the correct phoneme prediction in a given state as well as the most frequent confusions, and high variance Gaussians of low weight that build a 'floor value' for the phoneme predictions that are not modeled by sharp spikes in the distribution.

Table 3.7 shows the word accuracies on the COSINE test set which were obtained for Tandem modeling using the different network architectures explained before. We can observe a similar trend as for framewise phoneme recognition (Table 3.6): The best performance is achieved with a Tandem model using a BLSTM network that consists of three hidden layers (word accuracy 45.04 %), leading to a significant improvement over the HMM baseline. By contrast, incorporating the phoneme predictions of a conventional RNN leads to similar, or even slightly lower word accuracies when compared to the baseline HMM.

3.2.2 Multi-Stream BLSTM-HMM

Building on the Tandem technique proposed in Section 3.2.1, which uses BLSTM phoneme predictions as additional feature vector components, this section introduces a multi-stream BLSTM-HMM architecture that models the BLSTM phoneme estimate as a second independent stream of observations. As shown in [281], the proposed multi-stream approach allows for more accurate modeling of observed phoneme predictions and outperforms the Tandem strategy outlined in [279] when trained and tested on the COSINE corpus [241]. An on-line version of the proposed multi-stream technique is applied in the SEMAINE system (version 3.0, see Section 2.1), and is available as part of the open-source speech processing toolkit openSMILE [73].

Furthermore, we investigate how feature frame stacking affects the performance of LSTM-based phoneme recognition and Tandem continuous speech recognition, aiming to determine whether learned or predefined context leads to better accuracies (also see [295]). Different bi- and unidirectional network architectures with and without Long Short-Term Memory employing varying ranges of predefined feature-level context are evaluated.

System Architecture

The structure of the multi-stream decoder can be seen in Figure 3.18: Again, b_t corresponds to the discrete phoneme prediction of the BLSTM network (see Equation 3.31). In every time frame t the HMM uses two independent observations: the MFCC features x_t and the BLSTM phoneme prediction feature b_t. With $y_t = [x_t; b_t]$ being the joint feature vector consisting of continuous MFCC and discrete BLSTM observations and the variable λ denoting the stream weight of the first stream (i. e., the MFCC stream), the multi-stream HMM emission probability while being in a certain state s_t can be written as

$$p(y_t|s_t) = \left[\sum_{m=1}^{M} c_{s_t m} \mathcal{N}(x_t; \mu_{s_t m}, \Sigma_{s_t m}) \right]^{\lambda} \times p(b_t|s_t)^{2-\lambda}. \tag{3.32}$$

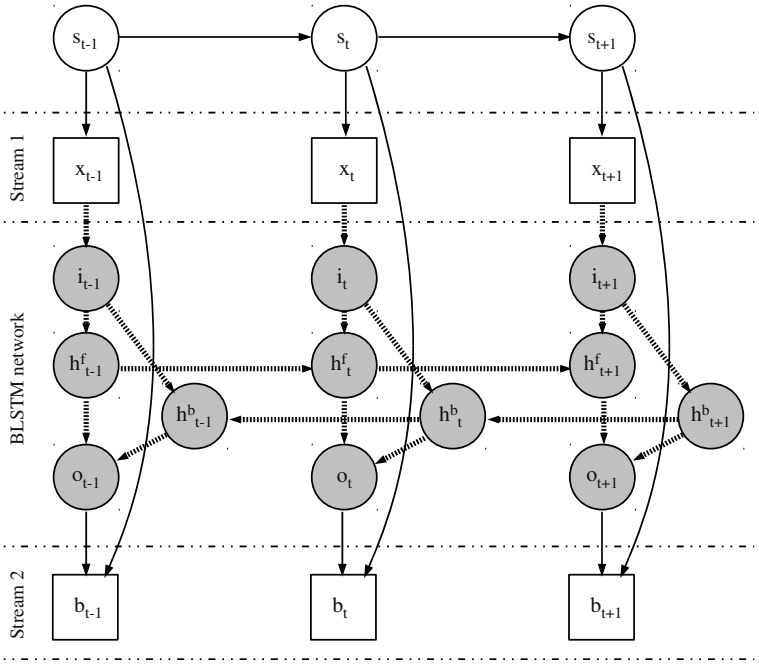


Figure 3.18: Architecture of the multi-stream BLSTM-HMM decoder.

Thus, the continuous MFCC observations are modeled via a mixture of M Gaussians per state while the BLSTM prediction is modeled using a discrete probability distribution $p(b_t|s_t)$. The index m denotes the mixture component, $c_{s_t m}$ is the weight of the m'th Gaussian associated with state s_t, and $\mathcal{N}(\cdot; \mu, \Sigma)$ represents a multivariate Gaussian distribution with mean vector μ and covariance matrix Σ. The distribution $p(b_t|s_t)$ is trained to model typical phoneme confusions that occur in the BLSTM network. In our experiments, we restrict ourselves to the 15 most likely phoneme confusions per state and use a floor value of 0.01 for the remaining confusion likelihoods.

Feature Frame Stacking

A straightforward method to model temporal context within neural networks is to stack a fixed number of n successive frames, so that a *sequence* of feature vectors is presented to the network at each time step. In MLP-based Tandem ASR systems, it is common to stack an equal number of past and future feature frames around the central feature vector x_t. Thus, a sliding window from $t-(n-1)/2$ to $t+(n-1)/2$ is applied to merge n successive feature vectors of size N to an $n \cdot N$ dimensional

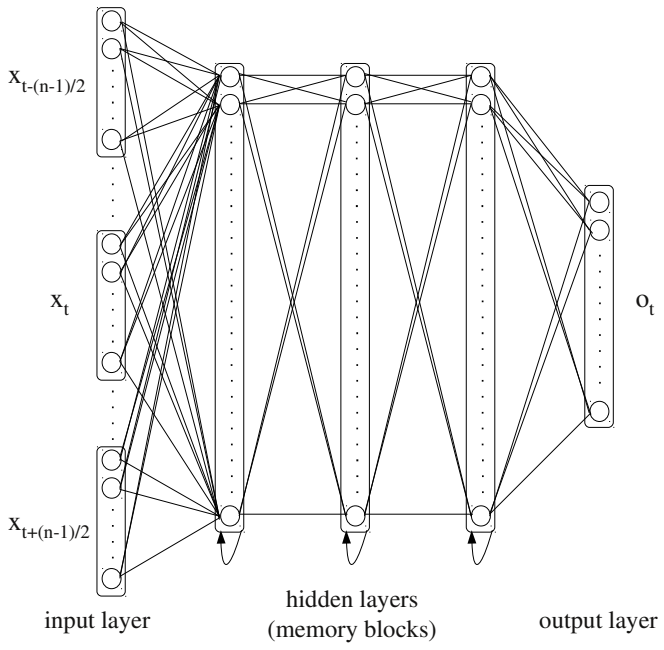


Figure 3.19: Example of a neural network processing n stacked feature frames.

extended feature vector x'_t, i. e.,

$$x'_t = [x_{t-\frac{n-1}{2}}; \dots x_t; \dots x_{t+\frac{n-1}{2}}] \quad \text{for} \quad \frac{n-1}{2} < t \leq T - \frac{n-1}{2}. \tag{3.33}$$

In order to obtain valid vectors for $t \leq (n-1)/2$ and $t > T-(n-1)/2$, the first and the last feature vector of $x_{1:T}$ has to be copied $(n-1)/2$ times. Figure 3.19 shows a schematic example of a network processing n frames to produce a vector of output activations o_t at time t. The network consists of three hidden layers, an input layer of size $n \cdot N$ and an output layer of size P.

Experiments and Results

All networks were trained on framewise phoneme targets obtained via HMM-based forced alignment of the COSINE training set. Feature vectors x_t consisted of 39 normalized MFCC features as in Section 3.2.1. For feature frame stacking, sliding windows of lengths up to $n = 9$ – which is typical for Tandem ASR systems [94] – were evaluated. This corresponds to stacked feature vectors of size 351. Four

Table 3.8: Framewise phoneme error rates (FER) on the COSINE validation and test set using different network architectures and stack sizes of 1 to 9 frames.

FER [%]	**number of frames**				
network type	1	3	5	7	9
validation set					
BLSTM	32.27	32.81	33.42	33.79	34.29
LSTM	40.57	40.84	40.76	40.50	40.26
BRNN	43.62	42.84	43.65	44.05	43.17
RNN	52.18	52.29	50.94	51.40	51.22
test set					
BLSTM	**30.04**	30.86	31.89	31.84	32.02
LSTM	38.21	38.36	37.81	37.67	**37.43**
BRNN	43.07	41.97	42.39	43.04	**41.83**
RNN	51.12	50.47	49.21	49.47	**48.82**

different network architectures were investigated: conventional recurrent neural networks, bidirectional neural networks, unidirectional LSTM networks, and bidirectional LSTM networks. Analogous to [281], all networks consisted of three hidden layers (per input direction) with a size of 78, 128, and 80 hidden units, respectively. The training procedure was identical to the network training applied in Section 3.2.1.

Table 3.8 shows the framewise phoneme error rates when applying different neural network architectures and stack sizes of 1 to 9 feature frames. For bidirectional LSTM networks the error rate increases from 30.04 % to 32.02 % as more successive frames are simultaneously processed. Hence, BLSTM networks seem to learn context better if feature frames are presented one by one and the increased size of the input layer rather harms recognition performance. For unidirectional LSTM networks we observe a different trend: The error rate slightly decreases from 38.21 % to 37.43 % as more frames are processed. This is most likely due to the (small amount of) *future* context information which is available to the LSTM networks if stacking is used and which is not available for fully causal LSTMs observing only one frame per time step. Still, the error rate is notably lower for BLSTM networks. In contrast to BLSTM networks, both, BRNNs and RNNs profit from feature frame stacking: Error rates decrease from 43.07 % to 41.83 % and from 51.12 % to 48.82 %, respectively. This indicates that even though recurrent networks can model a limited amount of context, it is beneficial to introduce a predefined amount of temporal context in the form of stacked feature vectors. However, if we compare the performance of LSTM and RNN architectures, we see that learned LSTM long-range context prevails over feature frame stacking.

Applying the multi-stream BLSTM-HMM system, the word accuracy on the COSINE test set when using the network type with the best framewise phoneme error rate (i. e., the BLSTM architecture) was evaluated. The underlying HMM system

Table 3.9: Word accuracies (WA) on the COSINE test set when applying different multi- and single-stream systems with three hidden LSTM layers (L-L-L) or with one backpropagation and two LSTM layers (B-L-L) and different frame stack sizes (# frames).

system architecture	hidden layers	# frames	WA [%]
multi-stream BLSTM-HMM [295]	L-L-L	1	**48.01**
multi-stream BLSTM-HMM	L-L-L	9	47.17
multi-stream BLSTM-HMM [281]	B-L-L	1	46.50
single-stream BLSTM-HMM [279]	B-L-L	1	45.04
triphone HMM	-	1	43.36

was configured as in [281] and the stream weight variable was set to $\lambda = 1.1$. Starting from the multi-stream system presented in [281], which used a standard backpropagation layer as first hidden layer in the BLSTM network, we observe that replacing the backpropagation layer with a third LSTM layer increases the word accuracy (WA) from 46.50 % to 48.01 % (see Table 3.9). The multi-stream system prevails over the single-stream Tandem approach introduced in [279] (WA of 45.04 %, see Section 3.2.1) and outperforms standard triphone HMMs using only MFCC vectors as observations (WA of 43.36 %). As observed for framewise phoneme classification, feature frame stacking leads to less accurate phoneme estimates if BLSTM networks are applied. This results in a decrease of the word accuracy for continuous speech recognition (WA of 47.17 % for stack size $n = 9$).

3.2.3 BLSTM Front-End for Tandem ASR

In this section, an alternative approach towards BLSTM feature generation for Tandem ASR is presented. We replace the discrete phoneme prediction feature used in Sections 3.2.1 and 3.2.2 by the continuous logarithmized vector of BLSTM output activations and merge it with low-level MFCC features. By that we obtain extended context-sensitive Tandem feature vectors that – given appropriate dimensionality reduction and decorrelation via principal component analysis (PCA) – were shown to give improved results when evaluated on the COSINE [241] and the Buckeye [175] corpora [291].

BLSTM Feature Extraction

The flowchart in Figure 3.20 provides an overview over the ASR system employing BLSTM feature extraction. Cepstral mean and variance normalized MFCC features, including coefficients 1 to 12, logarithmized energy, as well as first and second order temporal derivatives, build a 39-dimensional feature vector which serves as input for the BLSTM network. The common framerate of 10 ms and a window size of

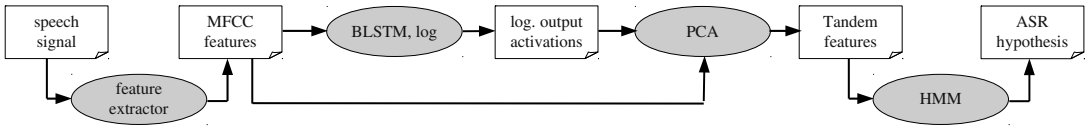


Figure 3.20: Tandem BLSTM front-end incorporated into an HMM-based ASR system.

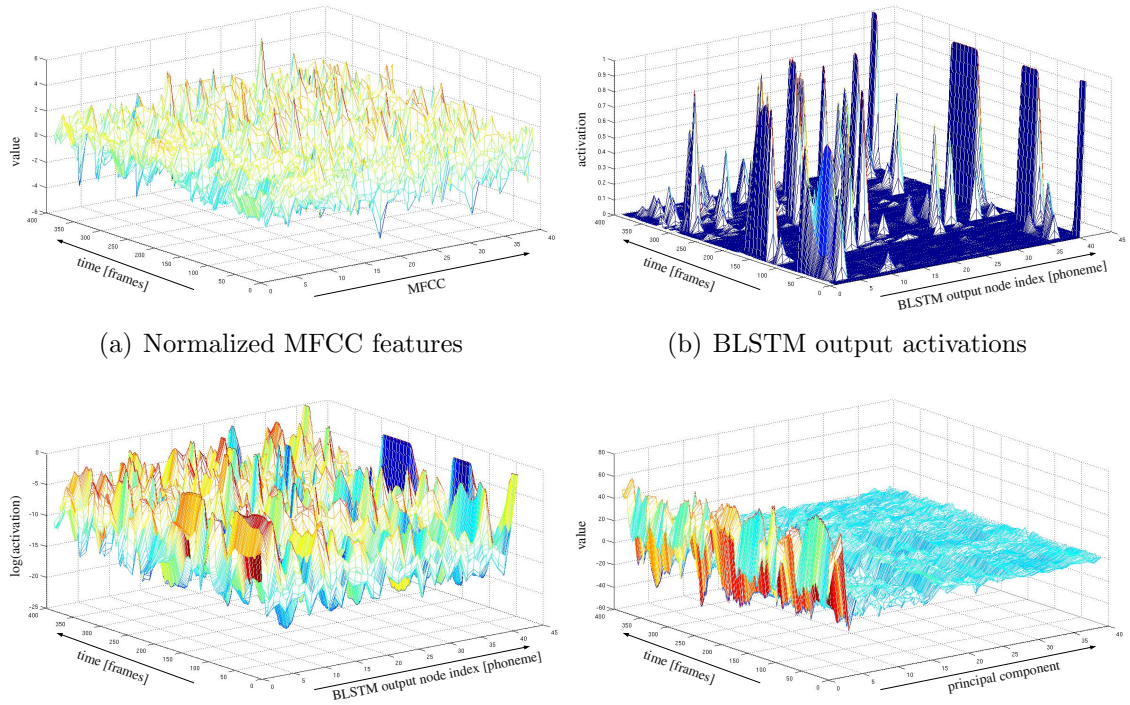
25 ms are used. The BLSTM network is trained on framewise phoneme targets and thus generates a vector of output activations whose entries correspond to estimated phoneme posteriors. Since the network uses a 'softmax' activation function for the output layer (see Equation 2.63), the output activations are approximately gaussianized via mapping to the logarithmic domain. The number of BLSTM features per time frame corresponds to the number of distinct phoneme targets (41 for the COSINE experiment). Merging BLSTM features and the original normalized MFCC features into one large feature vector, we obtain 80 Tandem features that are processed via principal component analysis in order to decorrelate and compress the feature space. The final feature vector is forwarded to an HMM-based ASR system generating the word hypothesis.

Figures 3.21(a) to 3.21(d) show the processing steps for an example speech sequence. The MFCC feature vectors are hardly correlated and approximately follow a Gaussian distribution (Figure 3.21(a)). Due to the softmax activation function generating the outputs of the BLSTM phoneme predictor, the network tends to produce sharp spikes that indicate the presence of a particular phoneme at a particular timestep (Figure 3.21(b)). To gaussianize the outputs, the logarithm is applied (Figure 3.21(c)). Finally, BLSTM and MFCC features are merged and the resulting feature vector sequence is decorrelated via PCA (Figure 3.21(d)).

Experiments and Results

At first, different variants of the proposed Tandem recognizer were trained and evaluated on the COSINE corpus. The underlying BLSTM network was the same as employed for generating the discrete phoneme prediction feature in Section 3.2.2 [295], i. e., the network consisted of three hidden LSTM layers per input direction (size of 78, 128, and 80, respectively) and each LSTM memory block contained one memory cell.

Only the first 40 principal components of the PCA-processed Tandem feature



(a) Normalized MFCC features

(b) BLSTM output activations

(c) Logarithmized BLSTM output activations

(d) Extended and PCA-transformed features

Figure 3.21: Figure 3.21(a): Normalized MFCC features (including deltas and double deltas) over time; Figure 3.21(b): Raw output activations of the BLSTM network. Mapping to the logarithmic domain (Figure 3.21(c)), subsequent concatenation of MFCC features, and PCA transformation (Figure 3.21(d)). Only the principal components corresponding to the 40 largest eigenvalues are shown.

vector were used as input for the HMM recognizer, i. e., the principal components corresponding to the 40 largest eigenvalues. Hence, the HMM back-end was based on the same number of features as the BLSTM-based recognizers proposed in [279, 281, 295]. In conformance with [295], the HMM system consisted of left-to-right HMMs with three emitting states per phoneme and 16 Gaussian mixtures per state. Tied-state cross-word triphone models with shared state transition probabilities were applied, together with a back-off bigram language model, all trained on the training partition of the COSINE corpus.

In Table 3.10, the results on the COSINE test set are summarized. Exclusively applying the raw output activations as BLSTM features leads to a word accuracy of 40.76 %. A slight improvement can be observed when taking the logarithm of the estimated phoneme posteriors (WA of 41.24 %). Decorrelation via PCA further increases the word accuracy to 44.18 % for 40 principal components. Finally, the

Table 3.10: COSINE test set: word accuracies (WA) obtained for the BLSTM front-end with and without taking the logarithm (log) of the BLSTM output activations, decorrelation via PCA, and including MFCC features in the final feature vector (prior to PCA); results are obtained using only the first 40 principal components.

model architecture	**log**	**PCA**	**MFCC**	**WA** [%]
BLSTM front-end + HMM	✗	✗	✗	40.76
BLSTM front-end + HMM	✓	✗	✗	41.24
BLSTM front-end + HMM	✓	✓	✗	44.18
BLSTM front-end + HMM	✓	✓	✓	**48.51**
multi-stream BLSTM-HMM [295]	-	✗	✓	48.01
multi-stream BLSTM-HMM [281]	-	✗	✓	46.50
Tandem BLSTM-HMM [279]	-	✗	✓	45.04
HMM	-	✗	✓	43.36

best BLSTM front-end performance is observed for the system as shown in Figure 3.20, i. e., an HMM processing PCA-transformed feature vectors that contain both, the original MFCC features and the logarithmized BLSTM activations (WA of 48.51 % for 40 principal components). This system prevails over the initial [281] and enhanced [295] version of a multi-stream BLSTM-HMM modeling MFCCs and a discrete BLSTM phoneme prediction feature as two independent data streams. Also a comparable single-stream HMM system modeling the BLSTM prediction as additional discrete feature (WA of 45.04 %, see Section 3.2.1 [279]) as well as a baseline HMM processing only MFCC features (43.36 %) are outperformed by the BLSTM front-end.

3.2.4 Bottleneck-BLSTM Front-End

As indicated in Section 3.2, so-called *bottleneck* features [95] are becoming more and more popular within Tandem ASR systems. Rather than employing the logarithmized and decorrelated activations of the *output* layer of neural networks as probabilistic features, bottleneck front-ends extract the activations of a narrow *hidden* layer within the network. Thus, the size of the resulting feature space can be chosen by adjusting the size of the network's bottleneck layer so that the dimension of the feature vectors is independent of the number of network training targets. Furthermore, the outputs of the bottleneck layer tend to be well decorrelated and do not have to be logarithmized. In this section, we examine how bidirectional LSTM networks can be combined with the bottleneck principle to design a robust and efficient ASR front-end for context-sensitive feature extraction. The Bottleneck-BLSTM system is evaluated on the COSINE and the Buckeye databases in Section 3.2.5 [292, 296].

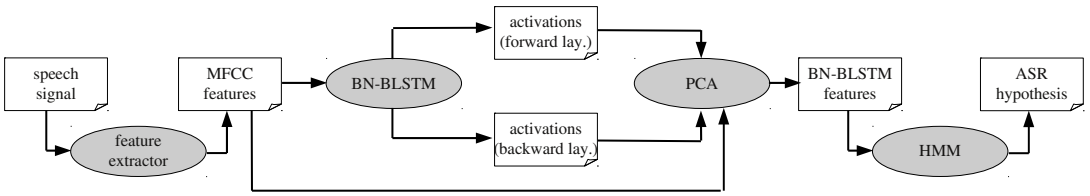


Figure 3.22: Bottleneck-BLSTM front-end incorporated into an HMM-based ASR system.

System Overview

The considered Bottleneck-BLSTM feature extractor can be seen as a combination of bidirectional LSTM modeling for improved context-sensitive Tandem feature generation and bottleneck front-ends. The bottleneck principle allows to generate Tandem feature vectors of arbitrary size by using the activations of the hidden (bottleneck) layer as features – rather than the logarithmized output activations corresponding to the estimated phoneme or phoneme state posteriors. Since we focus on *bidirectional* processing, we have two bottleneck layers: one within the network processing the speech sequence in forward direction and one within the network for backward processing. Figure 3.22 shows the system flowchart of an ASR system based on Bottleneck-BLSTM features. Again, 39 cepstral mean and variance normalized MFCC features are extracted from the speech signal. These features serve as input for a Bottleneck-BLSTM network that is trained on framewise phoneme targets. During feature extraction, the activations of the output layer are ignored; only the activations of the forward and backward bottleneck layer are processed (i. e., the memory block outputs of the bottleneck layers). Together with the original MFCC features, the forward and backward bottleneck layer activations are concatenated to one feature vector which is then decorrelated by PCA.

Bottleneck-BLSTM Feature Extraction

Figure 3.23 illustrates the detailed structure of the applied Bottleneck-BLSTM front-end. The input activations of the network correspond to the normalized MFCC features. Three hidden LSTM layers are used per input direction. Best performance could be obtained when using a hidden layer of size 78 (two times the number of MFCC features) as first hidden LSTM layer, a second hidden layer of size 128, and a comparably narrow third hidden layer, representing the bottleneck (size 20 to 80). The connections between the bottleneck layers and the output layer are depicted in grey, indicating that the activations of the output layer (o_t) are only used during

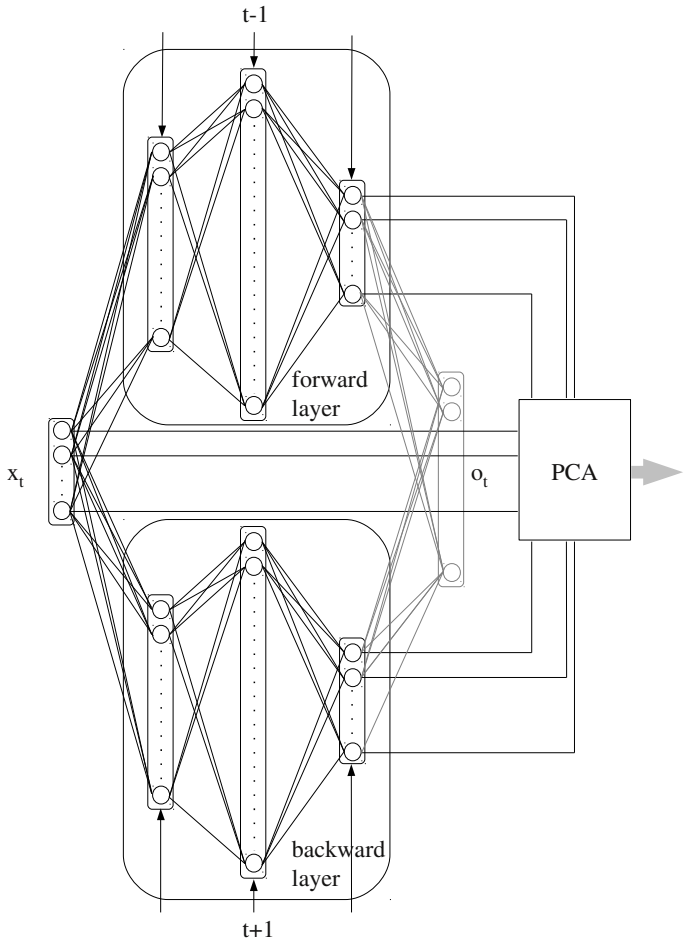


Figure 3.23: Architecture of the Bottleneck-BLSTM front-end.

network training and not during Bottleneck-BLSTM feature extraction. To obtain the final decorrelated feature vectors, PCA is applied on the joint feature vectors consisting of forward and backward bottleneck layer activations and MFCCs x_t.

3.2.5 Evaluation and Discussion

For Bottleneck-BLSTM feature extraction according to the flowchart in Figure 3.22, a number of different network architectures were evaluated. At first, a BLSTM network with a first and third hidden layer of size 128 and a second (bottleneck) layer of sizes 20, 40, and 80 was considered. Best performance could be obtained with a relatively large bottleneck of size 80 (for detailed results, see [296]). Next,

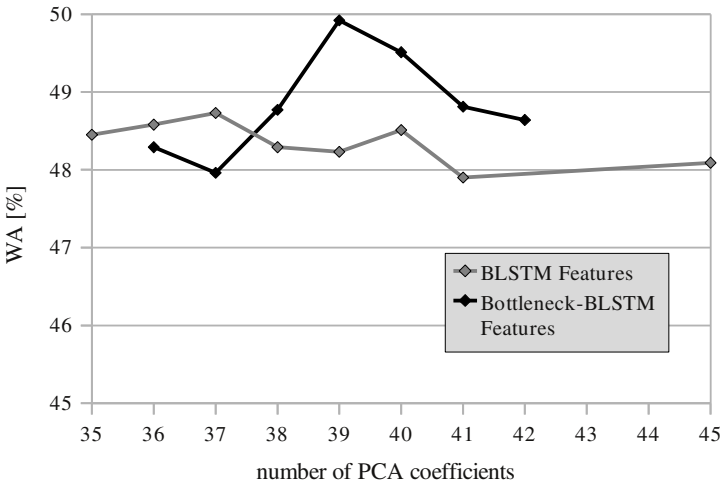


Figure 3.24: Word accuracy (WA) on the COSINE test set as a function of the number of principal components; results are obtained using PCA-transformed feature vectors that contain logarithmized BLSTM activations and MFCC features.

networks such as the one depicted in Figure 3.23 were trained and validated, i. e., networks consisting of a first and second hidden layer of size 78 and 128, respectively, with the third hidden layer used as bottleneck – again evaluating sizes 20, 40, and 80. Network training parameters were set exactly as for the Tandem BLSTM front-end (see Section 3.2.3). Again, only the first 40 principal components were used as final feature vector. The best word accuracy on the COSINE test set was 49.51 % and was achieved with a 78-128-80 hidden layer topology, using the activations of the third hidden layer as features. Thus, prior to PCA, the extended bottleneck feature vector is composed of 199 features (80 activations from the forward hidden layer, 80 activations from the backward hidden layer, and 39 MFCC features). Note that the best hidden layer topology for the Bottleneck-BLSTM front-end was the same as used for Tandem front-end investigated in Section 3.2.3.

Figure 3.24 shows the effect the number of PCA coefficients used as features has on recognition performance for evaluations on the COSINE test set. When applying Tandem BLSTM features, we observe comparable word accuracies for feature vector dimensionalities between 35 and 45, with two maxima for 37 coefficients (WA of 48.73 %) and 40 coefficients (WA of 48.51 %). When employing the Bottleneck-BLSTM feature extractor, we can see a clear global maximum of WA for 39-dimensional feature vectors (WA of 49.92 %). For feature vector sizes larger than 37, the bottleneck system prevails over the Tandem BLSTM front-end.

To investigate the effect of Long Short-Term Memory and bidirectional modeling,

the BLSTM networks in Figures 3.20 and 3.22 were replaced by unidirectional LSTM networks and bi- or unidirectional RNNs, respectively. For experiments on the COSINE database, all LSTM, BRNN, and RNN-based front-ends applied the 78-128-80 hidden layer topology. Prior to using the Tandem and bottleneck features for continuous ASR, the framewise phoneme recognition accuracy of the underlying neural network architectures was evaluated. As can be seen in the second column of Table 3.11, bidirectional LSTM networks perform notably better than unidirectional LSTM nets and that LSTM architectures outperform conventional RNNs.

All word accuracies shown in Table 3.11 are based on feature vectors of size 39 (except for the results taken from [279] and [295], which are obtained using 39+1 features, see Sections 3.2.1 and 3.2.2). The third column of Table 3.11 shows the word accuracies for systems trained and evaluated on the COSINE corpus. When applying bidirectional processing, front-ends using bottleneck activations from the third hidden layer outperform Tandem systems processing the logarithmized output activations. For both front-end types RNN architectures cannot compete with LSTM architectures, which shows the importance of long-range context modeling in challenging spontaneous and disfluent speech scenarios. The Bottleneck-BLSTM features (leading to a WA of 49.92 %) prevail over comparable BLSTM features based on continuous output activations (48.23 %), as well as over the multi-stream BLSTM-HMM technique [295] applying combined continuous-discrete modeling of MFCC features and BLSTM phoneme predictions (48.01 %). The performance difference between the front-ends applying Bottleneck-BLSTM features and BLSTM features derived from the output activations is statistically significant at the 0.002 level when using a z-test as described in [235]. For comparison, the last two rows of Table 3.11 again show the performance of the continuous-discrete BLSTM Tandem system introduced in [279] (45.04 %) and the word accuracy of a baseline HMM processing only MFCC features (43.36 %).

To verify whether word accuracy improvements obtained via BLSTM features can also be observed for other spontaneous speech scenarios, experiments were repeated applying the Buckeye corpus [175] (without further optimizations). The Buckeye corpus contains recordings of interviews with 40 subjects, who were told that they were in a linguistic study on how people express their opinions. The corpus has been used for a variety of phonetic studies as well as for ASR experiments [263]. Similar to the COSINE database, the contained speech is highly spontaneous. The 255 recording sessions, each of which is approximately 10 min long, were subdivided into turns by cutting whenever a subject's speech was interrupted by the interviewer, or once a silence segment of more than 0.5 s length occurred. The same speaker independent training, validation, and test sets as defined in [263] were used. The lengths of the three sets are 20.7 h, 2.4 h, and 2.6 h, respectively, and the vocabulary size is 9.1 k. Since the transcriptions of the Buckeye corpus also contain the events *laughter*, *noise*, *vocal noise*, and *garbage speech*, the size of the network output layers was increased by four from 41 to 45. Thus, the size of the third hidden layer

Table 3.11: Framewise phoneme accuracies (FPA) and word accuracies (WA) for different recognition systems processing activations from the (third) hidden layer (bottleneck) [296], activations from the output layer [291], discrete BLSTM phoneme predictions [279, 295], or conventional MFCCs (HMM). Training and evaluation on the COSINE database or on the Buckeye corpus. Results for bottleneck and Tandem front-ends are based on 39-dimensional feature vectors.

	COSINE		**Buckeye**	
model architecture	**FPA** [%]	**WA** [%]	**FPA** [%]	**WA** [%]
Bottleneck-BLSTM front-end [296]	69.96	**49.92**	69.89	**58.21**
Bottleneck-LSTM front-end	61.79	45.94	61.52	52.53
Bottleneck-BRNN front-end	56.93	41.39	53.40	49.28
Bottleneck-RNN front-end	48.88	40.74	47.05	48.78
BLSTM front-end [291]	69.96	48.23	69.89	57.80
LSTM front-end	61.79	46.68	61.52	53.86
BRNN front-end	56.93	40.67	53.40	48.64
RNN front-end	48.88	40.14	47.05	48.21
multi-stream BLSTM-HMM [295]	69.96	48.01	69.89	56.61
Tandem BLSTM-HMM [279]	66.41	45.04	69.89	55.91
HMM	56.91	43.36	53.20	50.97

was also increased from 80 to 90 to have roughly twice as many memory blocks as phoneme targets in the last hidden layer. As shown in the last column of Table 3.11, the baseline HMM achieves a word accuracy of 50.97 % which is comparable to the result reported in [263] (49.99 %). Accuracies for the Buckeye experiment are notably higher than for the COSINE task since the Buckeye corpus contains speech which is less disfluent and noisy than in the COSINE database. Performance can be boosted to up to 58.21 % when applying the proposed Bottleneck-BLSTM feature extraction. General trends are similar to the COSINE experiment: Again, the Bottleneck-BLSTM principle prevails over the BLSTM multi-stream approach employed in [295].

Finally, it was examined whether part of the performance gap between RNN and BLSTM network architectures can be attributed to the higher number of trainable weights in the BLSTM networks rather than to the more effective context learning abilities of BLSTM front-ends. To this end, an MLP that consists of three layers with sizes 321, 527 and 330 – resulting in a network with 370 345 weights – was trained. The ratio of the sizes of the hidden layers is similar to the BLSTM network with the 78-128-80 hidden layer topology and the total number of weights is comparable to the BLSTM network applied for the COSINE experiment which has 369 249 weights. As the word accuracy obtained with the resulting MLP front-end is 42.72 %, which is slightly lower than for the baseline HMM trained and evaluated on COSINE, we can conclude that simply increasing the size of the network does not lead to better

recognition performance.

This section aimed to show how speech recognition in challenging scenarios can be improved by applying bidirectional Long Short-Term Memory modeling within the recognizer front-end. BLSTM networks are able to incorporate a flexible, self-learned amount of contextual information in the feature extraction process which was shown to result in enhanced probabilistic features, prevailing over conventional RNN or MLP features. We investigated ASR systems, which exclusively use a discrete BLSTM phoneme estimate as additional feature as well as front-ends that generate feature vectors from the continuous logarithmized and PCA-transformed vector of BLSTM output activations. Fusing this concept with the bottleneck technique enables the generation of a well decorrelated and compact feature space that carries information complementary to the original MFCC features. The experiments presented in this section focused on the recognition of spontaneous, conversational, and partly disfluent, emotional, or noisy speech which usually leads to very poor ASR performance. Yet, the Bottleneck-BLSTM technique is able to increase word accuracies from 43.36 to 49.92 % and from 50.97 to 58.21 % for the COSINE and the Buckeye task, respectively.

3.3 Noise Robustness

Enhancing the noise robustness of automatic speech recognition is still an active area of research since one of the most severe limitations of ASR systems is their restricted applicability whenever speech is superposed with background noise [65, 125, 163, 226]. To improve recognition performance in noisy surroundings, different stages of the recognition process can be optimized: As a first step, filtering or spectral subtraction can be applied to enhance the signal before speech features are extracted. Well known examples for such approaches are the advanced front-end feature extraction (AFE) [71] scheme, Unsupervised Spectral Subtraction (USS) [133], or methods based on Non-Negative Matrix Factorization [260]. Then, suitable noise robust features have to be extracted from the speech signal to allow a reliable distinction between the phonemes or word classes in the vocabulary of the recognizer. Apart from widely-used features like MFCCs, the extraction of Tandem features as outlined in Section 3.2 was shown to be effective in noisy conditions [108, 283]. The third stage is the enhancement of the obtained features to remove the effects of noise. Normalization methods like Cepstral Mean Subtraction (CMS) [184], Mean and Variance Normalization (MVN) [249], or Histogram Equalization [53] are techniques to reduce distortions of the cepstral domain representation of speech. Alternatively, model-based feature enhancement approaches can be applied to compensate the effects of background noise. Using a Switching Linear Dynamic Model to capture the dynamic behavior of speech and a Linear Dynamic Model (LDM) to describe additive noise, is the strategy of the joint speech and noise modeling concept in [65]

which aims to estimate the clean speech features from the noisy signal.

The derivation of speech models can be considered as the next stage in the design of a speech recognizer. Even though most systems are based on Hidden Markov Models [183], numerous alternative speech modeling concepts such as Hidden Conditional Random Fields (HCRF) [182], Switching Autoregressive Hidden Markov Models (SAR-HMM) [70], and Autoregressive Switching Linear Dynamical Systems (AR-SLDS) [158] have been proposed in recent years.

Speech models can be adapted to noisy conditions when the training of the recognizer is conducted using noisy training material. Since the noise conditions during the test phase of the recognizer are not known a priori, equal properties of the noises for training and testing hardly occur in reality. However, in case the recognizer is designed for a certain field of application such as an in-car speech recognizer, the approximate noise conditions are known to a certain extent and methods like matched or multi-condition training can be applied [287].

This section outlines a number of different strategies to improve the recognition of noisy speech. First, in Section 3.3.1, an overview over well-known techniques for noise robust speech recognition is provided by evaluating different popular speech and feature enhancement techniques on a simple isolated digit recognition task. Here, the main focus is on feature enhancement via a Switching Linear Dynamic Model which is known to give excellent results in various noisy ASR scenarios [58, 65, 225, 227, 286]. Next, we concentrate on the SEMAINE scenario and investigate how keyword detection in conversational, noisy speech can be improved by multi-condition training [288]. Since the experiments in Sections 3.1 and 3.2 showed that BLSTM modeling of speech results in impressive keyword spotting and ASR performance gains, Section 3.3.3 is devoted to BLSTM-based Tandem ASR systems, aiming to enhance noise robustness by context-sensitive recognition frameworks. In Section 3.3.4, we examine different methods to integrate Long Short-Term Memory into an ASR system and combine them with Non-Negative Matrix Factorization and Non-Negative Sparse Classification (NSC) [81] to design a recognition system that can be applied in noisy multisource environments [301]. Finally, in Section 3.3.5, evaluations on the CHiME task [39] are shown.

3.3.1 Switching Linear Dynamic Models

Feature enhancement techniques attempt to determine the clean speech features from the observed noisy features. This can be done by either using a priori knowledge about how noise affects speech features (Cepstral Mean Normalization, Histogram Equalization [53]) or by building general models for speech and noise (model-based feature enhancement). Recently, extensive evaluations of different noisy speech recognition scenarios [226] led to the finding that modeling speech with a Switching Linear Dynamic Model for model-based feature enhancement as introduced in [65] leads to good results. Feature enhancement algorithms that use an SLDM for speech mod-

eling overcome some of the drawbacks of techniques using, e. g., Gaussian Mixture Models (GMM) or HMMs, since the dynamics of the SLDM capture the smooth time evolution of speech and do not produce artifacts such as sharp single frame transitions.

Modeling of Speech and Noise

As in [65], a Switching Linear Dynamic Model is used to capture the dynamics of clean speech. Similar to HMM-based approaches to model clean speech, the SLDM assumes that the signal passes through various states. Conditioned on the state sequence, the SLDM furthermore enforces a continuous state transition in the feature space. The modeling of noise is done by a simple Linear Dynamic Model obeying the following system equation:

$$x_t = Ax_{t-1} + b + v_t. \tag{3.34}$$

The matrix A and the vector b express how the noise process evolves over time, v_t represents a Gaussian noise source, and x_t denotes the feature vector. As LDM are time-invariant, they are suited for modeling signals like colored stationary Gaussian noise. The following equations can be used to characterize the LDM:

$$p(x_t|x_{t-1}) = \mathcal{N}(x_t; Ax_{t-1} + b, C) \tag{3.35}$$

$$p(x_{1:T}) = p(x_1) \prod_{t=2}^{T} p(x_t|x_{t-1}). \tag{3.36}$$

Here, $\mathcal{N}(x_t; Ax_{t-1} + b, C)$ is a multivariate Gaussian with mean vector $Ax_{t-1} + b$ and covariance matrix C.

The modeling of speech is realized by a more complex dynamic model which also includes a hidden state variable s_t at each time t. Now, A and b depend on the state variable s_t:

$$x_t = A(s_t)x_{t-1} + b(s_t) + v_t. \tag{3.37}$$

Consequently, every possible state sequence $s_{1:T}$ describes an LDM which is non-stationary due to A and b changing over time. Time-varying systems like the evolution of speech features over time can be described adequately by such models. The SLDM can be described as follows:

$$p(x_t, s_t|x_{t-1}) = \mathcal{N}(x_t; A(s_t)x_{t-1} + b(s_t), C(s_t)) \cdot p(s_t) \tag{3.38}$$

$$p(x_{1:T}, s_{1:T}) = p(x_1, s_1) \prod_{t=2}^{T} p(x_t, s_t|x_{t-1}). \tag{3.39}$$

To train the parameters $A(s)$, $b(s)$ and $C(s)$ of the SLDM conventional EM techniques are used (see [65]). In order to obtain a relationship between the noisy observation and the hidden speech and noise features, an observation model has to be defined. In the following, we assume an observation model corresponding to the zero variance observation model with signal to noise ratio (SNR) inference introduced in [66], where speech and noise mix linearly in the time domain corresponding to a non-linear mixing in the cepstral domain.

A possible approximation to reduce the computational complexity of posterior estimation is to restrict the size of the search space applying the generalized pseudo-Bayesian (GPB) algorithm [9]. The GPB algorithm is based on the assumption that the distinct state histories whose differences occur more than r frames in the past can be neglected. Consequently, if T denotes the length of the sequence and S represents the number of hidden states, the inference complexity is reduced from S^T to S^r with $r \ll T$.

If x_t denotes the clean speech features and y_t represents the observed noisy features, the Gaussian posterior $p(x_t, y_{1:t})$ obtained by the GPB algorithm can be used to obtain estimates of the moments of x_t. Those estimates represent the denoised speech features and can be used for speech recognition in noisy environments. The clean features are assumed to be the Minimum Mean Square Error (MMSE) estimate $E[x_t|y_{1:t}]$:

$$E[x_t|y_{1:t}] \cong \frac{\int x_t p(x_t, y_{1:t}) dx_t}{\int p(x_t, y_{1:t}) dx_t}. \tag{3.40}$$

Experiments and Results

The digits 'zero' to 'nine' from the TI 46 Speaker Dependent Isolated Word Corpus [62] are used as speech database for the noisy digit recognition task (for a detailed description of the database, see [226]). All utterances in the test partition of the corpus have been superposed by car noise that was recorded in different cars and at different speeds, resulting in SNR levels between -32 and 5 dB (see [226]). In spite of SNR levels reaching far below 0 dB, speech in the noisy test sequences is still well audible since the recorded noise samples are lowpass signals with most of their energy in the frequency band from 0 to 500 Hz. Consequently, there is no full overlap of the spectrum of speech and noise. Two further noise types were considered: a mixture of babble and street noise at SNR levels 12 dB, 6 dB, and 0 dB and additive white Gaussian noise at SNR levels 20 dB, 10 dB, and 0 dB.

For every digit, an HMM was trained on the clean training corpus to build an isolated word recognizer. Each model consisted of eight states with a mixture of three Gaussians per state. 13 Mel-frequency cepstral coefficients as well as their first and second order derivatives were extracted as features. Attempting to remove the effects of noise, various speech and feature enhancement strategies were applied: Cepstral Mean Subtraction, Mean and Variance Normalization, Histogram Equal-

Table 3.12: Mean isolated digit recognition rates in [%] for different noise types and noise compensation strategies. For each noise type, results are averaged over all evaluated SNR levels. Methods are sorted by mean recognition rate.

	noise type			
speech / feature enhancement method	**clean**	**car**	**babble**	**white**
Switching Linear Dynamic Model [65]	99.92	**99.52**	**99.29**	87.79
Histogram Equalization [53]	99.92	98.21	96.53	77.50
Mean and Variance Normalization	99.84	94.86	93.32	79.06
Cepstral Mean Subtraction	99.84	96.96	97.18	72.22
Unsupervised Spectral Subtraction [133]	99.05	93.52	92.27	53.19
Wiener Filtering [71]	100.0	87.85	92.84	64.14
none	99.92	75.09	88.37	63.67
Autoregressive SLDS [158]	97.37	47.24	78.51	**93.32**
Switching Autoregressive HMM [70]	98.10	54.26	83.16	41.91

ization, Switching Linear Dynamic Models, Unsupervised Spectral Subtraction, and Wiener Filtering. For the training of the global SLDM capturing the clean speech dynamics, all available clean training sequences were used. The SLDM speech model consisted of 32 hidden states. An utterance-specific LDM for noise modeling was derived from the first and last ten frames of the noisy test utterance and consisted of a single Gaussian mixture component.

As can be seen in Table 3.12, in most cases the recognition rate for clean speech is >99 %. For stationary lowpass noise like the car and babble noise types, the best average recognition rate can be achieved when enhancing the speech features using a global Switching Linear Dynamic Model for speech and a Linear Dynamic Model for noise. For speech disturbed by white noise, the best recognition rate (93.3 %, averaged over the different SNR conditions) is reached by the autoregressive Switching Linear Dynamical System introduced in [158]. An AR-SLDS models the noisy speech signal in the time domain as an autoregressive process and can be applied alternatively to HMMs. It can be interpreted as a fusion of a SAR-HMM [70] with an SLDS [226]. The AR-SLDS used in the experiment is based on a 10^{th} order SAR-HMM with ten states. Yet, this concept is not suited for lowpass noise at negative SNR levels: For the car noise type a poor recognition rate of 47.2 %, averaged over all car types and driving conditions, was obtained for AR-SLDS modeling. A possible reason for this is that the AR-SLDS assumes that the additive noise has a flat spectrum (see [158]), which is not true for the lowpass noise types.

The experiments in this section showed that SLDM-based feature enhancement is an effective method to remove the effects of additive noise. Further experiments conducted in [286] revealed that the SLDM concept also leads to improved ASR when incorporated into an LVCSR system for conversational speech recognition. Yet, it is important to note that since the LDM for noise modeling has to be trained on

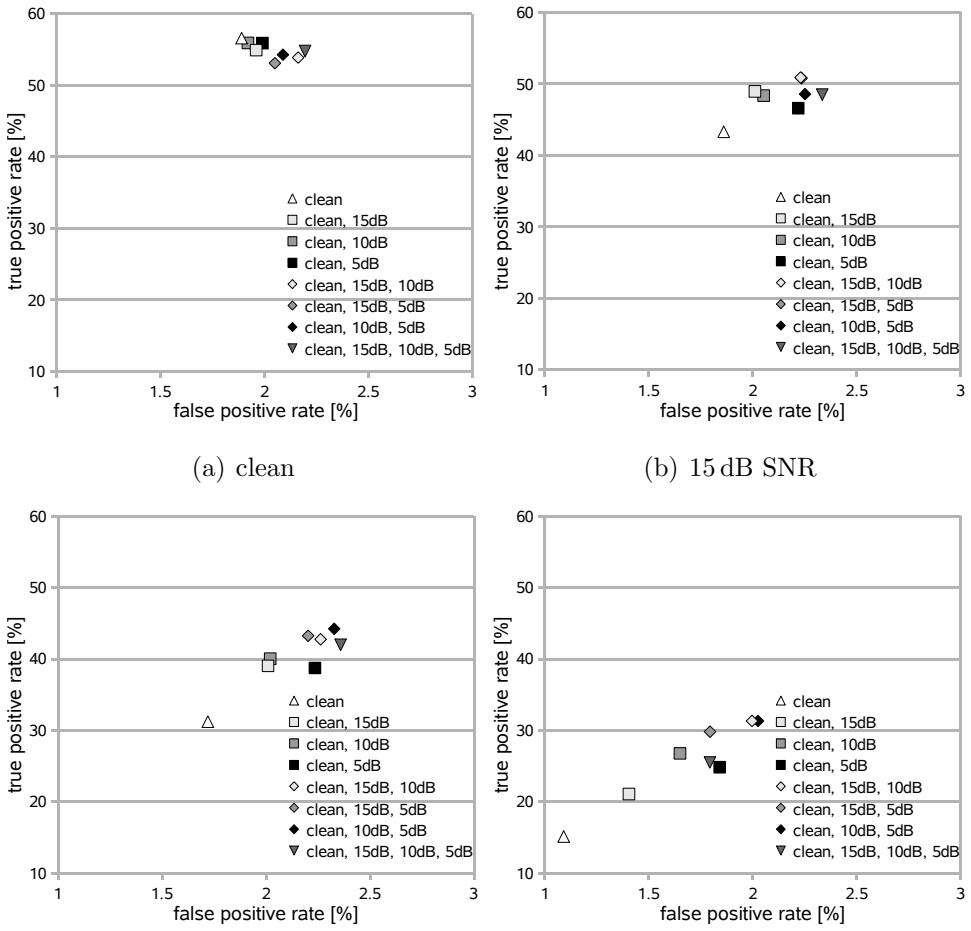
data containing noise only, the applicability of model-based feature enhancement such as the SLDM technique is restricted to scenarios in which some form of a priori knowledge about the noise source exists. By contrast, simpler feature normalization techniques like CMS, MVN, or HEQ can be applied independently of whether there exists some information about the noise characteristics that are expected during recognition.

3.3.2 Multi-Condition Training

So far, we have mostly focused on speech and feature enhancement methods enhancing the noise robustness of the ASR front-end. However, to obtain the best possible recognition performance in noisy conditions, also the back-end of the recognizer needs to be designed in a way that the sensitivity to noise is minimized. A simple and efficient method to improve the noise robustness of the speech recognition back-end is to use matched or multi-condition training strategies [287] by incorporating noisy training material which reflects the noise conditions expected while running the system. In this section, we focus on the SEMAINE scenario (see Section 2.1) in which keywords have to be robustly detected even if the user's speech signal is distorted, e. g., by people talking in the background. We will investigate to what extent models that have been trained on noisy data can maintain the recognition performance in noisy conditions characterized by different realistic SNR levels.

Experiments and Results

In what follows, we investigate the true positive and false positive rates for keyword detection when including noisy speech material in the training process. For all experiments, a part of the training material consisted of unprocessed versions of the SEMAINE database (recordings 1 to 10, see Section 3.1.6), the SAL corpus [64], and the COSINE database [241]. This speech material will be referred to as *clean* in the ongoing (even though the COSINE corpus was partly recorded under noisy conditions). In addition to the 'clean' models, different extensions of the training material were evaluated by adding distorted versions of the SEMAINE and the SAL corpus. To this end, the clean speech was superposed with additive babble noise from the NOISEX database at different SNR levels: 15 dB, 10 dB, and 5 dB. For evaluation, clean and distorted versions of the SEMAINE database (recordings 11 to 19) were used. Since conversational agents such as the SEMAINE system are often used while other people talk in the background, an evaluation scenario including babble noise is most relevant for our application. A set of 173 keywords and three different non-linguistic vocalizations (*breathing*, *laughing*, and *sighing*) were considered. Keyword detection was based on simply searching for the respective words in the most likely ASR hypothesis. The applied trigram language model was trained on the SEMAINE corpus (recordings 1 to 10), the SAL database, and the COSINE database (total



(a) clean

(b) 15 dB SNR

(c) 10 dB SNR

(d) 5 dB SNR

Figure 3.25: ROC operating points obtained for different acoustic models when tested on clean speech and speech superposed by babble noise at 15, 10, and 5 dB SNR; acoustic models were trained on unprocessed versions of the SEMAINE, SAL, and COSINE corpus ('clean') and on noisy versions of the SEMAINE and SAL corpus using different SNR level combinations (babble noise).

vocabulary size 6.1 k). 13 cepstral mean normalized MFCC features along with first and second order temporal derivatives were extracted from the speech signals every 10 ms. All cross-word triphone HMMs consisted of three emitting states with 16 Gaussian mixtures per state. For non-linguistic vocalizations, HMMs consisting of 9 states were trained.

Figures 3.25(a) to 3.25(d) show the ROC operating points for clean test material

as well as for speech superposed with babble noise at 15 dB, 10 dB, and 5 dB SNR, respectively, when using different acoustic models. As can be seen in Figure 3.25(a), models exclusively trained on clean speech lead to the best performance for clean test data. We obtain a true positive rate of 56.58 % at a false positive rate of 1.89 % which is in the range of typical recognition rates for highly disfluent, spontaneous, and emotionally colored speech [273]. Including noisy training material slightly increases the false positive rate to up to 2.20 % at a small decrease of true positive rates. Yet, when evaluating the models on speech superposed by babble noise, multi-condition training significantly increases the true positive rates. A good compromise between high true positive rates and low false positive rates in noisy conditions can be obtained by applying the acoustic models denoted as 'clean, 15 dB, 10 dB' in Figures 3.25(a) to 3.25(d), i. e., models trained on the clean versions of the SEMAINE, SAL, and COSINE corpus, on the SEMAINE and SAL database superposed by babble noise at 15 dB SNR, and on the 10 dB versions of the SEMAINE and SAL database. For test data superposed by babble noise, this training set combination leads to the highest average true positive rate (41.66 %) at a tolerable average false positive rate.

3.3.3 BLSTM Frameworks for Noise Robust ASR

The aim of this section is to examine whether Tandem ASR architectures similar to the keyword spotter introduced in Section 3.1.3, which incorporates context information in the form of phoneme predictions by a bidirectional Long Short-Term Memory network, are more robust with respect to background noise than conventional HMM or DBN-based recognizers. The target application dealt with in this section is noise robust spelling recognition – a functionality that is needed in voice command applications whenever the speech input cannot be restricted to a fixed set of words. For example in-car internet browsers which are already available in today's upper class cars, demand for fast, intuitive, and optionally hands-free operation. While basic browser commands may be covered by a few keywords, entering a URL via speech input presumes an ASR system that also allows for spelling. However, since many letters such as "b" and "d" sound fairly similar, spelling recognition in the presence of driving noise is very challenging – even for humans. In contrast to natural speech, spelling recognition cannot be improved by the usage of a language model but exclusively relies on discriminating the acoustic patterns of different letter utterances. Only for simplified cases such as matching the spelled sequence against a stored dictionary [162] 'language information' can be used.

As shown in Section 3.3.1, strategies for noise compensation, like the SLDM proposed in [65], lead to good performance for speech utterances with predefined speech on- and offset. Yet, their real-life applicability relies on proper discrimination between speech and noise segments [226]. Especially in the interior of a car, where SNR levels are typically negative, voice activity detection is a non-trivial task.

In this section, a Tandem decoder which combines BLSTM neural networks and

DBNs is evaluated with respect to its noise robustness in a spelling recognition task [283]. The modeling of long-range context information is used to learn typical in-car noise characteristics, allowing a better discrimination between speech and noise in the time and frequency domain. Similar to the recognition engine outlined in Section 3.1.3, the Tandem recognizer uses the phoneme predictions of a BLSTM net together with conventional MFCC features to reliably detect spelled letter sequences in driving noise. The resulting model architecture can not only cope with extremely low SNR levels but also with a mismatch between noise conditions during training and testing.

Tandem BLSTM-DBN Decoder

The Tandem BLSTM-HMM decoder for spelling recognition is depicted in Figure 3.26. Since the DBN is based on whole-word modeling of spelled letters, rather than on phoneme modeling, we only have a state and a word layer, which is the main difference between the DBN shown in Figure 3.26 and the keyword spotter DBN in Figure 3.6 consisting of a state, phoneme, and word layer. Furthermore, as the spelling recognizer does not distinguish between *garbage speech* and keywords, there is no 'garbage variable' g_t in the DBN. Similar to the Tandem keyword spotter, w_{t-1}^{tr} is a switching parent of w_t and controls whether w_t is equal to w_{t-1}. If w_{t-1}^{tr} indicates a word transition, a word bigram which makes each word equally likely, but assumes a short silence between two words (or, in our case, between two letters), is used.

Note that the BLSTM network is trained on phoneme targets rather than on words representing spelled letters. This means that the vector o_t contains one probability score for each of the P different phonemes contained in the letters 'a' to 'z'. Again, b_t is the index of the most likely phoneme (see Equation 3.23). The CPFs $p(x_t|s_t)$ are described by Gaussian mixtures while $p(b_t|s_t)$ and $p(s_t^{tr}|s_t)$ are represented by discrete CPFs.

Experiments and Results

For the evaluation of the noise robustness of the Tandem BLSTM-DBN spelling recognizer, the letter utterances from 'a' to 'z' from the TI 46 Speaker Dependent Isolated Word Corpus [62] were used to generate a large set of spelling sequences. The database contains utterances from 16 different speakers – eight females and eight males. Per speaker, 26 utterances were recorded for every letter. Ten samples are used for training and 16 for testing. Consequently, the overall isolated letter training corpus consists of 4 160 utterances while the test set contains 6 656 samples. In order to obtain connected spelling sequences, the isolated letters from every speaker were randomly combined to sequences including between three and seven letters. The silence at the beginning and at the end of the isolated letters was not cut, leading to short silence segments in between the letters. Each individual letter utterance

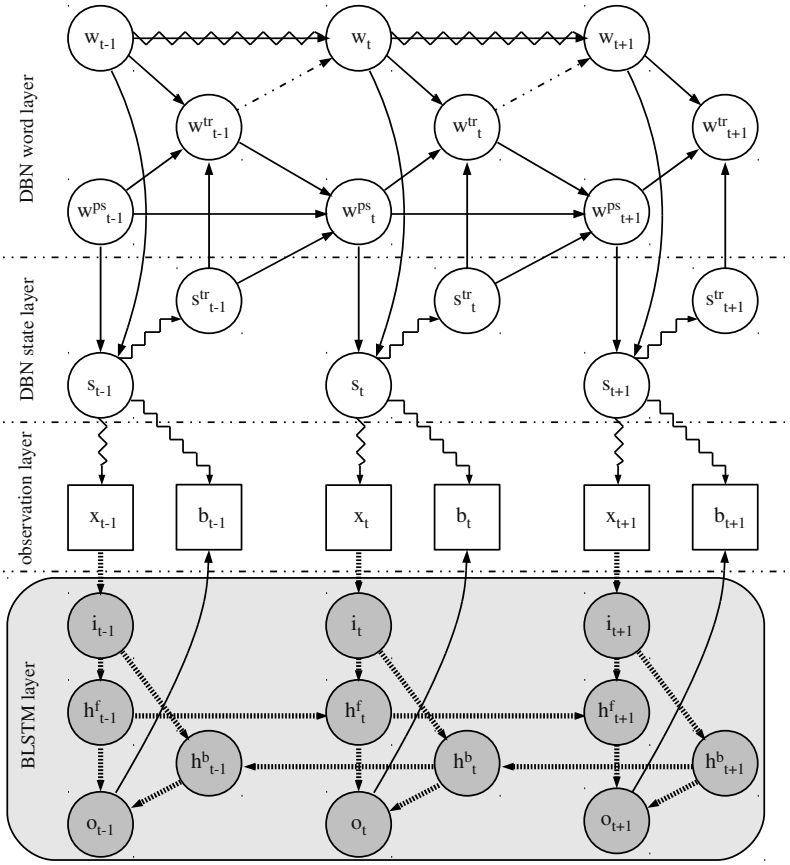


Figure 3.26: Architecture of the Tandem BLSTM-DBN for spelling recognition.

occurs only once within the whole corpus of connected letters. The resulting corpus consists of 839 sequences for training and 1 354 for testing.

Out of the clean spelling utterances, noisy sequences were generated by superposing the speech signal with different in-car noise types as in Section 3.3.1 (see [283] for the resulting SNR histograms). Three different road surfaces in combination with typical velocities have been considered: a smooth city road at 50 km/h, a highway drive at 120 km/h, and a road with big cobbles at 30 km/h.

Feature vectors x_t consisted of 12 cepstral mean normalized MFCC features together with log. energy as well as first and second order delta coefficients. Best results could be obtained when applying a simple FIR highpass filter with a cut-off frequency of 200 Hz in order to partly remove frequency bands that correspond to motor drone etc. before extracting the acoustic features. However, filtering was only

Table 3.13: Spelling recognition accuracies for the Tandem BLSTM-DBN and the DBN (matched and mismatched condition).

training condition	**test condition**	**accuracy [%]**	
		DBN	**BLSTM-DBN**
clean	clean	98.19	98.80
city	city	92.64	96.55
highway	highway	84.06	91.15
cobbles	cobbles	81.65	91.96
city	highway	60.50	77.13
city	cobbles	64.38	79.70
highway	city	54.25	87.51
highway	cobbles	59.09	85.44
cobbles	city	79.07	90.34
cobbles	highway	74.32	87.58
	mean	**74.82**	**88.62**

conducted prior to the extraction of the feature vectors x_t processed by the DBN layer of the Tandem recognizer, whereas the BLSTM network processed MFCC features from unfiltered speech before providing the phoneme prediction b_t as additional feature for the DBN layer.

Each letter HMM consisted of eight states while silence was modeled with three states. In addition to the 'clean' model, one BLSTM-DBN system was trained for every noise condition using the corresponding noisy training material. The BLSTM input layer had a size of 39 (one input for each acoustic feature) and the size of the output layer was 25, corresponding to the 25 different phonemes occurring in the spelled letters from 'a' to 'z'. The network was trained on the forced aligned framewise phoneme transcriptions of the spelling sequences. Both hidden LSTM layers contained 100 memory blocks of one cell each.

Table 3.13 shows the word accuracies for the Tandem BLSTM-DBN recognizer and the corresponding DBN without a BLSTM layer. The first column shows the noise type during training and the second column contains the noise condition during testing. The upper half of the table indicates the matched condition case which is valid whenever the recognition system has exact information about the current velocity and road surface. In the lower half of the table, the mismatched condition case (when noise types during training and testing are different) can be seen. Note that a model trained on perfectly clean data fails in noisy test conditions since the silence model will tolerate no signal variance at all, which would lead to permanent insertion errors. In clean conditions both recognizer architectures show almost perfect performance. As soon as the speech signal is corrupted by noise, performance decreases. In the matched condition case the BLSTM-DBN outperforms the DBN by up to 10 %. Also for the mismatched condition case, the Tandem recognizer is far more robust with respect to noise than the DBN. The greatest improvement can be observed for a recognizer trained on the highway noise type and tested on a smooth

inner city road. There, the Tandem architecture can increase accuracy by 33 %.

In general, we can conclude that BLSTM modeling in a Tandem ASR framework can not only enhance performance in relatively clean conditions as examined in Section 3.1.3, but also leads to better noise robustness. A similar conclusion could be drawn in [298] where a Tandem BLSTM-DBN model for connected digit recognition was proposed and evaluated on the well-known Aurora 2 task [109]. A major difference between the recognizer introduced in [298] and the BLSTM-DBN investigated in this section is that the connected digit recognizer evaluated in [298] uses a DBN that only observes the BLSTM prediction b_t, rather than both, x_t and b_t. Thus, the low-level features x_t only serve as input for the BLSTM network. In conformance with the baseline HMM defined for the Aurora 2 task [109], the DBN for digit recognition applies 16 states per digit model and three states for modeling *silence*. It was trained on clean data and evaluated on the 'set A' test fraction of the Aurora database. On average, the BLSTM-based Tandem model could outperform the baseline HMM system by 7.1 % (see [298]).

3.3.4 Combining NMF and BLSTM for Robust ASR in Multisource Environments

As discussed in Section 3.3, speech enhancement techniques can be applied prior to feature extraction to compensate the effect of noise. In the last decade, monaural source separation techniques by Non-Negative Matrix Factorization have emerged as a promising solution that is portable across application scenarios and acoustic conditions [72, 105, 186, 189, 233]. For instance, the 2006 CHiME Challenge [42] featured an NMF-based approach for cross-talk separation that used speaker models (speech dictionaries) in a supervised NMF framework [204]. In this section, we focus on a convolutive extension of NMF that has delivered promising results for speech denoising (see [233]), and use its capability to model spectral sequences corresponding to the words encountered in the 2011 PASCAL CHiME Challenge recognition task [39].

In addition to speech enhancement techniques, a number of advanced feature extraction approaches have emerged as alternatives to conventional speech features such as MFCCs (see [291], for example). As shown in Section 3.2, an effective approach to enhance the front-end of recognition systems is the application of probabilistic features generated by a neural network that is trained on phoneme or phoneme state targets. Such Tandem systems unite the advantages of discriminative modeling via neural networks and generative frameworks such as HMMs [107]. Due to their ability to exploit long-range context information for phoneme or word prediction, LSTM networks were proven to be especially suited for improving ASR accuracy in challenging conditions [281].

An alternative way to generate framewise phoneme or word predictions that can

be processed in an HMM-based back-end is Non-Negative Sparse Classification (see [81]). If the speech dictionaries are appropriately labeled – e. g., by correspondence to words, phonemes, or HMM states – the activations of their entries directly reveal content of the utterance if sparsity constraints are followed. This has been successfully exploited for *exemplar-based* techniques in speech decoding [81, 112].

In this section, various BLSTM- and NMF/NSC-based ASR architectures that are robust with respect to noise and reverberation are presented and compared. Both, front-end features and back-end decoding of the system are enhanced by using long-range context, exploiting the source separation capabilities of NMF/NSC to complement the context modeling by BLSTM networks. In addition to Tandem BLSTM features, CTC networks that can be used as an alternative to HMMs and can be trained on unsegmented speech data [90] are evaluated on the CHiME task in Section 3.3.5. Further, we examine how the multi-stream BLSTM-HMM recognizer presented in Section 3.2.2 can be enhanced by employing speaker adapted BLSTM predictors.

All systems are evaluated on the PASCAL CHiME corpus [39] which was designed to allow researchers a comparison of their ASR systems in a noisy and reverberated multisource environment. Building on the contribution of the Technische Universität München to the 2011 PASCAL CHiME Challenge [260], we investigate alternative BLSTM-based speech recognition architectures and improve previous results by fully speaker adapted BLSTM networks and Non-Negative Sparse Classification (see also [301]).

The CHiME Corpus

The 2011 PASCAL CHiME Challenge [39] task is to recognize voice commands of the form *command–color–preposition–letter–digit–adverb*, e. g., *"set white by U seven again"*, spoken in a noisy living room. The vocabulary size is 51. For best comparability with the challenge results, we evaluate by the official challenge competition measure, which is keyword accuracy, i. e., the recognition rate of letters (25 spoken English letters excluding 'W') and digits (0–9). The challenge task is speaker dependent. The CHiME corpus contains 24 200 utterances (34 speakers), subdivided into a training (17 000 utterances), development, and test set (3 600 utterances each). These utterances have been created by convolving recordings from the Grid corpus [43] with a binaural room impulse response (BRIR). A different BRIR has been used for each set. The BRIR was measured at a position two meters directly in front of a binaural mannikin. Different BRIRs are obtained by varying the room configuration (e. g., doors open/closed, curtains drawn/undrawn). The development and test sets have been mixed with genuine binaural recordings from a domestic environment, which have been obtained over a period of several weeks in a house with two small children. On top of a quasi-stationary noise floor there are abrupt changes such as appliances being turned on/off, impact noises such as

banging doors, and interfering speakers. The six signal-to-noise ratios employed in the challenge range from 9 dB down to -6 dB in steps of 3 dB; note that the range of SNRs has not been constructed by scaling the speech or noise amplitudes, but instead by choosing different noise segments. More details on the domestic audio corpus and the mixing process can be found in [39]. For the challenge, six hours of pure background noise (divided into seven subsets which were recorded on different days) were provided in addition to the noisy speech. All these data are publicly available at http://spandh.dcs.shef.ac.uk/projects/chime/PCC/datasets.html.

Convolutive NMF for Speech Enhancement

In addition to using LSTM-based ASR architectures in the back-end, the ASR engines evaluated in Section 3.3.5 employ speech enhancement by convolutive Non-Negative Matrix Factorization as in [260]. This is to exploit two – arguably complementary – model-based approaches to coping with noise: using context information in the LSTM back-end, and retrieving a clean speech estimate in the front-end.

The NMF speech enhancement approach is based on the assumption that speech is corrupted by additive noise:

$$V = V^{(s)} + V^{(n)}, \tag{3.41}$$

where $V \in \mathbb{R}_+^{M \times N}$ is an observed magnitude spectrogram of noisy speech, $V^{(s)}$ is the (true) spectrogram of the speech signal, and $V^{(n)}$ is the (true) noise spectrogram. Furthermore, we assume that both, the speech and noise spectrograms, can be modeled as convolutions of base spectrograms (dictionaries) $X^{(s)}(j) \in \mathbb{R}_+^{M \times P}$, $j = 1, \ldots, R^{(s)}$, respectively $X^{(n)}(j)$, $j = 1, \ldots, R^{(n)}$, with non-negative activations $H^{(s)} \in \mathbb{R}_+^{R^{(s)} \times N}$, $H^{(n)} \in \mathbb{R}_+^{R^{(n)} \times N}$:

$$V_{:,t}^{(s)} \approx \sum_{j=1}^{R^{(s)}} \sum_{p=1}^{\min\{P,t\}} H_{j,t-p+1}^{(s)} X_{:,p}^{(s)}(j), \tag{3.42}$$

$$V_{:,t}^{(n)} \approx \sum_{j=1}^{R^{(n)}} \sum_{p=1}^{\min\{P,t\}} H_{j,t-p+1}^{(n)} X_{:,p}^{(n)}(j), \tag{3.43}$$

for $1 \leq t \leq N$. Let $X_{:,j}$, symbolize the j-th column of X as a column vector. Defining

$$W^{(s)}(p) = [X_{:,p+1}^{(s)}(1) \ \cdots \ X_{:,p+1}^{(s)}(R^{(s)})], \tag{3.44}$$

$p = 0, \ldots, P-1$ and $W^{(n)}(p)$ analogously, one obtains an NMF-alike notation of this signal model. Here, the approximation of $V^{(s)}$ and $V^{(n)}$ is denoted by $\Lambda^{(s)}$ and $\Lambda^{(n)}$, and $\overset{p\rightarrow}{\cdot}$ introduces a matrix 'shift' where the entries are shifted p spots to the

right, filling with zeros from the left:

$$\begin{aligned} V &\approx \Lambda^{(s)} + \Lambda^{(n)} \\ &= \sum_{p=0}^{P-1} W^{(s)}(p) \overset{p\rightarrow}{H^{(s)}} + \sum_{p=0}^{P-1} W^{(n)}(p) \overset{p\rightarrow}{H^{(n)}}. \end{aligned} \tag{3.45}$$

In the remainder of this section, we assume that both, $W^{(s)}(p)$ and $W^{(n)}(p)$ can be estimated from training data. The speech enhancement problem is thus reduced to finding non-negative coefficients (activations) $H^{(s)}$ and $H^{(n)}$ that match the observed spectra in V – then, the estimated clean speech spectrogram $\widehat{V}^{(s)}$ is obtained by filtering the observed spectrogram V

$$\widehat{V}^{(s)} = \frac{\Lambda^{(s)}}{\Lambda^{(s)} + \Lambda^{(n)}} \otimes V \tag{3.46}$$

where the symbol $\otimes$ corresponds to the elementwise matrix product. To jointly determine a solution for $H^{(s)}$ and $H^{(n)}$, we iteratively minimize the element-wise sum of the β-divergence d_β between the observed spectrogram V and the approximation $\Lambda := \Lambda^{(s)} + \Lambda^{(n)}$:

$$d_\beta(V|\Lambda) = \sum_{i=1}^{N} \sum_{j=1}^{M} d_\beta(V_{i,j}|\Lambda_{i,j}), \tag{3.47}$$

starting from a (Gaussian) random solution. In NMF-based speech enhancement, using d_1 (equivalent to the generalized Kullback-Leibler divergence) is very popular ([186, 233, 267]), since it seems to provide a good compromise between separation quality and computational effort.

The minimization of d_1 (Equation 3.47) is performed by the multiplicative update algorithm for convolutive NMF proposed by [233] and [257], which can be very efficiently implemented using linear algebra routines employing vectorization. Note that the asymptotic complexity of this algorithm is polynomial ($O(RMNP)$), and linear in each of $R := R^{(s)} + R^{(n)}$, M, N, and P. All experiments in Section 3.3.5 were performed with the NMF implementations found in the open-source toolkit openBliSSART [261] which was developed at the Technische Universität München.

Non-Negative Sparse Classification

As an alternative method to obtain framewise word predictions from a low-level speech feature vector sequence, the principle of Non-Negative Sparse Classification can be applied. It is based on decomposition in the spectral domain rather than long-range context modeling of speech features; similarly to supervised NMF speech enhancement, the main idea is to use the results of spectral factorization directly for speech recognition by determining the sources which contribute to a mixed observation. To this end, the non-negative activation weights of dictionary atoms are

determined by applying sparse NMF. As the identities of the atoms correspond to the phonetic content, phone or word classification can be performed based on the activation weights. In the NSC experiments presented in Section 3.3.5, atoms represent sampled spectrogram patterns and thus are called 'exemplars'. This is in contrast to the approach pursued for speech enhancement, where atoms are learned from training data – in fact, using the very same NSC approach for source separation has been shown to be inferior to the convolutive NMF enhancement pursued in this section [82]. Thus, while there is some methodological overlap between NSC and NMF enhancement, the parametrizations of the algorithms are considerably different and further improvements can be expected when combining them. Further details on the applied NSC technique can be found in [81] and [112].

For NSC, 26 Mel-scale spectral magnitude bands were used as features, employing the common frame size of 25 ms and a 10 ms frame shift. Exemplar windows spanning 20 frames were applied. Each window was factorized independently as in [112]. Other factorization options, including weighting of features, sparsity penalty values and the number of iterations were exactly set as by [112]. For the sparse classification task, 5 000 speaker-dependent speech exemplars and 5 000 noise exemplars were extracted from the training data. This combined speech-noise basis was kept fixed during NMF iterations. After receiving the sparse activation weight vector for each window, the weights and the predetermined label sequences encoding the phonetic information of speech exemplars were used to construct a state likelihood matrix for the observation. For details on this NSC setup and its standalone recognition results in a hybrid ASR system see [112]. In this section, we determine the most likely word identity n_t for each frame t of the observation by summing state likelihoods corresponding to each word. The resulting sequence of word predictions is then used as a feature stream in a multi-stream decoder.

Evaluated ASR systems

In what follows, the basic architecture of the recognition systems evaluated in Section 3.3.5 will be outlined.

- *Baseline HMM*

  The baseline recognition system, as provided by the 2011 CHiME Challenge organizers, employs 51 word-level HMMs [39]. The HMMs use a left-to-right model topology with no state skips. In order to model the different lengths of the words in the vocabulary, two states per phoneme are used. This results in a varying number of states per word (between 4 and 10). State emission probabilities are modeled using seven Gaussian mixture components per state with diagonal covariance matrices.

  The models are trained starting with a single Gaussian and applying iterative mixture splitting and EM training. After each EM iteration, the number of

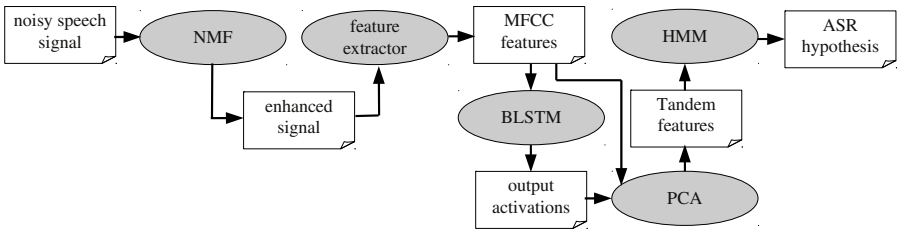


Figure 3.27: Flowchart of the Tandem BLSTM-HMM recognizer processing speech enhanced via NMF.

mixture components is increased by splitting the component with the largest mixture weight. This is repeated until the final number of seven Gaussian mixtures is reached. For recognition, the baseline system uses a grammar which strictly follows the grammar of the Grid corpus utterances.

Several minor modifications of the baseline HMM system were evaluated, including a larger number of Gaussian mixtures (up to 15) and the incorporation of a silence model. However, as these changes of the baseline recognizer did not result in an increased keyword recognition accuracy on the development set, the HMM system as provided by the CHiME Challenge organizers was employed as baseline system.

The features used for the baseline HMMs consist of standard 39-dimensional cepstral mean normalized MFCCs (12 Mel-cepstral coefficients and the logarithmic energy plus the corresponding delta and acceleration coefficients) computed from overlapping frames with a frame length of 25 ms and a frame shift of 10 ms.

- *Tandem BLSTM-HMM Approach*

  As a first attempt to improve the baseline HMM system via feature-level BLSTM modeling, a BLSTM front-end similar to the context-sensitive feature extractor introduced in Section 3.2.3 is evaluated as extension of the standard MFCC features. Thus, a BLSTM network for *framewise* word prediction (without CTC) was trained, i. e., the network inputs correspond to the 39 cepstral mean normalized MFCC features and the resulting output activations represent the posterior probabilities of the 51 words. In each time frame, we obtain a vector of 51 output activations which is logarithmized and appended to the original 39-dimensional MFCC feature vector, resulting in 90 Tandem features per time step. Next, these features are decorrelated using principal component analysis and only the first 40 principal components are applied for HMM-based recognition. A flowchart of the Tandem BLSTM front-end

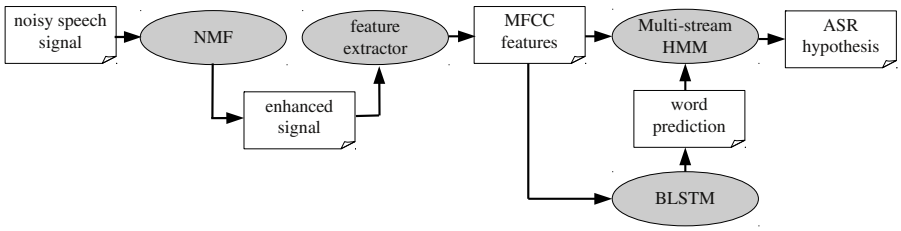


Figure 3.28: Flowchart of the multi-stream BLSTM-HMM recognizer processing speech enhanced via NMF.

processing NMF-enhanced speech can be seen in Figure 3.27.

- *CTC System*

  Using a CTC output layer, a word hypothesis can be obtained without HMM decoding (see Section 2.3.10). Hence, a CTC back-end, replacing the baseline HMM system was built. Again, output activations represent occurrences of words. Note that purely CTC-based recognition is rather suited for small to medium vocabulary tasks, since for large vocabulary ASR the network output layer would get too large. The recognition grammar of the CTC framework is not restricted in any way, meaning that any word can be detected at any time. To determine the keyword recognition rate, we simply take the first letter and digit that are detected in an utterance. Applying the CTC recognizer, two different front-ends were evaluated: the conventional MFCC features and the Tandem BLSTM-MFCC feature extractor explained before.

- *Multi-Stream BLSTM-HMM*

  The multi-stream BLSTM-HMM recognizer outlined in Section 3.2.2 [281] is a further method to integrate LSTM modeling into speech decoding. Employing the same framewise BLSTM word predictor as used within the Tandem front-end, a discrete word prediction feature b_t can be generated for each time step. Similar to Equation 3.31, b_t corresponds to the index of the estimated word that can be obtained by determining the maximum BLSTM output activation:

$$b_t = \underset{w}{\operatorname{argmax}}(o_t^1, ..., o_t^w, ..., o_t^V). \tag{3.48}$$

  In every time frame t the multi-stream HMM uses two independent observations: the MFCC features x_t and the BLSTM word prediction feature b_t. Again, $y_t = [x_t\ b_t]$ denotes the joint feature vector and the variables λ_1 and λ_2 represent the stream weight of the MFCC stream and the BLSTM stream,

respectively, so that the multi-stream HMM emission probability for state s_t is

$$p(y_t|s_t) = \left[\sum_{m=1}^{M} c_{s_t m}\mathcal{N}(x_t; \mu_{s_t m}, \Sigma_{s_t m})\right]^{\lambda_1} \times p(b_t|s_t)^{\lambda_2} \tag{3.49}$$

(also see Section 3.2.2). One advantage of the multi-stream approach compared to the Tandem features is that the BLSTM can be integrated without time-consuming re-estimation of Gaussian mixture components.

Using the development set, the stream weights were optimized independently for speaker independent and speaker adapted BLSTM nets, resulting in an optimum of $\lambda_1 = 1.3$ and $\lambda_2 = 0.7$ for speaker independent networks and $\lambda_1 = 1.1$ and $\lambda_2 = 0.9$ for speaker dependent networks. Figure 3.28 shows a flowchart of the multi-stream BLSTM-HMM.

- *Triple-Stream HMM Exploiting BLSTM and NSC Word Predictions*

  To exploit both, the BLSTM-based word prediction feature and the word prediction obtained via Non-Negative Sparse Classification in addition to the MFCC feature stream, a triple-stream HMM architecture, which can be seen in Figure 3.29 was implemented. Similar to the multi-stream recognition architecture described in Section 3.2.2, the HMM uses continuous MFCC features as well as the discrete BLSTM feature b_t and the word prediction obtained by NSC (n_t) as three independent streams of observations. In contrast to the NSC-only decoder proposed in [112], using NSC in a multi-stream approach along with MFCC and BLSTM predictions can be useful to exploit the properties of spectral (such as additiveness) and cepstral representation (such as a degree of speaker independence) in parallel.

  The triple-stream HMM emission probability in a certain state s_t can be written as

$$p(y_t|s_t) = \left[\sum_{m=1}^{M} c_{s_t m}\mathcal{N}(x_t; \mu_{s_t m}, \Sigma_{s_t m})\right]^{\lambda_1} \times p(b_t|s_t)^{\lambda_2} \times p(n_t|s_t)^{\lambda_3}. \tag{3.50}$$

  Best results on the development set could be obtained when Mel-frequency bands (MFB) that are computed from the raw speech signal (i. e., the signal not enhanced via NMF) are used as input for Non-Negative Sparse Classification (see also Figure 3.29). Stream weights were set to $\lambda_1 = \lambda_2 = \lambda_3 = 1$.

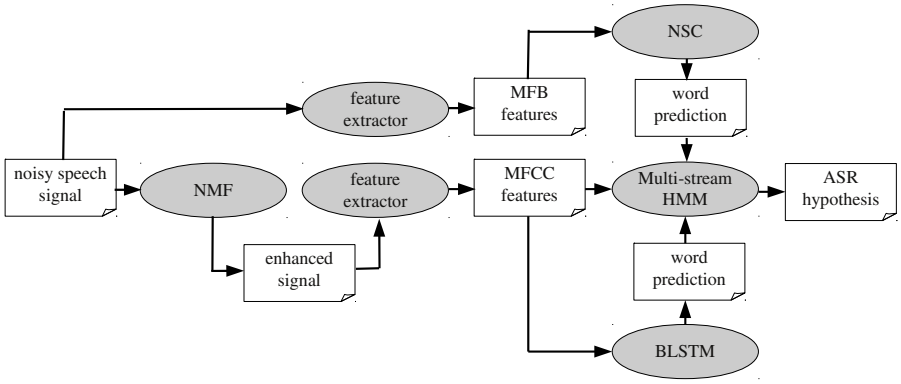


Figure 3.29: Flowchart of the triple-stream recognizer exploiting word predictions obtained via BLSTM and NSC.

3.3.5 Evaluation and Discussion

The experiments presented in this section aim to compare and evaluate the different BLSTM- and NMF/NSC-based noise robust ASR approaches proposed in Section 3.3.4 on the 2011 PASCAL CHiME Challenge [39] task.

Preprocessing

The binaural audio signals were down-mixed from stereo to mono by averaging channels. For NMF speech enhancement, they were transformed to the spectral domain by short-time Fourier transformation using a window size of 64 ms (corresponding to 1 024 samples at a sample rate of 16 kHz) and 75 % overlap, i. e., 16 ms frame shift. This kind of parametrization has been proven to deliver excellent results in speech enhancement [186, 233] at an acceptable computational effort. The square root of the Hann function is used for windowing both in forward and backward transformation in order to reduce artifacts. As in [260], the Mel filter bank for MFCC feature extraction was modified to have a cutoff frequency of 5 000 Hz.

Dictionaries for NMF-based Speech Enhancement

As mentioned in Section 3.3.4, the applied approach for NMF speech enhancement uses convolutive bases of both, speech and noise which are learned from training data. However, in contrast to purely unsupervised learning algorithms for speech dictionaries as proposed, e. g., in [204] using basic NMF and in [233] using convolutive NMF, knowledge about the speech recognition task is exploited already in dictionary learning. This is partly motivated by the study in [185] which showed that in the context of speech enhancement for large vocabulary continuous speech recognition,

incorporating phonetic information into NMF by using phoneme-dependent speech dictionaries is highly beneficial. However, in contrast to that study, which uses single spectra to model phonemes, convolutive NMF is exploited for the fact that it is very well suited for capturing spectral sequences corresponding to words [232]. Hence, convolutive NMF appears to be particularly suited to the small vocabulary CHiME recognition task.

In summary, each dictionary entry corresponds to a 'characteristic' spectrogram of a certain word ($R^{(s)} = 51$) that is learned from training examples. Since speaker-dependent dictionaries are used for the separation, the characteristic spectrograms are obtained from the training set by convolutive NMF as follows. For each of the 34 speakers, the forced alignments obtained by the baseline HMM-MFCC recognizer on the noise-free training set of the CHiME corpus was used to extract all occurrences of each word (51 words in total). Then, for each speaker $k \in \{1, \ldots, 34\}$ and word $w \in \{1, \ldots, 51\}$, the magnitude spectra were concatenated into a matrix $T^{(s,k,w)}$, which was reduced to convolutive base $\omega^{(s,k,w)}(p)$ by a 1-component convolutive NMF,

$$T^{(s,k,w)} \approx \sum_{p=0}^{P-1} \omega^{(s,k,w)}(p) \overset{p\rightarrow}{h}^{(s,k,w)}, \tag{3.51}$$

to form a speaker-dependent dictionary

$$W^{(s,k)}(p) = [\omega^{(s,k,1)}(p) \ \cdots \ \omega^{(s,k,51)}(p)]. \tag{3.52}$$

The parameter P was set to 13 through inspection of the word lengths in the CHiME corpus training set. This corresponds to a spectrogram of a 256 ms signal segment at 64 ms window size and 16 ms frame shift.

In contrast to the speech, the background noise is assumed to be highly variable. Thus, to create a noise dictionary as general as possible, the set of training noise (approximately 6 hours) available for the challenge was sub-sampled, selecting 4 000 random segments of 256 ms length, concatenated into a spectrogram $T^{(n)}$, and reduced to a dictionary $W^{(n)}(p)$. In analogy to the speech dictionary, it contains 51 characteristic noise spectrograms ($R^{(n)} = 51$).

Training and Network Parametrization

For increased robustness, multi-condition training (MCT) is performed by adding noisy speech to the training data. This noisy training data is obtained by mixing all 17 000 training utterances with random segments of the training noise provided in the CHiME corpus. Thus, the complete clean and noisy training database consists of 34 000 utterances. Since the training noise provided by the CHiME Challenge organizers consists of seven different background noise recordings, a larger MCT training set of 136 000 utterances was also evaluated. This set comprised the clean

training utterances as well as seven different noisy versions of the training material, created by superposing the clean utterances with random segments of all seven noise recordings. However, since the performance gain compared to the smaller MCT set was found to be relatively small (at the cost of an increased training time) the smaller MCT set of 34 000 utterances was used for all further experiments.

The BLSTM network applied for generating the Tandem features and the estimates b_t for the multi-stream systems was trained on framewise word targets obtained via HMM-based forced alignment of the clean training set. By contrast, the CTC network was trained on the unsegmented ground truth transcription of the training corpus. Similar to the network configuration used in [281], the BLSTM network consisted of three hidden LSTM layers (per input direction) with a size of 78, 150, and 51 hidden units, respectively. Each LSTM memory block contained one memory cell. The remaining training configurations were the same as those used in [281].

Speaker Adaptation

Various techniques to create speaker adapted recognition systems were investigated: First, speaker dependent HMMs were created by adapting means and variances of the speaker-independent HMMs and performing additional EM iterations using the training utterances for each speaker. This procedure is equivalent to the one applied for the baseline CHiME Challenge results. Second, mean-only MAP adaptation as employed by [260] was applied. Note that for all speaker adaptation methods, only material from the *training* set was used.

Finally, also the BLSTM and CTC networks were adapted by performing additional training epochs using only the training utterances of the respective speaker. All network weights were initialized with the weights of the speaker independent networks and training was aborted as soon as no further improvement on the development set could be observed. Note that for experiments using multi-condition training, multi-condition training data was also used for speaker adaptation.

Development Set Results

Table 3.14 shows the keyword recognition accuracies obtained for the various system combinations on the development set of the CHiME corpus. The first row corresponds to the challenge baseline result (56.30 % mean accuracy) using MFCC features and speaker adapted HMMs [39]. Applying multi-condition training increases the mean performance to 69.85 %. A further gain is obtained by convolutive NMF as detailed in Section 3.3.4, leading to an average accuracy of 80.92 % for a comparable HMM system and to 82.65 % for a MAP adapted recognizer.

- *The effect of speaker adaptation:* As expected, all speaker adaptation techniques increase the keyword recognition accuracies of the respective systems.

Table 3.14: Development set: Keyword recognition accuracies [%] for different SNR levels applying NMF, multi-condition training (MCT), MFCC, Tandem BLSTM-MFCC, or word prediction features (b_t, n_t) in combination with HMM, CTC, or multi-stream (MS) back-ends. Speaker adaptation techniques: MAP adaptation of HMMs and re-training of BLSTM, CTC, and/or HMM recognizers. ⋆no MCT.

			speaker adaptation				SNR						mean
NMF	Features	Back-end	BLSTM	CTC	HMM	MAP	-6 dB	-3 dB	0 dB	3 dB	6 dB	9 dB	
✗	MFCC⋆	HMM	-	-	✓	✗	31.08	36.75	49.08	64.00	73.83	83.08	56.30
✗	MFCC	HMM	-	-	✓	✗	47.25	55.67	66.33	76.08	84.67	89.08	69.85
✓	MFCC	HMM	-	-	✗	✗	63.75	66.33	71.67	75.92	79.92	81.58	73.20
✓	MFCC	HMM	-	-	✓	✗	70.33	76.08	80.08	83.17	88.08	87.75	80.92
✓	MFCC	HMM	-	-	✓	✓	73.58	77.33	82.17	84.25	88.58	90.00	82.65
✓	MFCC	CTC	-	✗	-	-	71.00	73.67	79.50	82.42	87.25	88.75	80.43
✓	MFCC	CTC	-	✓	-	-	77.00	81.00	84.58	87.50	90.58	92.08	85.46
✓	Tandem	HMM	✗	-	✗	✗	75.75	78.05	83.42	85.73	89.58	90.58	83.85
✓	Tandem	HMM	✗	-	✓	✗	74.08	79.72	83.58	86.56	89.17	91.83	84.16
✓	Tandem	HMM	✗	-	✓	✓	77.09	80.38	84.50	87.48	91.00	92.75	85.53
✓	Tandem	HMM	✓	-	✓	✓	78.34	84.72	87.08	89.73	92.33	93.92	87.69
✓	Tandem	CTC	✗	✗	-	-	74.08	78.42	81.92	85.17	88.42	89.67	82.95
✓	Tandem	CTC	✗	✓	-	-	75.92	79.58	83.58	87.08	90.50	90.75	84.57
✓	Tandem	CTC	✓	✓	-	-	79.17	84.25	87.00	89.67	92.08	93.42	87.60
✓	MFCC, b_t	MS-HMM	✗	-	✗	✗	77.08	80.33	84.17	88.08	89.25	90.92	84.97
✓	MFCC, b_t	MS-HMM	✗	-	✓	✗	78.67	81.75	85.67	88.67	90.83	92.58	86.36
✓	MFCC, b_t	MS-HMM	✗	-	✓	✓	81.50	83.00	86.75	90.58	92.25	93.67	87.96
✓	MFCC, b_t	MS-HMM	✓	-	✓	✓	83.36	86.73	90.00	91.49	94.08	95.00	90.11
✓	MFCC, b_t, n_t	MS-HMM	✓	-	✓	✓	86.04	89.48	92.67	94.57	96.25	96.58	**92.60**

For the baseline MFCC-HMM system, a large improvement from 73.20 % to 80.92 % is observed when adapting HMMs by re-training the models employing speaker-specific training material. A further 1.73 % (absolute) gain is reached by MAP adaptation of the HMMs. Interestingly, the performance difference between speaker-independent HMMs and re-trained speaker adapted HMMs is considerably smaller when BLSTM-modeling is applied in the front-end (83.85 % vs. 84.16 % for the Tandem BLSTM-HMM front-end and 84.97 % vs. 86.36 % for the multi-stream BLSTM-HMM). This indicates that BLSTM features are less speaker-specific than conventional MFCCs. Also for CTC back-ends, speaker adaptation boosts recognition performance (80.43 % vs. 85.46 % when using MFCC features and 82.95 % vs. 84.57 % when applying Tandem features). Finally, also framewise BLSTM word predictors tend to produce better Tandem features / word estimates when speaker-specific training material is used to adapt the networks.

- *MFCC features vs. Tandem features:* Tandem features based on bidirectional Long Short-Term Memory modeling (see Section 3.2.3) consistently outperform standard MFCC features: Using speaker adapted networks, performance

can be boosted from 82.65 to 87.69 % for an HMM system and from 85.46 to 87.60 % for a CTC back-end. Note, however, that the performance gain achieved via Tandem features is much smaller when applying a CTC back-end. Thus, BLSTM modeling in the front- and back-end seem to be not fully complementary.

- *HMM vs. CTC back-end:* Replacing the HMM back-end by a CTC network as outlined in Section 2.3.10 enhances ASR performance (82.65 vs. 85.46 % for speaker adapted systems). However, when applying context-sensitive Tandem features, the performance difference between HMMs and CTC networks disappears, which indicates that also HMMs can reach improved performance if long-range context is modeled on the feature level.

- *Methods for BLSTM-modeling:* Overall, the configurations shown in Table 3.14 reflect three different methods to integrate BLSTM context-modeling into an ASR system: using Tandem BLSTM-MFCC features in the front-end, applying a BLSTM-based CTC back-end, and exploiting BLSTM word predictions in a multi-stream HMM framework. When comparing the keyword recognition performances of the individual methods, we see that incorporating BLSTM-modeling in a CTC back-end (85.46 % accuracy) is less effective than employing Tandem features (up to 87.69 % accuracy). The highest average keyword accuracy achieved with systems not performing NSC is 90.11 % and can be obtained with the speaker adapted multi-stream BLSTM-HMM outlined in Section 3.2.2. Hence, the multi-stream architecture seems to be the most effective strategy of applying bidirectional Long Short-Term Memory for noise robust small-vocabulary ASR.

- *Non-Negative Sparse Classification:* The last line of Table 3.14 shows the keyword recognition accuracy of the triple-stream architecture which, in addition to the BLSTM word prediction, also takes into account the word prediction n_t generated via Non-Negative Sparse Classification as described in Section 3.3.4. Compared to the best BLSTM-based multi-stream system (90.11 % accuracy), the triple-stream approach enables a remarkable increase in recognition performance, leading to an average accuracy of 92.60 %. Thus, we can conclude that performance gains achieved via BLSTM word predictors and NSC word predictors are complementary to a certain degree.

Test Set Results

Results on the CHiME test set are shown in Table 3.15. Generally, the same trends as for the development set can be observed. Applying convolutive NMF, multi-condition training, speaker adaptation, BLSTM modeling, and NSC leads to an

Table 3.15: Test set: Keyword recognition accuracies [%] for different SNR levels applying NMF, multi-condition training (MCT), MFCC, Tandem BLSTM-MFCC, or word prediction features (b_t, n_t) in combination with HMM, CTC, or multi-stream (MS) back-ends. Speaker adaptation techniques: MAP adaptation of HMMs and re-training of BLSTM, CTC, and/or HMM recognizers. ⋆no MCT.

NMF	Features	Back-end	speaker adaptation				SNR						mean
			BLSTM	CTC	HMM	MAP	-6 dB	-3 dB	0 dB	3 dB	6 dB	9 dB	
✗	MFCC	HMM⋆	-	-	✓	✗	30.33	35.42	49.50	62.92	75.00	82.42	55.93
✗	MFCC	HMM	-	-	✓	✗	47.67	56.25	67.42	76.50	82.42	88.50	69.82
✓	MFCC	HMM	-	-	✗	✗	65.92	68.33	75.33	77.67	79.92	83.33	75.08
✓	MFCC	HMM	-	-	✓	✗	72.08	76.50	82.08	84.25	87.17	89.17	81.88
✓	MFCC	HMM	-	-	✓	✓	75.58	79.25	84.08	87.67	88.33	90.58	84.25
✓	MFCC	CTC	-	✗	-	-	70.83	76.25	80.17	84.25	86.00	88.50	81.00
✓	MFCC	CTC	-	✓	-	-	74.92	79.25	83.33	88.08	89.50	90.92	84.33
✓	Tandem	HMM	✗	-	✗	✗	75.67	79.22	82.08	87.81	88.17	89.92	83.81
✓	Tandem	HMM	✗	-	✓	✗	76.00	79.97	84.25	87.48	88.58	91.75	84.67
✓	Tandem	HMM	✗	-	✓	✓	77.67	80.72	84.75	88.56	90.00	92.00	85.62
✓	Tandem	HMM	✓	-	✓	✓	80.42	85.64	89.17	91.57	93.00	94.25	89.01
✓	Tandem	CTC	✗	✗	-	-	73.33	77.67	80.83	85.83	86.58	90.25	82.42
✓	Tandem	CTC	✗	✓	-	-	74.42	79.50	82.50	87.58	87.25	91.58	83.81
✓	Tandem	CTC	✓	✓	-	-	80.00	84.33	87.25	90.75	91.92	93.75	88.00
✓	MFCC, b_t	MS-HMM	✗	-	✗	✗	76.58	81.33	83.00	88.25	89.08	91.17	84.90
✓	MFCC, b_t	MS-HMM	✗	-	✓	✗	79.00	82.75	86.58	89.42	89.58	92.67	86.67
✓	MFCC, b_t	MS-HMM	✗	-	✓	✓	80.33	83.50	86.67	90.00	90.25	92.92	87.28
✓	MFCC, b_t	MS-HMM	✓	-	✓	✓	82.92	87.15	90.25	93.66	93.92	94.83	90.45
✓	MFCC, b_t, n_t	MS-HMM	✓	-	✓	✓	84.75	88.31	92.08	93.91	95.67	96.42	**91.86**

impressive increase of keyword recognition accuracy from 55.93 to 91.86 %. Note that when evaluating the test set, the Tandem BLSTM-HMM system as well as the BLSTM-based CTC back-end can both almost reach the performance of multi-stream BLSTM-HMM decoding with an average accuracy of 89.01 and 88.00 %, respectively. However, as for the development set evaluations, the most efficient way to integrate BLSTM is the multi-stream architecture (accuracy of 90.45 %). Again, NSC further improves performance (significance level < 0.005), so that the best result of 91.86 % is obtained with the triple-stream model. This approach slightly outperforms the best CHiME Challenge contribution of 91.65 % average accuracy which was reported in [56]. The system described in [56] is the result of a combination of three different systems exploiting spatial, spectral, and temporal modeling of speech and noise, in addition to dynamic variance adaptation.

3.4 Summary and Outlook

Automatic verbal behavior analysis is an essential precondition for speech-based human-machine interfaces incorporated into conversational agents such as the SE-

MAINE system (see Section 2.1). This chapter provided an overview over recent progress in the field of automatic speech recognition, aiming to advance the state-of-the-art in the extraction of the spoken content from speech signals containing spontaneous, conversational, and partly emotional, noisy, and reverberated speech. We focused on three major ASR-related topics, including reliable keyword spotting (Section 3.1), continuous recognition of conversational speech in challenging scenarios (Section 3.2), and robustness with respect to noise and reverberation (Section 3.3). In all three research fields, we explored novel context-sensitive machine learning methods that go beyond the standard ASR method of using well-known features such as MFCCs for HMM-based speech decoding. Efficient exploitation of contextual information within speech recognition systems is known to be an important aspect that can increase recognition accuracy. While conventional ASR systems model context on multiple levels by including delta features, stacking successive feature frames for probabilistic MLP feature generation, applying triphone HMMs for co-articulation modeling, and using language models in addition to acoustic models, this chapter showed how recognition systems can be enhanced by incorporating a technology developed for context-sensitive sequence labeling with neural networks: the so-called Long Short-Term Memory architecture (see Section 2.3.9). LSTM networks are able to model a flexible amount of temporal long-range contextual information that can be exploited in multiple stages of the speech recognition process.

First, in Section 3.1.1, a discriminative keyword spotter [123] was enhanced by integrating phoneme predictions generated by a bidirectional LSTM network [275]. This vocabulary independent keyword spotting approach is not based on Hidden Markov Models, but on a set of non-linear feature functions and a discriminative learning strategy. Exploiting BLSTM could significantly increase the AUC as shown in experiments on the SAL corpus of emotional human-machine conversations. Next, Section 3.1.2 showed how a Graphical Model can be used for the task of keyword spotting. We derived the explicit graph representation of a GM that can be used to train phoneme models and extended the graph in a way that a set of defined keywords can be reliably detected in continuous speech. The aim was to encode all model assumptions via hidden variables and conditional probability functions in a unified GM framework and to create a basis for investigating architectural modifications and refinements in the following sections. A major advantage of using Graphical Models in general, and *explicit* graph representations in particular, is that they allow for rapid prototyping if the potential of new model architectures shall be investigated (as done in [273], for example). The Graphical Model was designed in a way that it overcomes most of the drawbacks of standard keyword spotting techniques. The model is vocabulary independent meaning that during the training phase no knowledge about the specific set of keywords the system shall be applied for, is necessary. This implies that the GM can be trained on *any* corpus, no matter if and how many times the keywords occur in the training database. It is only in the testing phase that the model needs to know the pronunciations of the keywords

(also see [278]). Moreover, in contrast to many other approaches, the proposed GM does not need an explicitly trained garbage model. It rather uses a hidden garbage variable that serves as a switching parent of the phoneme node in the network. Thus, the model can switch between keywords and non-keyword parts of a speech sequence without requiring a model that was trained on 'garbage speech'.

Applying the DBN structure introduced in Section 3.1.2, Section 3.1.3 showed how the keyword spotter can be extended to a Tandem BLSTM-DBN. The idea was to unite the high-level flexibility of Graphical Models and the low-level context-sensitive sequence labeling capabilities of BLSTM networks to build a keyword spotting system prevailing over the DBN approach. Experiments for the evaluation of the concept focused on a child-robot interaction scenario and investigated the benefit of flexible co-articulation modeling in children's speech via BLSTM networks [293]. The comparison of the Tandem approach with other state-of-the-art keyword spotting techniques showed that the BLSTM-DBN can achieve the same performance as a recently proposed Connectionist Temporal Classification approach [75], which, however, is less flexible since it is based on whole-word modeling. Furthermore, the Tandem technique outperformed an HMM system that is based on triphone modeling rather than on Long Short-Term Memory. Further experiments demonstrated that the proposed Tandem technique is equally well suited for female and male children and that the word spotting performance of the Tandem BLSTM-DBN shows no dependency on the age of the children, while other approaches lead to larger variations of the ROC curves for different age groups and genders. Co-articulation modeling via bidirectional Long Short-Term Memory was shown to increase recognition performance when compared to pure triphone or monophone modeling – especially for younger children who tend to show more variability in their speech production.

Sections 3.1.4 and 3.1.5 introduced two further keyword spotters based on DBNs and BLSTM networks [280]. In contrast to the Tandem BLSTM-DBN, these techniques apply the principle of Connectionist Temporal Classification (see Section 2.3.10), which means that the models can be trained on unsegmented data. Finally, in Section 3.1.6, all of the proposed vocabulary independent keyword detectors were evaluated and compared on two different keyword spotting tasks. All considered approaches exclusively rely on acoustic evidence and do not require an in-domain language model [123, 273, 275, 278, 280, 297]. It was found that the best vocabulary independent keyword spotting performance on read speech can be obtained with the Tandem CTC-DBN approach outlined in Section 3.1.5. For spontaneous speech, purely discriminative modeling in combination with BLSTM prevails over all other investigated methods (see Section 3.1.1).

Section 3.2 demonstrated that BLSTM networks cannot only be applied for enhanced keyword detection, but also for traditional ASR tasks such as the transcription of continuous speech signals. Again, we mainly focused on challenging conversational speaking styles which tend to lead to high ASR error rates. Experiments showed how speech recognition can be improved by applying BLSTM

modeling – either within the recognizer front-end for context-sensitive feature generation [279, 291, 296], or as part of a multi-stream HMM back-end [281]. As BLSTM networks incorporate a self-learned amount of contextual information in the feature extraction process, we were able to obtain enhanced probabilistic features, prevailing over conventional RNN or MLP features. We examined systems using a discrete BLSTM phoneme estimate as additional feature as well as Tandem architectures processing probabilistic feature vectors that are derived from the continuous logarithmized and PCA-transformed vector of BLSTM output activations. Evaluations in Section 3.2.5 revealed that fusing the BLSTM concept with the so-called bottleneck technique [94] enables the generation of a well decorrelated and compact feature space that leads to the best ASR accuracies.

As the third major topic within the field of verbal behavior analysis, Section 3.3 was devoted to techniques that increase the noise robustness of ASR. Sections 3.3.1 and 3.3.2 dealt with different popular approaches such as feature enhancement via SLDM and multi-condition training [65, 226, 287], before Section 3.3.3 examined how Long Short-Term Memory networks can be applied for noise robust ASR [283, 298]. Next, in Section 3.3.4 various LSTM-based frameworks for robust speech recognition that can be applied in high levels of non-stationary background noise and reverberation were proposed. In addition to well-known techniques such as speaker adaptation and multi-condition training, the systems applied convolutive NMF for speech enhancement as well as LSTM to efficiently exploit contextual information. Three different methods to integrate bidirectional LSTM modeling into speech decoding were evaluated: First, we considered a Tandem front-end employing framewise BLSTM word posterior probabilities as features. Second, we examined a CTC-ASR system that uses BLSTM modeling in the back-end and does not need HMMs. Third, a multi-stream system that decodes MFCC features and BLSTM word predictions was built. All three system variants achieved remarkable performance on the CHiME Challenge task, which consists of recognizing digits and letters in a noisy and reverberated multisource environment. Best accuracy could be reached by a fully speaker adapted triple-stream technique which uses Non-Negative Sparse Classification in addition to BLSTM and achieves a 4 % (absolute) performance gain compared to the original challenge submission of the Technische Universität München [260]. As discussed in more detail in [264], this remarkable performance can be attributed to exploitation of complementary methods for noise robustness in different components of the system (NMF speech enhancement, NSC, and BLSTM context modeling). Another interesting result was that CTC networks can be a promising alternative to HMM-based back-ends. The proposed system prevails over previously introduced methods (e. g., [149]) and outperforms the best technique introduced in the context of the PASCAL CHiME Challenge 2011 [56].

The main conclusion of Chapter 3 is that bidirectional Long Short-Term Memory is a promising machine learning architecture that, if integrated into systems for automatic speech recognition, can significantly reduce error rates via context-sensitive

decoding of speech features. Impressive performance gains could be observed in multiple ASR disciplines, including keyword spotting, LVCSR, and noise robust ASR. The experiments in this chapter show the importance of close collaborations between the machine learning community and the ASR community, or, in other words, the necessity of uniting theory and application. Ideas such as the LSTM technique are of limited relevance if their application is restricted to initial proof of concept studies, but become fruitful as soon as they are intelligently integrated into applications like ASR where they replace outdated approaches (such as MLPs, in our example). A clear confirmation of this conclusion can be seen, e. g., in Section 3.3.5, which shows evaluations on the PASCAL CHiME Challenge 2011 task: The idea of LSTM context modeling contributes to a recognition engine that achieves the best recognition results reported until the time of writing.

Of course there are multiple possibilities to improve LSTM-based speech recognition in the future. It would be interesting to investigate combinations of the techniques discussed in this chapter, e. g., by fusing multiple prediction sequences via hybrid fusion methods as outlined in Sections 2.3.4 and 2.3.5. To analyze and understand co-articulation effects in speech on the one hand and the degree of context modeled by LSTM phoneme predictors on the other hand, it might be interesting to examine the sequential Jacobian [89], i. e., the influence of past RNN inputs on the output at a given time step in the phoneme sequence. Moreover, future work should focus on hierarchical BLSTM topologies and on networks trained on phoneme state targets as alternative to phoneme targets. Language modeling with BLSTM networks could be an effective way to enhance word-level context usage. Furthermore, future studies should be devoted to developing a better integration of system components such as BLSTM, NSC, and NMF, i. e., of recognition and enhancement. This could be achieved by iterative methods exploiting decoded phonetic information in speech enhancement and vice versa, such as in [185].

4

Non-Verbal Behavior Analysis

In natural emotion-sensitive human-agent conversation scenarios, such as in the SEMAINE set-up outlined in Section 2.1, not only verbal, but also non-verbal aspects of communication play an important role. When speaking of non-verbal behavior, we refer to conveyed information that goes beyond the spoken content and includes everything that is affect- and emotion-related. Humans have different ways and use different modalities to express emotions or emotion-related states. In multimodal dialogue systems, speech and vision are the most important modalities to be exploited for inferring the affective state of the user. They allow for an automatic estimation of a user's emotion via analysis of speaking style, facial expression, etc. which in turn can be forwarded to the dialogue manager. Based on the estimated user state, the dialogue system can react to the user's current emotion and select appropriate system responses (see also Section 2.1.1 and Figure 2.1).

This chapter introduces machine learning techniques for automatic non-verbal behavior analysis as needed in emotionally intelligent human-computer interaction systems. Based on suitable acoustic low-level speech descriptors such as those mentioned in Section 2.2, Section 4.1 focuses on emotion recognition from the speech signal. In addition to acoustic descriptors like prosodic, spectral, and voice quality features, we will also exploit *linguistic* features, i. e., information extracted from the recognized spoken content in a user's utterance. Thus, keyword spotters and speech recognizers as outlined in Chapter 3 can be interpreted as linguistic feature extractors needed within emotion recognition systems. In Section 4.2, we investigate audio-visual appraoches towards affect recognition by considering also visual features encoding a user's facial expression. Since human emotion is highly context-sensitive, most of the proposed recognition engines will include machine learning frameworks for long-range temporal context modeling, such as the Long Short-Term Memory architecture explained in Section 2.3.9 and advanced in Chapter 3.

4.1 Speech-Based Affect Recognition

For the design of intelligent environments which enable natural human-machine interaction, it is important to consider the principles of interhuman communication as the ideal prototype [252]. While automatic speech recognition is already an integral part of most intelligent systems such as virtual agents, in-car interfaces, or mobile phones, a lot more pattern recognition modules are needed to close or at least narrow the gap between the human ability to permanently observe and react to the affective state of the conversational partner in a socially competent way, and the straightforwardness of system responses generated by today's state-of-the-art human-computer interfaces [49]. Thus, automatic emotion recognition (AER) is an essential precondition to make e. g. virtual agents more human-like and to increase their acceptance among potential users [174, 231, 306].

Even though researchers report outstanding recognition accuracies when trying to assign an affective state to an emotionally colored speech turn [33, 223], systems that apply automatic emotion recognition still are only rarely found in every day life. The main reason for this is that emotion recognition performance is often overestimated: Apart from examples such as call-center data [60, 138, 173], databases for interest recognition [228], or other spontaneous speech evaluations [12, 103, 216, 238, 253], most speech-based AER systems are trained and tested on corpora that contain segmented speech turns with acted, prototypical emotions that are comparatively easy to assign to a set of pre-defined emotional categories [29, 69, 152]. Often, only utterances that have been labeled equally by the majority of annotators are used to evaluate AER performance. Yet, these assumptions fail to reflect the conditions a recognition system has to face in real-life usage. Next generation AER systems must be able to deal with non-prototypical speech data and have to continuously process naturalistic and spontaneous speech as uttered by the user (e. g., as in the Interspeech 2009 Emotion Challenge [219]). More specifically, a real-life emotion recognition engine has to model 'everything that comes in', which means it has to use all data as recorded, e. g., for a dialogue system, media retrieval, or surveillance task by using an *open microphone* setting. According to [237], dealing with non-prototypicality is "one of the last barriers prior to integration of emotion recognition from speech into real-life technology". Thus, in this section we investigate speech-based systems for emotion recognition which are able to cope with spontaneous, non-prototypical, and partly unsegmented speech.

In contrast to static classification scenarios for which pattern classifiers such as Support Vector Machines (see Section 2.3.1) are applied, modern AER is influenced by the growing awareness that context plays an important role in expressing and perceiving emotions [10, 282]. Human emotions tend to evolve slowly over time and utterances observed in isolation might not be sufficient to recognize the expressed emotion. This motivates the introduction of some form of context-sensitivity in emotion classification frameworks. For example, it was shown that AER performance

in dyadic interactions profits from taking into account speech cues from the past utterance of a speaker and his interlocutor [136]. As first shown in [276], capturing temporal long-range dependencies via Long Short-Term Memory modeling (see Section 2.3.9) can enhance the prediction quality of an AER system and is superior to static SVM modeling. Hence, this section mainly focuses on LSTM-based recognition of emotion and emotion-related states like the 'level of interest' (Section 4.1.3). This concept is able to model *emotional history* and – as shown in Section 4.1.2 – enables a completely novel approach towards RNN-based affect recognition that uses low-level features on a frame basis instead of turnwise computed statistical functionals or fixed length feature vector sequences, as applied in other context-independent RNN systems [166].

Section 4.1.1 introduces a speech-based emotion recognition framework using LSTM and investigates the tasks of estimating the quadrant of a continuous two-dimensional emotional space spanned by the two emotional dimensions *valence* and *arousal*. The degree of valence indicates whether the current emotion is rather positive or negative, while the degree of arousal refers to 'excited' vs. 'calm'. Alternatively to quantizing the valence-arousal space to four quadrants, the derivation of data-driven clusters in the emotional space is examined in Section 4.1.1 [277]. In Section 4.1.2, a combined acoustic-linguistic emotion recognition system is proposed [294]. Different LSTM-based modeling techniques are contrasted, including frame-wise and turnwise modeling as well as uni- and bidirectional context exploitation. Next, in Section 4.1.3, we concentrate on estimating a user's level of interest (LOI) [300] using acoustic and linguistic cues in combination with a BLSTM network as back-end. Finally, Section 4.1.4 deals with speech-based recognition of affect in reverberated environments [302].

4.1.1 Data-Driven Clustering in Emotional Space

In most cases, annotators of databases that are used to train and evaluate emotion recognition engines either focus on assigning discrete classes like *anger*, *happiness*, or *neutral* to the emotionally colored speech turns [11, 29] or they try to use continuous scales for predefined emotional dimensions such as *valence*, *arousal*, or *dominance* [64, 97]. Yet, both strategies are suboptimal: In the first case the class division has to be determined in advance, e. g., by defining emotional prototypes that typically occur in a given database. This implies inflexible, fixed classes that can only be changed by combining or splitting certain classes to reduce or increase the 'emotional granularity' [228]. Annotating and modeling emotional dimensions is more flexible and precise since annotation tools like FEELtrace [47] enable a quasi-infinite resolution of human affect. Yet, when evaluating and processing the output of emotion recognizers that provide continuous values for valence, arousal, etc., the emotional continuum has to be discretized again, e. g., in order to reduce the multiplicity of possible system responses of an emotionally sensitive virtual agent. A common

practice is to use a mapping to *quadrants* such as *positive-active*, *positive-passive*, *negative-active*, and *negative-passive* [177]. However, these classes often do not optimally reflect typical emotional states that occur within the training data or are to be expected when applying the emotion recognition engine in a real-world scenario. For example in [48], the *positive-passive* quadrant had to be excluded since it did not occur in the training set. This suggests that a categorization of affective sates in the valence-arousal space should not just involve a simple discretization of the axes but rather closely investigate continuous annotations of the training examples to find meaningful classes.

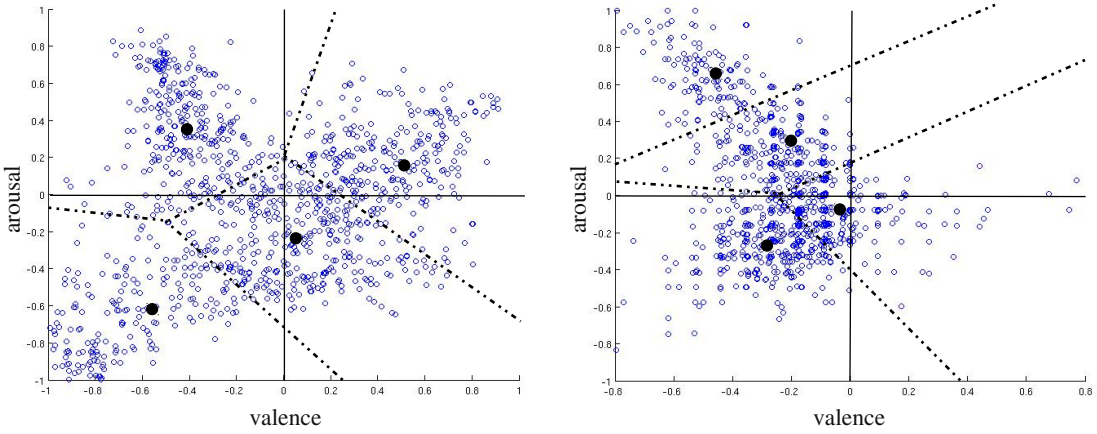
In this section, we investigate a data-driven clustering of the valence-arousal space in order to find classes that better fit the data on which the recognizer is trained, and to optimally model the affective states that actually occur in the specific recognition task. Between two and six emotional states are determined via k-means clustering of the training data. We consider two databases with completely different distributions in emotional space: The Belfast Sensitive Artificial Listener (SAL) database [64] where the occurrence of positive and negative emotions is relatively balanced, and TV talkshow data from the *Vera am Mittag* (VAM) corpus [97] which contains mainly negative emotions. For emotion recognition, both databases imply the great challenge of having to deal with all data – as observed and recorded – and not only with manually selected 'emotional prototypes' as in many other databases.

Databases

The first database is the Belfast Sensitive Artificial Listener corpus which is part of the HUMAINE database [64]. We consider a subset which contains 25 recordings in total from four speakers (two male, two female) with an average length of 20 minutes per speaker. The data contains audio-visual recordings from natural human-computer conversations that were recorded through a SAL interface designed to let users work through a range of emotional states. Data has been labeled continuously in real-time by four annotators with respect to valence and arousal using a system based on FEELtrace [47]. The adjusted values for valence and arousal were sampled every 10 ms to obtain a temporal quasi-continuum. As continuous ground truth label, the mean of the four annotators was used.

The 25 recordings have been split into turns using an energy-based voice activity detection. A total of 1 692 turns is accordingly contained in the database. The turns were randomly divided into training (1 102 turns) and test (590 turns) splits for the experiments. Both sets contain all speakers, thus results are not speaker independent, which in turn would not be feasible with only four speakers. Labels for each turn were computed by averaging the frame level valence and arousal labels over the complete turn.

Finally, k-means clustering (with Euclidean distance) was conducted to find between two and six clusters and the corresponding class borders in a two-dimensional



(a) SAL (b) VAM

Figure 4.1: Annotations of the speech turns in the SAL and VAM databases with cluster midpoints and class borders (dashed lines) determined via k-means clustering.

valence-arousal space. Figure 4.1(a) shows the cluster midpoints obtained for four clusters (black points) as well as the annotations of all utterances in the training set in terms of small circles. While three clusters roughly correspond to the common quadrants, one cluster centre marks an emotional state of neutral valence and slightly negative arousal which can hardly be assigned to one of the quadrants but obviously represents a typical affective user state when interacting with virtual agents.

The second emotional speech corpus used in this section is the VAM database [97]. It contains 947 spontaneous and emotionally coloured utterances from 47 guests of the German talkshow 'Vera am Mittag' and was recorded from unscripted, authentic discussions. For speaker independent evaluation ten speakers were randomly selected for testing while utterances from the remaining 37 speakers were used as training set. A large number of labelers was used to obtain continuous transcriptions for the emotional dimensions valence and arousal (17 labelers for one half of the data, six for the other).

Due to the topics discussed in the talkshow (friendship crises, defalcation, etc.) mostly negative emotions occur in the database. This points out the need to determine emotional clusters that are representative for affective states occurring within the database. Of course we cannot expect an emotion recognition or automatic TV-show annotation system trained on the valence dimension of VAM data to reliably detect utterances of positive valence, since such speech turns hardly occur in the corpus. In the case of four clusters, all cluster midpoints represent negative valence (see Figure 4.1(b)).

Table 4.1: 39 acoustic low-level descriptors.

feature group	features in group	#
signal energy	root mean-square and log. energy	2
pitch	F_0, two measures for probability of voicing	3
voice quality	Harmonics-to-Noise Ratio	1
cepstral	MFCC	16
time signal	zero-crossing-rate, max. / min. value, DC component	4
spectral	energy in bands 0-250 Hz, 0-650 Hz, 250-650 Hz, 1000-4000 Hz	4
	10 %, 25 %, 50 %, 75 %, and 90 % roll-off	5
	centroid, flux, and relative position of max. and min.	4
	sum:	**39**

Feature Extraction

Table 4.1 lists the 39 acoustic low-level descriptors that were extracted from the audio signal to train and evaluate the emotion recognition system. Additionally, first and second order temporal derivatives were used, resulting in 117 features. 51 statistical functionals such as maximum, minimum, mean, quartiles, percentiles, centroids, etc. have been applied, so that the total set consists of 5 967 features. To reduce the feature space dimensionality, relevant features were determined via Correlation-based Feature Subset (CFS) selection. The main idea of CFS is that useful feature subsets should contain features that are highly correlated with the target class while being uncorrelated with each other (for further details, see [101] and [269]). Depending on the classification task, between 102 and 132 features have been automatically selected for the SAL experiment and between 132 and 155 features have been selected for the VAM experiment. All features were normalized to have zero mean and unit variance.

Experiments and Results

In [276], a regression technique was used to train LSTM networks for the prediction of continuous values for valence and arousal under consideration of *emotional history*. In the following, such networks will be referred to as Regression-LSTMs. As an alternative to the regression technique, LSTM networks were discriminatively trained on the discrete clusters in a way that the size of the output layer corresponds to the number of different emotional clusters that shall be distinguished. Thus, for a given speech turn, the activations of the network outputs indicate the probability of the corresponding cluster. The size of the input layer is equal to the number of acoustic features. One hidden LSTM layer containing 100 memory blocks was used. Similar to the networks applied for speech recognition in Chapter 3, zero mean Gaussian noise with standard deviation 0.6 was added to the inputs during training to enhance generalization. All networks were trained using Resilient Propagation (rProp) [191].

For both databases, the performance of discriminatively trained LSTMs, the

Table 4.2: Results (in [%]) for the discrimination of 2, 3, 4, 5, and 6 emotional clusters as well as for the 4 quadrants (4q) when using discriminatively trained LSTM networks ($LSTM_d$), Regression-LSTMs ($LSTM_r$), Support Vector Machines (SVM), or a 'dummy' feature (for chance reference); results are shown for the SAL and VAM database; best F1-measures are highlighted.

database	**SAL**						**VAM**					
# of clusters	**2**	**3**	**4**	**5**	**6**	**4(q)**	**2**	**3**	**4**	**5**	**6**	**4(q)**
$LSTM_d$												
accuracy	77.1	61.0	50.7	41.4	40.0	50.5	82.1	71.3	59.0	45.6	48.2	74.4
recall	67.1	55.5	46.4	40.1	37.5	48.1	80.7	75.8	63.0	50.3	47.4	41.3
precision	77.1	59.5	44.6	36.3	35.2	51.6	75.8	69.2	59.5	47.6	47.6	36.8
F1-measure	**71.7**	**57.4**	**45.5**	**38.1**	**36.3**	**49.8**	78.2	**72.3**	**61.2**	**48.7**	**47.5**	38.9
$LSTM_r$												
accuracy	70.8	47.1	30.9	38.0	27.5	34.9	85.6	72.3	52.8	43.1	43.6	67.2
recall	58.9	48.6	33.4	33.0	27.8	58.9	80.8	71.5	55.5	45.9	41.3	38.8
precision	64.3	50.0	31.0	34.5	24.3	35.4	80.0	71.4	57.8	49.2	32.7	42.5
F1-measure	61.5	49.3	32.2	33.8	26.0	35.6	**80.4**	71.5	56.6	47.5	36.5	**40.6**
SVM												
accuracy	66.1	51.4	38.6	30.0	27.1	41.4	81.5	68.7	53.8	46.2	45.1	71.8
recall	55.3	46.6	38.1	30.3	26.0	41.4	75.1	70.5	56.8	50.1	45.0	41.1
precision	57.6	43.7	34.6	27.9	23.7	42.2	74.4	67.6	56.0	49.2	43.2	48.1
F1-measure	54.9	42.0	32.8	25.2	21.8	38.9	74.7	68.9	56.1	47.9	43.3	40.1
dummy												
accuracy	68.3	60.2	44.1	31.7	30.7	35.9	76.4	51.8	43.1	28.2	33.9	52.3
recall	50.0	33.3	25.0	20.0	16.7	25.0	50.0	33.3	25.0	20.0	16.7	25.0

Regression-LSTMs as used in [276], and SVMs was evaluated on six different emotion recognition tasks: the distinction of two to six emotional clusters as well as the assignment to one of the four quadrants in the valence-arousal space. The SVMs used a polynomial kernel function of degree 1 and Sequential Minimal Optimization (see Section 2.3.1). In contrast to the discriminative LSTM and SVM, the Regression-LSTM outputs continuous values for valence and arousal which were discretized afterwards, according to the clusters and quadrants they would have been assigned to using the minimum Euclidean distance. In order to be able to carry out feature selection separately for valence and arousal, two separate networks (one for valence and one for arousal) have been trained for Regression-LSTM-based emotion recognition while for the discriminative LSTM and for SVM only one classifier has been trained directly on the discrete cluster or quadrant indices to jointly classify valence and arousal.

Table 4.2 shows the performance of the different classifiers for six different recognition tasks using the two databases. For chance reference, the results obtained through a single constant 'dummy' feature resulting in picking the majority class at any time are included. Note that due to unbalanced class distributions, accuracy is a rather inappropriate performance measure. Thus, the F1-measure (harmonic mean between unweighted recall and unweighted precision) was used for performance comparison. As can be seen, the discriminative LSTM outperforms both,

the Regression-LSTM and SVM. Since in the SAL database all quadrants are sufficiently 'occupied' (see Figure 4.1(a)), the F1-measure for the discrimination of four quadrants is slightly higher than for the discrimination of four emotional clusters. However, this is not true for the VAM corpus. Here, two quadrants are almost unoccupied (see Figure 4.1(b)), which leads to better F1-measures for the discrimination of four clusters and highlights the importance of defining class borders according to the application and the database, respectively, rather than just discretizing emotional space to equidistant fields. Apart from the quadrant discrimination and the task of distinguishing two clusters in the VAM corpus, the discriminative LSTM again prevails over the Regression-LSTM and the SVM.

On both datasets, the absolute F1-measure is rather low compared to results for the discrimination of 'prototypical emotions' as published in [223], for example. Yet, in real-life applications of emotion recognition, not only unambiguous emotions have to be classified. The challenge for next-generation emotion recognition systems is rather to develop advanced classifiers using long-range context to continuously deal with all data, as it is necessary for the scenarios considered in this section.

4.1.2 Acoustic-Linguistic Emotion Recognition

As the Sensitive Artificial Listener scenario (see Section 2.1.1) is of utmost relevance for human non-verbal behavior analysis as needed in the SEMAINE system (see Section 2.1), this section focuses on emotion recognition using the SAL database. Unlike in Section 4.1.1, where we considered emotion recognition from acoustic features only, we now extend our analysis to combined acoustic-linguistic recognition of affect. Since the experiments in Section 4.1.1 revealed that in the SAL database, all four quadrants in the emotional space are occupied, we abstain from finding clusters in the emotional space and address the problem of predicting the quadrant of the emotional space (again spanned by the two dimensions *valence* and *arousal*), which best describes the current affective state of the speaker. Consequently, the continuum of emotional states is reduced to the four quadrants which can be described as *happy/excited* (I), *angry/anxious* (II), *sad/bored* (III), and *relaxed/serene* (IV) in order to keep the affective state information as simple as possible. A further motivation for quadrant quantization of the continuous emotional space is to reduce the number of possible system responses for the emotion dependent dialogue management of virtual agents, since at some stage, a categorical decision about the user's emotion has to be made before determining a suitable system output. The AER framework outlined in this section is optimized for usage within virtual agent scenarios such as the SEMAINE system [206, 294], which demands for incremental real-time emotion estimation. Applications like the SEMAINE system require customized and immediate feedback based on the emotional state of the user, and responses have to be prepared already before the user has finished speaking. This, however, would hardly be feasible using traditional static classification approaches

like SVMs which classify segmented or fixed length speech segments at the end of a speech turn. Instead, incremental processing demands for techniques that operate on short speech segments while incorporating an adequate and gradually increasing amount of contextual information.

The experimental part of this section shows that the LSTM principle allows to use low-level features on a frame basis as alternative to turnwise computed statistical functionals of low-level features for classification and regression. The principle of framewise emotion estimation is related to strategies for speech recognition, where the temporal evolution of low-level descriptors is not only captured by functionals of *features* but by the *classifier*. Such an approach has many advantages: It allows for incremental real-time emotion estimation from speech as it is needed for emotionally sensitive virtual agents and does not need to operate on supra-segmental units of speech (as in almost any other method [214, 239, 306]). Moreover, the precondition of perfect segmentation is not needed anymore and the AER system can update the emotion prediction *while* the user is speaking. The Long Short-Term Memory RNN architecture copes with the fact that speech emotion is a phenomenon observed over a longer time window. Typical units of analysis for static classifiers are complete sentences, sentence fragments (i. e., chunks), or words [236]. Yet, finding the optimal unit of analysis is still an active area of research [215, 222, 223]. Unlike HMM-based methods [217, 251] which also focus on low-level features and perform best-path decoding on the complete input fragment, the LSTM technique offers the great advantage that the *amount* of contextual information that is used for emotion recognition is learned during training. In order to refine and update the estimation of a user's emotion once the complete spoken utterance is available, we also investigate the usage of *bidirectional* context (see Section 2.3.8).

In addition to the acoustic features, the system presented in this section also uses linguistic features derived from a Tandem BLSTM-DBN keyword spotter as introduced in Section 3.1.3. Keywords which are correlated to the user's emotion are detected to provide a binary linguistic feature vector that is fused with the acoustic features.

In what follows, we investigate the accuracy of predicting the quadrants of the emotional space as well as the ability to distinguish high from low arousal and valence, respectively. Furthermore, we evaluate the AER performance when considering *neutrality* as a fifth emotional state. We consider both, turnwise and framewise classification using BLSTM, LSTM, SVM, and conventional RNN architectures – with and without linguistic features. In addition to continuously estimating valence and arousal before assigning the prediction to one of the four quadrants, we also investigate discriminative training on the quadrants (as in Section 4.1.1).

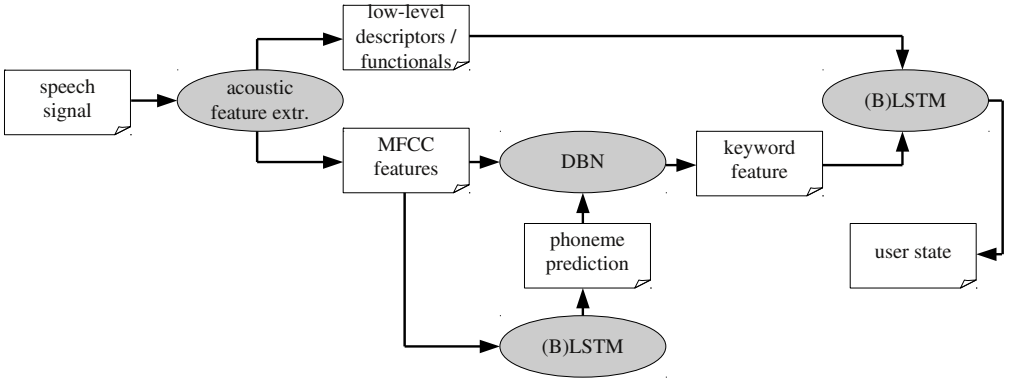


Figure 4.2: Architecture of the acoustic-linguistic affect recognition system.

System Architecture

In Figure 4.2, a flowchart of the considered incremental affect recognition system is shown. Depending on whether framewise or turnwise processing is used, the openSMILE feature extraction module [73] provides either low-level descriptors or statistical functionals of acoustic low-level features to the LSTM network for emotion estimation. Additionally, MFCC features are provided to both components of the Tandem keyword spotter, consisting of a DBN and a further LSTM network for phoneme prediction. Together with the produced phoneme predictions, the MFCC features are observed by the DBN, which then can detect the occurrence of a relevant keyword (i. e., a word that is relevant for valence or arousal prediction). Both, the discrete keyword feature and the acoustic features extracted by openSMILE are used by an LSTM network to predict the user's current emotion.

Acoustic Feature Extraction

The 28 low-level descriptors extracted from the audio signal for time-continuous emotion recognition are summarized in Table 4.3 (column 'C'). The descriptors were extracted every 20 ms for overlapping frames with a frame-length of 32 ms. First order regression coefficients are appended to the 28 low-level descriptors, resulting in a 56 dimensional feature vector for each frame.

For turn-based emotion recognition experiments, we follow the traditional approach of generating a large set of features by applying statistical functionals to low-level descriptor contours. Thus, alternatively, an extended set of 39 low-level descriptors detailed in Table 4.3 (column 'T') is extracted, first and second order delta coefficients are appended, and 36 functionals are applied to each of the resulting 117 low-level descriptor contours, resulting in a total of 4 212 features. The 36

Table 4.3: 28 low-level audio features for time-continuous emotion analysis (C) and 39 features for turn-based recognition (T); features in italics are used for both, continuous and turn-based recognition.

feature group	**features in group**	**#(C)**	**#(T)**
signal energy	*root mean-square* and log. energy	1	2
pitch	F_0, two measures for probability of voicing	1	3
voice quality	*Harmonics-to-Noise Ratio*	1	1
cepstral	MFCC 0, *MFCC 1-12*, MFCC 13-15	12	16
time signal	*zero-crossing-rate*, max. / min. value, DC component	1	4
spectral	*energy in bands 0-250Hz, 0-650Hz, 250-650Hz, 1000-4000Hz*	4	4
	10 %, 25 %, 50 %, 75 %, and 90 % roll-off	5	5
	centroid, flux, and relative position of max. and min.	3	4
	sum:	**28**	**39**

functionals include maximum / minimum values and relative positions, range (max.-min.), mean and mean of absolute values, max.-mean, min.-mean, quartiles and inter-quartile ranges, 95 % and 98 % percentiles, standard deviation, variance, kurtosis, skewness, centroid of contour, linear regression coefficients and approximation error, quadratic regression coefficients and approximation error, zero-crossing-rate, 25 % down-level time, 75 % up-level time, rise-time, and fall-time (see also [294]).

The 4 212 features for turn-based emotion recognition are reduced to relevant features for arousal and valence independently by a Correlation-based Feature Subset selection (see Section 4.1.1). Conducting CFS for turn-based emotion recognition via regression resulted in 60 features being selected for arousal and 64 features for valence. As termination criterion a maximum of five non-improving nodes before terminating the greedy hill climbing forward search was considered (see [269]). Binary targets for arousal and valence (high vs. low) led to the selection of 110 and 55 features, respectively. For the discriminative four-class quadrant classification task 121 features were selected, and for the five-class task applying CFS resulted in 123 selected features. For framewise emotion recognition the full set of $28 \cdot 2 = 56$ features was used without further reduction. As in Section 4.1.1, all features (turn-based functionals and low-level features) were standardized to have zero mean and unit standard deviation. These parameters were computed from the training data only and applied to both, training and test data.

Linguistic Feature Extraction

Not only acoustic features, but also spoken or written text carries information about the underlying affective state [8, 40, 67]. This is usually reflected in the usage of certain words or grammatical alterations. A number of approaches exist for this analysis: keyword spotting [46, 68], rule-based modeling [146], Semantic Trees [309], La-

tent Semantic Analysis [88], Transformation-based Learning [303], World-knowledge Modeling [147], key-phrase spotting [218], and Bayesian Networks [28, 193]. Two methods seem to be predominant, presumably because they are shallow representations of linguistic knowledge and have already been frequently employed in automatic speech processing: (class-based) N-Grams [7, 59, 137, 178] and vector space modeling [13, 213]. In emotion recognition, mostly unigrams have been applied so far [59, 137]. The technique applied in this section is related to Bag of Words (BoW) modeling [13, 120, 213] via keyword spotting, however, when applying framewise emotion recognition, only one keyword can be present at a given time frame. In the case of turnwise AER, the linguistic feature vector can contain more than one keyword. This would enable techniques like (Bag of) N-Gram modeling or other forms of linguistic information integration [210], which however were not used in order to allow a fair comparison between framewise and turnwise affect recognition.

For combined acoustic and linguistic AER, the acoustic feature vector is extended by appending binary linguistic features. Each binary feature corresponds to the occurrence of one of the 56 keywords that were shown to be correlated to either valence or arousal. Note that using a single linguistic feature containing the current word identity in form of a word index would not be feasible with LSTM networks since they assume that the absolute value of a feature is always correlated or proportional to the 'intensity' of the corresponding feature. This, however, would not be true for a 'word index feature'.

When applying framewise acoustic-linguistic analysis, a short buffer has to be included in order to allow the keyword spotter to provide the binary features *after* the keyword has been decoded. Yet, this causes only a short delay as linguistic features can still be delivered while the user is speaking. In order to reduce the vocabulary to a small set of emotionally meaningful keywords, CFS feature selection was applied on the training set. Pace Regression [258] based CFS used the continuous labels for valence and arousal for Bag of Words keyword selection with a minimum term frequency of two (without stemming). Keywords like *again*, *angry*, *assertive*, *very* etc. were selected for arousal, and typical keywords correlated to valence were, e. g., *good*, *great*, *lovely*, or *totally*. For keyword spotting, the Tandem BLSTM-DBN outlined in Section 3.1.3 was applied. Phoneme models were trained on the TIMIT database and adapted using the training split of the SAL database to allow a better modeling of emotionally colored speech. All means, variances, and weights of the Gaussian mixture probability distributions $p(x_t|s_t)$, as well as the state transition probabilities $p(s_t^{tr}|s_t)$ were re-estimated until the change of the overall log likelihood of the SAL training set became less than 0.02 % (see also Sections 3.1.2 and 3.1.3). The BLSTM network of the Tandem keyword spotter consisted of 100 memory blocks of one cell each for each input direction. All other DBN and BLSTM parameters correspond exactly to those applied in Section 3.1.3. Using these settings, the keyword spotter achieved a true positive rate of 0.59 at a false positive rate of 0.05 on the test partition of the SAL corpus.

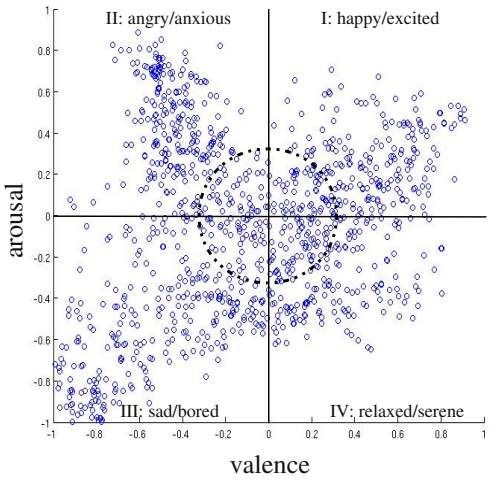


Figure 4.3: Turnwise annotations of the SAL database.

Experiments and Results

In all of the following experiments, the SAL database (see Section 4.1.1) was applied for training and testing. More details on the annotation process and database characteristics can be found in [294]. In order to fit the requirements of the SEMAINE dialogue management [206], the recognition framework was designed in a way that it estimates the current quadrant in the two-dimensional valence-arousal space. In addition to quadrant classification, we also investigate a five-class task including a 'neutral' state, as well as the (two-class) discrimination of low and high valence and arousal, respectively. The distribution of the averaged continuous-valued labels can be seen in Figure 4.3. The dashed circle (with a radius of 0.33, dividing the axes into thirds) in the center of the valence-arousal space marks a fifth region which represents a neutral emotional state. For the five-class task, the coordinates that lie within this circle will be considered as belonging to a fifth, neutral class.

For quadrant prediction two different strategies were followed: First, LSTM networks for regression were trained to obtain continuous predictions for valence and arousal which were then mapped onto one of the four quadrants. In order to perform feature selection independently for both, the valence and the arousal dimension, separate networks were used for the two dimensions. Alternatively, the continuous labels for the emotional dimensions were mapped *before* training the network in order to allow a discriminative training on the quadrants, following the strategy applied in Section 4.1.1. These two strategies were also evaluated for the five-class task and for both of the two-class tasks (discrimination of low vs. high arousal and valence, respectively).

For each of the two techniques, traditional turnwise classification with statisti-

cal functionals of acoustic features and framewise classification using only low-level features was evaluated. The gain of appending the binary keyword feature vector for combined acoustic-linguistic affect recognition was examined for every recognizer configuration.

The size of the LSTM input layer corresponds to the number of selected acoustic and linguistic features, while the size of the output layer is equal to the number of regression/classification targets (one, two, four, and five, respectively). Each LSTM network consists of one hidden layer with 50 memory blocks of one LSTM cell each. The BLSTM networks have two hidden layers of 50 memory blocks, one for each direction (forwards, backwards). For the acoustic-linguistic experiments, the LSTM network size was increased to 70 memory blocks due to the increased size of the combined acoustic-linguistic feature vector. The networks were trained applying Resilient Propagation. Prior to training, all weights were randomly initialized in the range from -0.1 to 0.1. Since the training converged faster for turnwise classification, turnwise training was aborted after 10 epochs, whereas the training procedure for framewise classification was aborted after 250 epochs.

Before mapping the (B)LSTM predictions o_t onto quadrants, they were smoothed using a first order low-pass filter to obtain the filtered predictions o_t^s:

$$o_t^s = \alpha o_{t-1}^s + (1 - \alpha) \cdot o_t. \tag{4.1}$$

An α of 0.99 was used for time-continuous emotion recognition and an α of 0.7 was used for turn-based recognition. Both values were optimized on the training set.

Alternatively to Regression-LSTMs, Support Vector Regression (SVR) was performed for comparison [96, 269, 276]. The SVR used a polynomial kernel function of degree 1 and Sequential Minimal Optimization. The discriminatively trained LSTM networks were compared to Support Vector Machines instead of SVR. Since SVR and SVM do not model contextual information, only turnwise classification was evaluated in this case. In order to determine the gain of Long Short-Term Memory modeling, also conventional RNN classification were evaluated for comparison. The RNNs were trained in the same way as the LSTM networks, however, the network consisted of 50 hidden neurons instead of the 50 one-cell LSTM memory blocks.

Furthermore, inter-labeler consistency was evaluated as an upper benchmark for automatic emotion recognition. To obtain an impression of human emotion prediction, the annotations of one labeler were compared to the mean of the annotations of the remaining three labelers. This was done for all of the four labelers so that eventually the average inter-labeler consistency could be determined. As a further evaluation of inter-labeler agreement, Table 4.4 shows the kappa values for the four different annotators. Since each of the kappa values is larger than 0.4, the labeler agreement can be characterized as sufficiently high.

Table 4.5 shows the recognition result for the assignment of quadrants using the regression method and the discriminative technique, respectively. Results for the

Table 4.4: Kappa values for the four different annotators in the SAL database (turnwise quadrant labeling); ILA: inter-labeler agreement.

κ	**1**	**2**	**3**	**4**
ILA	0.68	0.67	0.67	0.60
1		0.49	0.48	0.46
2			0.48	0.45
3				0.52

five-class task which also considers a 'neutral' state (see Figure 4.3) can be seen in Table 4.6, and Table 4.7 contains the results for separate classification of the degree of arousal and valence (i. e., positive vs. negative arousal and valence, respectively). Again, the F1-measure was applied as main performance measure. Compared to emotion recognition on prototypical speech turns (as in [223], for example), the overall performance is significantly lower. Yet, the accuracies are in the order of magnitude that is typical for real-life experiments, attempting to classify natural, non-prototypical, and ambiguous emotional speech turns [219].

A rating of the prediction quality can be obtained when comparing the best result in Table 4.5 (framewise BLSTM classification using acoustic and linguistic features) with the prediction performance of a human labeler (*lab*, *frame* in Table 4.5): When comparing the annotation of a single labeler to the mean of the annotations of the remaining three labelers, the obtained average F1-measure (57.4 %) is only 7 % higher than the F1-measure of the best classifier (50.4 %). This reflects the ambiguity of perceived emotion and the resulting low degree of inter-labeler agreement. A further reason for the low annotator F1-measure is that a high amount of utterances are near the class borders (see Figure 4.3). Consequently, those speech turns are hard to assign, even for human annotators.

The best F1-measure for valence (72.2 %) is notably below the average 'performance' or consensus of a human labeler (85.7%). However, the best recognition result for arousal (68.9 %) is only 2.2 % below the inter-human labeling consistency (71.1 %). For the five-class task, the performance gap between the best classifier and human labelers is 8.6 % (see Table 4.6).

In what follows, we will analyze the results in Tables 4.5 - 4.7 with respect to six different aspects: the number of emotion classes, the difference between regression and discriminative training, the gain of LSTM context modeling, the benefit of including bidirectional context, the difference between turnwise and framewise classification, and the integration of linguistic features.

- *Four quadrants vs. five classes:* The best F1-measure for quadrant classification can be obtained when using a discriminative BLSTM for turnwise prediction with acoustic features (51.3 %, see Table 4.5). However, additionally modeling the 'neutral' state can lead to a comparable prediction performance

Table 4.5: Regression and discriminative (B)LSTM and RNN performance, SVR/SVM performance, and average labeler (*lab*) consistency for **quadrant classification** using turnwise or framewise prediction with acoustic (A) or acoustic-linguistic (A+L) features: accuracy (acc.), unweighted recall (rec.), unweighted precision (prec.), and F1-measure (F1) in [%].

			Regression				**Discriminative**			
model	**unit**	**features**	**acc.**	**rec.**	**prec.**	**F1**	**acc.**	**rec.**	**prec.**	**F1**
			quadrants							
BLSTM	turn	A	37.1	34.9	35.5	35.2	49.3	51.3	51.2	**51.3**
BLSTM	turn	A+L	41.0	36.9	37.8	37.3	47.6	48.6	46.8	47.7
BLSTM	frame	A	41.7	44.8	42.0	43.3	42.5	43.9	41.3	42.5
BLSTM	frame	A+L	48.2	51.6	49.3	**50.4**	39.0	37.4	37.1	37.2
LSTM	turn	A	37.3	37.9	35.4	36.6	48.6	47.4	48.2	47.8
LSTM	turn	A+L	38.6	38.4	39.8	39.7	44.9	49.1	48.3	48.7
LSTM	frame	A	31.2	33.4	37.2	35.2	37.4	38.0	38.1	38.1
LSTM	frame	A+L	34.2	30.7	37.9	33.9	32.0	37.8	32.6	35.3
RNN	turn	A	33.7	34.8	34.7	34.7	46.3	47.2	47.2	47.2
RNN	turn	A+L	37.1	35.5	36.7	36.1	45.9	46.5	45.8	46.1
RNN	frame	A	31.0	36.9	33.8	35.3	28.3	32.1	30.9	31.5
RNN	frame	A+L	28.2	31.7	34.8	33.2	22.1	28.2	27.3	27.7
SVR/SVM	turn	A	28.8	30.0	27.3	28.6	39.0	39.6	41.2	40.4
SVR/SVM	turn	A+L	33.3	32.2	30.4	31.3	37.8	38.5	36.7	37.6
lab	*turn*		*62.0*	*59.2*	*58.7*	*58.9*				
lab	*frame*		*59.2*	*58.3*	*56.7*	*57.4*				

(47.2 %, see Table 4.6). Interestingly, for the five-class task framewise regression prevails. Obviously, the higher number of class borders a discriminative classifier has to face in the five-class experiment downgrades performance significantly. As can be seen in Table 4.6, a BLSTM network modeling all five classes benefits from frame by frame modeling of the fineness of emotional dynamics via regression. Tables 4.8 and 4.9 show typical confusions when distinguishing four and five classes, respectively. In both cases, the best prediction quality can be obtained for quadrant II (*angry/anxious*). Table 4.9 demonstrates that, due to the non-prototypicality of emotions in the SAL corpus, almost all quadrants are most frequently confused with the neutral state. An impression of the prediction quality for more prototypical utterances (or utterances with emotions of higher intensity) can be obtained when masking the last column and the last line of Table 4.9: Quadrant-quadrant confusions obviously occur less frequent than quadrant-neutral confusions. Another interesting aspect is the effect of emotional intensity – and thus indirectly prototypicality – of the test set on the obtained recognition performance: When using the Regression-BLSTM for framewise prediction with acoustic and linguistic features (trained on *all* training data and characterized by the five-class confusion matrix in Table 4.9) and evaluating only those utterances that are *not* annotated as 'neutral', the

Table 4.6: Regression and discriminative (B)LSTM and RNN performance, SVR/SVM performance, and average labeler (*lab*) consistency for the **quadrant/neutral five-class task** using turnwise or framewise prediction with acoustic (A) or acoustic-linguistic (A+L) features: accuracy (acc.), unweighted recall (rec.), unweighted precision (prec.), and F1-measure (F1) in [%].

			Regression				**Discriminative**			
model	**unit**	**features**	**acc.**	**rec.**	**prec.**	**F1**	**acc.**	**rec.**	**prec.**	**F1**
			quadrants + neutral							
BLSTM	turn	A	37.9	34.1	38.6	36.2	39.8	40.1	38.4	39.2
BLSTM	turn	A+L	40.9	30.6	39.5	34.5	41.9	41.8	41.7	**41.7**
BLSTM	frame	A	34.6	39.3	34.3	36.6	28.0	25.3	29.5	27.2
BLSTM	frame	A+L	44.2	49.4	45.2	**47.2**	29.0	32.3	25.8	28.7
LSTM	turn	A	36.0	35.1	32.5	33.7	40.0	38.7	36.0	37.3
LSTM	turn	A+L	39.0	30.0	35.5	32.5	41.9	41.5	37.1	39.2
LSTM	frame	A	29.0	28.3	32.5	30.3	27.8	28.6	29.6	29.1
LSTM	frame	A+L	33.2	30.4	30.3	30.4	30.4	30.0	24.7	27.1
RNN	turn	A	35.1	30.9	33.2	32.0	38.0	39.8	35.4	37.5
RNN	turn	A+L	36.8	30.8	34.4	32.5	39.0	41.6	37.1	39.2
RNN	frame	A	35.6	21.1	41.4	27.9	28.7	24.3	25.0	24.6
RNN	frame	A+L	36.8	20.5	41.0	27.4	27.0	25.6	26.4	26.0
SVR/SVM	turn	A	32.8	25.5	24.9	25.2	34.8	35.8	35.2	35.5
SVR/SVM	turn	A+L	32.0	25.2	24.9	25.0	34.8	35.9	35.0	35.4
lab	*turn*		*56.8*	*55.1*	*53.7*	*54.3*				
lab	*frame*		*56.3*	*56.9*	*54.9*	*55.8*				

resulting quadrant prediction F1-measure is 58.2 %. On the other hand, when evaluating only those turns that are annotated as 'neutral', the F1-measure for quadrant prediction is as low as 34.3 %. For very 'intense' test utterances that are labeled as having an absolute value of arousal and valence that is higher than 0.5, the obtained quadrant prediction F1-measure is 85.1 %.

- *Regression vs. discriminative training:* For almost every experimental setting we can observe that discriminative training prevails for turnwise recognition while regression prevails for framewise recognition. Complete turns that are characterized by statistical functionals of features can be distinguished better with a discriminative technique. On the other hand, when predicting a class frame by frame the network fails to model 'label transitions' when discriminatively trained on the discrete labels. For framewise prediction, modeling the smooth progression of valence and arousal is necessary before mapping the output activations to quadrants.

- *LSTM context modeling vs. RNN and SVM:* For both, framewise and turnwise prediction, the LSTM architecture outperforms a conventional RNN in most cases. The major reason for this is the *vanishing gradient problem* (see Section 2.3.9) which limits the amount of context a recurrent neural network can access.

Table 4.7: Regression and discriminative (B)LSTM and RNN performance, SVR/SVM performance, and average labeler (*lab*) consistency for **classification of valence and arousal** (high vs. low) using turnwise or framewise prediction with acoustic (A) or acoustic-linguistic (A+L) features: accuracy (acc.), unweighted recall (rec.), unweighted precision (prec.), and F1-measure (F1) in [%].

			Regression				**Discriminative**			
model	**unit**	**features**	**acc.**	**rec.**	**prec.**	**F1**	**acc.**	**rec.**	**prec.**	**F1**
					arousal					
BLSTM	turn	A	64.8	65.0	64.9	64.9	68.3	68.9	68.8	**68.9**
BLSTM	turn	A+L	64.1	64.3	64.1	64.2	66.4	66.5	66.4	66.4
BLSTM	frame	A	64.0	64.1	64.1	64.1	62.8	63.6	64.0	63.8
BLSTM	frame	A+L	65.7	65.7	65.6	**65.6**	58.0	57.9	57.8	57.9
LSTM	turn	A	59.8	60.9	61.3	61.1	63.4	64.8	65.6	65.2
LSTM	turn	A+L	60.2	60.7	60.7	60.7	65.3	66.2	66.5	66.4
LSTM	frame	A	56.4	57.2	57.4	57.3	50.0	50.8	50.8	50.8
LSTM	frame	A+L	59.1	59.9	60.1	60.0	56.3	56.8	56.9	56.9
RNN	turn	A	54.6	55.1	55.2	55.2	61.7	63.0	63.8	63.4
RNN	turn	A+L	55.6	56.4	56.5	56.5	61.5	62.9	63.7	63.3
RNN	frame	A	53.4	55.1	56.4	55.7	50.6	52.7	53.8	53.3
RNN	frame	A+L	49.3	49.4	49.4	49.4	54.4	55.2	55.4	55.3
SVR/SVM	turn	A	53.8	53.3	53.3	53.3	55.8	56.7	56.8	56.8
SVR/SVM	turn	A+L	55.5	55.2	55.8	55.2	54.4	55.2	55.3	55.3
lab	*turn*		*68.6*	*70.6*	*71.6*	*71.1*				
lab	*frame*		*67.7*	*69.4*	*70.1*	*69.8*				
					valence					
BLSTM	turn	A	56.5	58.0	58.3	58.1	63.7	64.6	64.7	64.7
BLSTM	turn	A+L	60.0	61.1	61.4	61.3	71.2	71.8	71.7	**71.7**
BLSTM	frame	A	65.8	64.0	64.7	64.3	63.8	65.1	64.8	65.0
BLSTM	frame	A+L	72.8	72.2	72.1	**72.2**	55.0	58.4	59.7	59.0
LSTM	turn	A	61.0	62.5	62.9	62.7	56.4	59.4	63.4	61.3
LSTM	turn	A+L	58.8	60.3	60.9	60.6	66.8	68.5	70.1	69.3
LSTM	frame	A	55.9	57.4	57.4	57.4	65.3	66.3	65.9	66.1
LSTM	frame	A+L	63.6	57.7	67.3	62.1	58.3	56.1	56.6	56.4
RNN	turn	A	58.8	60.3	60.8	60.5	67.5	67.9	67.8	67.9
RNN	turn	A+L	62.9	64.2	64.8	64.5	69.5	70.5	70.6	70.5
RNN	frame	A	60.9	63.6	64.3	63.9	57.5	60.3	61.0	60.6
RNN	frame	A+L	57.5	62.0	66.0	63.9	64.2	64.6	64.2	64.4
SVR/SVM	turn	A	53.1	55.0	55.6	55.3	61.4	63.5	65.7	64.6
SVR/SVM	turn	A+L	56.0	57.5	58.0	57.8	59.3	61.4	62.9	62.1
lab	*turn*		*88.6*	*88.4*	*88.6*	*88.6*				
lab	*frame*		*86.0*	*85.8*	*85.6*	*85.7*				

Using no contextual information at all leads to comparatively low performance as can be seen in the SVR and SVM experiments.

- *Unidirectional vs. bidirectional context:* Independent of the classification task, bidirectional context mostly prevails over unidirectional context. Both, regres-

Table 4.8: Confusion matrix for the best quadrant classification setting (discriminative BLSTM for turnwise prediction with acoustic features only); rows: ground truth; columns: predictions (white to black represents 0-100 %).

%	I	II	III	IV
I	54	25	12	9
II	21	67	9	3
III	27	22	47	4
IV	31	21	9	39

Table 4.9: Confusion matrix for the best 'quadrants + neutral' (N) classification setting (regression BLSTM for framewise prediction with acoustic and linguistic features); rows: ground truth; columns: predictions (white to black represents 0-100 %).

%	I	II	III	IV	N
I	40	8	3	25	24
II	9	80	1	2	8
III	1	14	48	12	25
IV	13	4	6	40	37
N	11	16	10	22	41

sion and discriminative BLSTM networks outperform all other models (LSTM, RNN, SVR, and SVM) for the discrimination of five, four, and two classes (numbers in bold face in Tables 4.5 - 4.7).

- *Turnwise vs. framewise classification:* As already mentioned, turnwise prediction can successfully be combined with discriminative learning, while framewise emotion recognition is rather suited for predictors based on regression. For both strategies, modeling contextual information is essential. When additionally modeling 'neutrality', the best result can be obtained with framewise prediction (see Table 4.6). Note that the amount of contextual information a BLSTM network models is a lot more flexible when framewise prediction is applied, since the temporal granularity is higher than it is for turnwise recognition. This can be seen as the major reason why framewise recognition outperforms turnwise prediction if Regression-BLSTM networks are used.

- *Acoustic features vs. combined acoustic and linguistic features:* When inspecting Table 4.5, one can assert that the Regression-LSTM seems to profit more from the inclusion of linguistic features. In some cases the quadrant prediction performance of the discriminative classifier is even degraded when adding

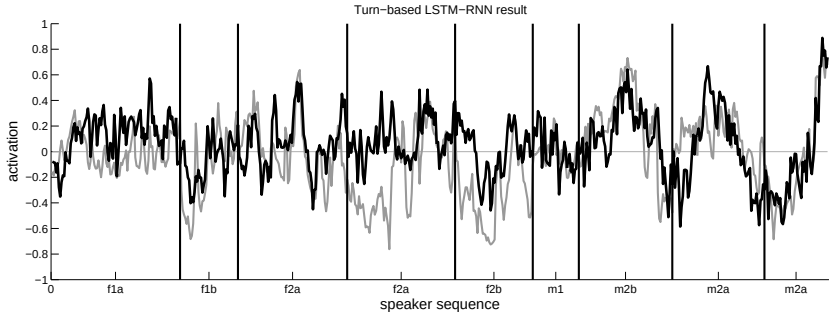


Figure 4.4: Prediction of arousal (black) using a Regression-LSTM and ground truth (grey) over all turns of the test set (only acoustic features used).

keyword features. Obviously, the presence of single keywords is not discriminative enough in this case. Linguistic features are rather suited for modeling tendencies within a continuous scale for valence and arousal. When modeling 'neutrality' as a fifth class, also the *discriminative* BLSTM profits from linguistic features (while this is not the case for the discriminative *four-class* task). This supports the finding that a performance gain through keyword features presumes a certain level of granularity of the prediction targets.

As an example for emotion recognition using regression, Figure 4.4 shows the turnwise arousal predictions of a Regression-LSTM before the output activations are mapped onto quadrants. Prediction and ground truth are correlated with a correlation coefficient of 0.56, leading to an F1-measure of 61.1 % (see Table 4.7) when distinguishing positive and negative arousal for every speech turn.

4.1.3 Acoustic-Linguistic Recognition of Interest

Detecting whether a user is interested or disinterested can be relevant for many applications of human-computer interaction, including sales and advertisement systems, virtual guides, or conversational agents. Recently investigated use-cases for automatic interest recognition comprise topic switching in infotainment or customer service systems [228], meeting analysis, and tutoring systems [164]. In the light of this growing amount of research on interest-related affective computing, the organizers of the Interspeech 2010 Paralinguistic Challenge [220] defined an interest recognition task with unified system training and test conditions in order to make the recognition approaches developed by different researchers easily comparable. In the *Affect Sub-Challenge*, the task is to automatically predict a user's level of interest from the speech signal applying a pre-defined acoustic feature set and (optionally) linguistic information. Participants used the Audiovisual Interest Corpus recorded at the Technische Universität München ("TUM AVIC") [228]. It contains highly

spontaneous speech from face-to-face commercial presentations and reflects the conditions a real-life interest recognition system has to face. The challenge task was to predict a speaker's level of interest by suited regression techniques.

This section shows how contextual information can be exploited for enhanced acoustic-linguistic interest recognition by employing the LSTM neural network architecture. Similar to the experiments on emotion recognition discussed in Section 4.1.2, a bidirectional LSTM network is applied to model how the user's interest level evolves over time. Yet, in contrast to LSTM-based emotion recognition systems which contain one hidden layer [294], we investigate *Bottleneck*-BLSTM networks by using three hidden layers with a narrow middle layer (the 'bottleneck') [299]. As outlined in Section 3.2.4, bottleneck networks can be incorporated into systems for automatic speech recognition where they can be applied for feature dimensionality reduction within Tandem systems [94, 296], i. e., speech recognizers that use RNNs or MLPs to generate features. For the interest recognition system proposed in this section, the bottleneck principle is combined with the BLSTM technique to generate a compact feature representation within the BLSTM network. In addition to acoustic features, the Bottleneck-BLSTM network processes linguistic information obtained from an ASR module.

Database

The following experiments are based on the TUM AVIC corpus [228] which has also been used for the Affect Sub-Challenge of the Interspeech 2010 Paralinguistic Challenge [220]. In the scenario setup, an experimenter and a subject are sitting on opposite sides of a desk. The experimenter plays the role of a product presenter and leads the subject through a commercial (car) presentation. The subject's role is to listen to explanations and topic presentations of the experimenter, ask several questions of his/her interest, and actively interact with the experimenter considering his/her interest in the addressed topics.

The 'level of interest' is annotated for every turn using five levels of interest from disinterest to curiosity (LOI -2, -1, 0, 1, 2). Further, the spoken content as well as non-linguistic vocalizations have been transcribed. For the Interspeech 2010 Paralinguistic Challenge, the ground truth has been established by shifting to a continuous scale obtained by averaging the single annotator LOI. In accordance with the scaling applied in other corpora, the original LOI scale reaching from -2 to +2 is mapped to the interval from -1 to 1.

The speech data from the 21 speakers (3 880 turns) were split into speaker independent training, development, and test sets. The training set consists of 1 512 turns and 51.7 minutes of speech, respectively, and comprises four female and four male speakers, while the development set contains 1 161 turns, corresponding to 43.1 minutes of speech (three female and three male speakers). The test set includes 1 207 turns and 42.7 minutes of speech, respectively (three female and four male speakers).

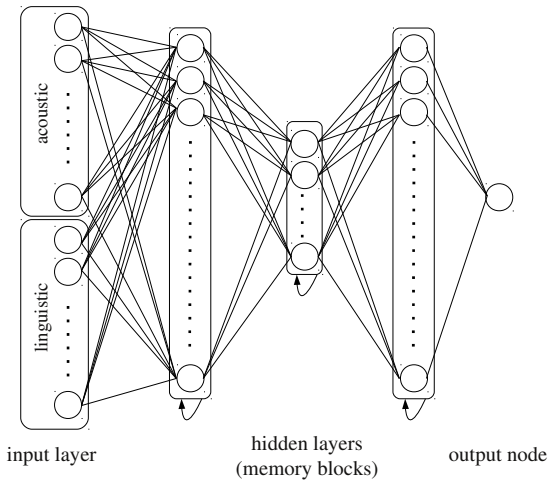


Figure 4.5: Structure of the bottleneck networks used for interest recognition.

More details on the TUM AVIC corpus can be found in [220].

Bottleneck-BLSTM Nets

Building on recent successes of LSTM-based affective computing and speech recognition [281, 289, 294], Long Short-Term Memory RNNs were applied for context-sensitive interest recognition as well as within the ASR engine for linguistic feature generation. As in the emotion recognition system depicted in Figure 4.2, a feature extractor provides MFCC features to a BLSTM network which computes a phoneme prediction. Together with the MFCC features, those phoneme predictions are decoded by the multi-stream HMM introduced in Section 3.2.2, which outputs linguistic features. Both, linguistic features and acoustic features are processed by a second BLSTM network which infers the final level of interest prediction.

In the following, we consider a combination of the LSTM principle and bottleneck network architectures. As explained in Section 3.2.4, bottleneck MLPs or RNNs consist of (at least) three hidden layers with a narrow layer in the middle. In ASR, bottleneck systems process features that are obtained from the linear outputs of the neurons in the bottleneck layer, i. e., only the first two hidden layers are involved during feature extraction. This offers the advantage that by choosing the size of the bottleneck layer, the dimensionality of the feature vector can be defined. Thus, the network implicitly performs dimensionality reduction and generates decorrelated and compressed features – independent of the number of training targets and without the need for explicit decorrelation and dimensionality reduction techniques such as PCA. Unlike static techniques based on PCA (or MLPs), combining LSTM and

bottleneck architectures enables *context-sensitive* feature compression.

For interest recognition, five-layer Bottleneck-LSTMs as shown in Figure 4.5 are applied. The networks are composed of an input layer whose size corresponds to the dimensionality of the acoustic-linguistic feature vector, three hidden layers including the bottleneck layer in the middle, and an output layer consisting of one node whose activation indicates the estimated level of interest. Unlike in bottleneck ASR systems, where the third hidden layer is only used during network training and not during decoding / feature generation, the networks applied for interest recognition in this section employ *all* layers and thus perform dimensionality reduction and decorrelation *within* the network.

Acoustic and Linguistic Feature Extraction

The acoustic features applied in this section correspond to the baseline feature set of the Interspeech 2010 Paralinguistic Challenge [220]. Again, they are extracted via the real-time speech analysis toolbox openSMILE [73]. 1 582 acoustic features are obtained in total by systematic 'brute-force' feature generation in three steps: First, 38 low-level descriptors (see [220]) are extracted at 100 frames per second with varying window type and size (Hamming and 25 ms, respectively, for all but pitch which is extracted using a Gaussian window and a window size of 60 ms) and smoothed by simple moving average low-pass filtering with a window length of three frames. Next, their first order regression coefficients are added. Then, 21 statistical functionals are applied to each low-level feature stream in order to capture time-varying information in a fixed-length static feature vector for each instance in the database. Note that 16 zero-information features (e. g., minimum F0, which is always zero) are discarded. Finally, the two single features 'number of pitched segments' and turn duration are added.

For linguistic feature extraction, the multi-stream BLSTM-HMM ASR system detailed in Section 3.2.2 is applied. The main idea of this technique is to enable improved recognition accuracies by incorporating context-sensitive phoneme predictions generated by a BLSTM network into the speech decoding process (see also [281]).

Via early fusion, the linguistic information extracted by the multi-stream speech recognizer is fused with the supra-segmental acoustic features. To obtain linguistic feature vectors from the ASR output, a standard Bag of Words technique is employed – similar to the approach explained in Section 4.1.2. For each word in a segment, the term frequency is computed. Only words with a minimum term frequency of two throughout the training set are considered (152 words). A vector space representation of the word string is built from the word's term frequencies.

To reduce the size of the fused acoustic-linguistic feature space prior to subsequent dimensionality reduction and decorrelation within the bottleneck network, a cyclic Correlation-based Feature Subset Selection based on the TUM AVIC training

Table 4.10: Size of the hidden layers for networks with one hidden layer and bottleneck networks processing acoustic (A) or combined acoustic-linguistic (A+L) information.

		size of hidden layers	
classifier	**bottleneck**	**A**	**A+L**
BLSTM	yes	32-6-32	32-8-32
LSTM	yes	64-12-32	64-16-32
BRNN	yes	32-6-16	32-8-16
RNN	yes	64-12-16	64-16-16
BLSTM	no	32	32
LSTM	no	64	64
BRNN	no	16	16
RNN	no	32	32

set is conducted. As a result, 92 selected acoustic features are obtained and the combined acoustic-linguistic feature vectors are of size 123.

Experiments and Results

Various neural network architectures were evaluated with respect to their suitability for acoustic and acoustic-linguistic interest recognition: conventional recurrent neural networks, bidirectional recurrent neural networks, LSTM networks, and bidirectional LSTM networks. For each network type, architectures with one hidden layer (as used in Section 4.1.2) and bottleneck structures consisting of three hidden layers were considered. The number of memory blocks (or hidden nodes) per layer was optimized on the development set and can be seen in Table 4.10. For example, the Bottleneck-BLSTM processing acoustic and linguistic features applied 32 memory blocks in the first and third hidden layer and contained a bottleneck layer of size eight. Networks processing only acoustic features used slightly less memory blocks in the bottleneck layer (six for bidirectional networks). Note that simply increasing the number of hidden cells in networks consisting of one hidden layer or applying networks with an equal number of hidden cells (or memory blocks) in all three hidden layers led to lower performance on the development set than bottleneck architectures. The number of input nodes corresponds to the number of selected acoustic or combined acoustic-linguistic features. All memory blocks of the (B)LSTMs were composed of one memory cell. The networks had one (regression) output node whose activation represents the predicted level of interest.

For improved generalization, Gaussian noise was added to the inputs during training (standard deviation of 1.2). Note that all input features were z-normalized before being processed by the networks. Means and standard deviations for z-normalization were computed from the training set. The multi-stream ASR system was parametrized as in [281]. Both, the multi-stream acoustic models and a back-off

Table 4.11: Results for interest recognition as defined in the Affect Sub-Challenge [220]: cross correlation obtained for different network architectures when using either acoustic (A) or combined acoustic-linguistic (A+L) information with and without bottleneck structure; baseline results reported in [220] when applying unpruned REP-Trees with and without correlation-based feature selection (CFS); results reported in [116] and [79] when using SVM and GMM, respectively.

			cross correlation	
classifier	**CFS**	**bottleneck**	**A**	**A+L**
BLSTM	yes	yes	0.459	**0.504**
LSTM	yes	yes	0.454	0.479
BRNN	yes	yes	0.427	0.440
RNN	yes	yes	0.434	0.433
BLSTM	yes	no	0.442	0.475
LSTM	yes	no	0.431	0.459
BRNN	yes	no	0.406	0.438
RNN	yes	no	0.422	0.439
REP-Trees	yes	-	0.439	0.435
REP-Trees [220]	no	-	0.421	0.423
SVM [116]	no	-	-	0.428
GMM [79]	no	-	0.390	-

bigram language model were trained on the TUM AVIC training and development set (vocabulary size of 1.9 k).

Table 4.11 shows the results obtained on the Interspeech 2010 Paralinguistic Challenge (more precisely the *Affect Sub-Challenge*) when applying the different context-sensitive neural network architectures. In conformance with [220], the cross correlation (CC) between the ground truth level of interest and the predicted level of interest was chosen as evaluation criterion. Note that the mean linear error (MLE) is *not* reported, since the MLE strongly depends on the variance of the ground truth labels and is hardly suited for revealing the accuracy of the predictions. As an example, when evaluating a ('dummy') classifier that always predicts the mean of the training set ground truth labels, we obtain an MLE of 0.148 (which is only 0.002 below the MLE reported in [220]), while we get a CC of zero.

All results reflect the recognition performance on the TUM AVIC test set, when training the predictors on the training and development partition of the TUM AVIC corpus. Using only the training set did not lead to satisfying results since the neural network architectures require a comparatively large amount of training data for generalization. Incorporating linguistic information leads to higher cross correlations for all network architectures which is in line with results shown in Section 4.1.2. Furthermore, Bottleneck-(B)LSTM architectures consistently outperform networks with one hidden layer. The best performance can be obtained when applying Bottleneck-BLSTM networks processing both, acoustic and linguistic features (CC

of 0.504). Bidirectional LSTM modeling gives slightly better results than unidirectional LSTM, which indicates that also future information (if available) can be efficiently exploited for interest recognition. The performance difference between LSTM-based architectures and conventional RNN techniques reveals that the ability to model long-term temporal context is beneficial for the classification task.

For comparison, also the Paralinguistic Challenge baseline result (CC of 0.421, obtained with unpruned REP-Trees in Random-Sub-Space meta-learning [220]) is shown in Table 4.11. The REP-Trees approach profits from feature selection via CFS but cannot compete with the Bottleneck-BLSTM technique. Results obtained for BLSTM modeling are notably better than the highest cross correlation that has been reported for the Affect Sub-Challenge so far (CC of 0.428 using SVMs in combination with acoustic and linguistic information [116]) and prevail over the CC reported in [79] for GMMs.

4.1.4 Emotion Recognition in Reverberated Environments

As discussed in Section 4.1, past research on AER has mostly been restricted to prototypical, acted, and speaker dependent emotion recognition. The focus of today's research is on speaker independence and on affective state estimation from non-prototypical, spontaneous speech as it is needed for real-life applications [211]. Reflecting these challenging conditions, which typically lead to recognition accuracies that are lower than those reported for prototypical emotions, the Interspeech 2009 Emotion Challenge [219] has been organized to define unified system training and test conditions involving spontaneous emotion recognition during child-robot interaction. Yet, one simplification of the Emotion Challenge task that might not necessarily hold for real-life systems is the restriction to speech captured by close-talk microphones. The effect of speech signal distortions caused by reverberation or background noise has been largely neglected in the Emotion Challenge – and generally in the field of speech-based emotion recognition. Only a few studies address the topic of noise robust AER, e. g., [243]. The impact of reverberation on AER from acoustic cues has been investigated in [262].

In this section, research on affect recognition from reverberated speech [262] is extended to systems that apply both, acoustic and linguistic features obtained via an ASR module. We examine how different microphones and room acoustics affect the quality of the ASR output on the one hand, and the accuracy of combined acoustic-linguistic emotion recognition on the other hand. To this end, emotional child-robot interaction speech as contained in the FAU Aibo Emotion Corpus [236] is considered in combination with different artificial and real reverberation conditions. Furthermore, matched, mismatched, and multi-condition training are investigated to increase the robustness of the proposed recognition engine.

Database

The German FAU Aibo Emotion Corpus [236] with 8.9 hours of spontaneous, emotionally colored children's speech comprises recordings of 51 children at the age of 10 to 13 years from two different schools (see also Section 3.1.3, where the FAU Aibo Emotion Corpus is used for keyword spotting experiments). Speech was transmitted with a wireless head set (UT 14/20 TP SHURE UHF-series with microphone WH20TQG) and recorded with a DAT-recorder. The sampling rate of the signals is 48 kHz; quantization is 16 bit. The data is downsampled to 16 kHz.

As explained in Section 3.1.3, the children were given five different tasks where they had to direct Sony's dog-like robot Aibo to certain objects and through a given 'parcours'. The children were told that they could talk to Aibo the same way as to a real dog. However, Aibo was remote-controlled and followed a fixed, pre-determined course of actions, which was independent of what the child was actually saying. At certain positions Aibo disobeyed in order to elicit negative forms of emotions. The corpus is annotated by five human labelers on the word level using 11 emotion categories that have been chosen prior to the labeling process by iteratively inspecting the data. The units of analysis are not single words, but semantically and syntactically meaningful chunks (2.66 words per chunk on average, see [236]). Heuristic algorithms were used to map the decisions of the five human labelers on the word level onto a single emotion label for the whole chunk. The emotional states that can be observed in the corpus are rather non-prototypical, emotion-related states than 'pure' emotions. Mostly, they are characterized by low emotional intensity.

Acoustic and Linguistic Feature Extraction

A set of 384 segmental acoustic features suited for static chunk-level classification was extracted. These features exactly correspond to those used for the Interspeech 2009 Emotion Challenge baseline (Classifier Sub-Challenge) and include MFCCs, prosodic, and voice quality features (see [219]). Note that none of the Challenge participants could outperform the baseline features in the Feature Sub-Challenge [211].

To create linguistic features for early fusion with the chunk-level acoustic features, the chunk-level ASR results (i. e., reclassification of the training set, and recognition of the test set), were converted into a vector space representation by forming Bag of Words vectors counting term frequencies. The components of the BoW vectors represent all words occurring in the reclassification of the training set by the ASR engine. As a result, the BoW feature space differs among training conditions. The BoW size ranges from 198 (training on room microphone data) to 379 (multi-condition training) since the ground truth transcriptions available in the FAU Aibo Emotion Corpus were intentionally *not* used for building linguistic features, both to enforce

realism, and to adapt to typical ASR confusions in the varying acoustic conditions.

Two different ASR systems for linguistic feature generation were evaluated: a standard single-stream HMM system applying cross-word triphone acoustic models and the multi-stream BLSTM-HMM system introduced in Section 3.2.2.

Experiments and Results

Along the lines of the Interspeech 2009 Emotion Challenge [219], the complete corpus was used for the experiments reported in this section (i. e., not just chunks containing prototypical emotions). Yet, due to technical problems with the video camera recording the reverberated 'room microphone' data, only 17 076 of the 18 216 chunks could be used. Thus, the training set comprises 9 190 chunks and the test set consists of 7 886 chunks. The 2-class problem with the two main classes *negative valence* (NEG) and the default state *idle* (IDL, i. e. neutral) was considered. A summary of the challenge task and results is given in [211].

As the children of one school were used for training and the children of the other school for testing, the partitions feature speaker independence, which is needed in most real-life settings, but can have a considerable impact on classification accuracy. Furthermore, this partitioning provides realistic differences between the training and test data on the acoustic level due to the different room characteristics. Finally, it ensures that the classification process cannot adapt to socio-linguistic or other specific behavioral cues. Note that – as it is typical for realistic data – the two emotion classes are highly unbalanced (5 642 NEG-chunks vs. 11 434 IDL-chunks).

The data which was used for the 2009 Emotion Challenge was recorded with a close-talk microphone and will be called 'close-talk' (CT) in the following. Additionally, during creation of the FAU Aibo Emotion Corpus, the experiment was filmed with a video camera for documentary purposes. The child was not facing the microphone, and the camera was approximately 3 m away from the child. Thus, the audio channel of the videos is reverberated and contains background noises, e. g., the noise of Aibo's movements. While the recordings for the training set took place in a normal, rather reverberant class room, the recording room for the test set was a recreation room, equipped with curtains and carpets, i. e., with more favorable acoustic conditions. Thus, the data set provides realistic differences between training and test data on the acoustic level. This version will be called 'room microphone' (RM).

Another version [150] of the corpus was created using *artificial* reverberation: The data of the close-talk version was convolved with 12 different impulse responses recorded in a different room using multiple speaker positions (four positions arranged equidistantly on one of three concentric circles with the radii 60, 120, and 240 cm) and alternating echo durations $T_{60} \in \{250\,\text{ms}, 400\,\text{ms}\}$ spanning 180°. The training and test set were evenly split in twelve parts, of which each was reverberated with a different impulse response, to enforce a roughly equal distribution of the impulse

responses among the training and test set instances. This version will be called 'close-talk reverberated' (CTRV).

The acoustic feature vectors processed by the ASR system consisted of cepstral mean normalized MFCC coefficients 1 to 12, log. energy, as well as first and second order delta coefficients. The framewise BLSTM phoneme predictor of the multi-stream system was trained on forced aligned (framewise) phoneme targets of the FAU Aibo Emotion Corpus training set. According to past studies [281], three hidden layers of size 56, 150, and 56 were chosen, to model 53 German phonemes as well as *silence*, *short pause*, and *non-verbal events*. All other parameters of the multi-stream ASR system, such as the stream weight of the BLSTM phoneme prediction feature stream, were configured as in [281]. The underlying HMM system applied phoneme models consisting of three emitting states (left-to-right HMMs) with eight Gaussian mixtures. Initial monophones HMMs were mapped to tied-state cross-word triphone models with shared state transition probabilities. The acoustic models and a back-off bigram language model were trained on the training set of the FAU Aibo Emotion Corpus.

Table 4.12 shows the word accuracies when applying standard triphone acoustic models and the multi-stream BLSTM-HMM approach, respectively. Four different ASR training conditions were considered: training on data recorded by the close-talk microphone (CT), artificially reverberated data (CTRV), data recorded by the room microphone (RM), and all data (CT + CTRV + RM). Accuracies are consistently higher for the multi-stream model with performance gains of up to 16 % (absolute) when training on RM data and testing on CTRV data. This indicates that BLSTM context modeling within the multi-stream technique leads to higher robustness with respect to different reverberation conditions. However, also for 'friendly' scenarios, e. g, training and testing on data recorded by close-talk microphones, the multi-stream model prevails over standard HMMs (word accuracy of 87.03 % vs. 85.28 %). These accuracies are notably higher than those reported in [212], for example. As expected, matched condition training performs best, with the exception that RM data is best recognized using models trained on data reflecting all three acoustic conditions. Generally, multi-condition training leads to high accuracies for all test conditions and achieves the best average ASR performance (WA of 76.6 % for the multi-stream model).

To investigate the impact of ASR performance on emotion recognition, linguistic and joint acoustic-linguistic analysis by early feature-level fusion was evaluated, using the SimpleLogistic algorithm [131] implemented in the Weka toolkit [102]. It is based on boosting of one-dimensional regression functions, implicitly performing a feature relevance analysis and selection. This technique seems to be particularly suited for feature-level fusion dealing with varying reliability of features according to acoustic conditions. The number of boosting iterations was cross-validated on the training set, using the default parameters in the Weka toolkit for straightforward reproducibility. Since the class distribution in the training set of the FAU

Table 4.12: ASR word accuracies for different training and test conditions. The best result per test condition is highlighted.

	test condition							
word accuracy [%]	**single-stream HMM**				**multi-stream BLSTM-HMM**			
training condition	CT	CTRV	RM	**mean**	CT	CTRV	RM	**mean**
CT	**85.28**	79.21	28.66	64.38	**87.03**	80.48	43.97	70.49
CTRV	82.86	**82.03**	48.82	71.24	85.33	**84.52**	56.83	75.56
RM	13.35	33.78	53.00	33.38	25.77	49.79	57.82	44.46
CT + CTRV + RM	83.05	81.11	**61.21**	**75.12**	83.76	82.13	**63.90**	**76.60**

Aibo Emotion Corpus is heavily unbalanced, the Synthetic Minority Oversampling Technique (SMOTE) was applied. Unlike the AER engines presented in Sections 4.1.1 to 4.1.3, the AER strategy investigated in this section does not use long-range temporal context modeling via LSTM networks, as the pre-determined progression of obedient and disobedient actions performed by the Aibo robot tends to lead to easily predictable dynamics in the succession of the child's emotion that could be learned by the LSTM network. This would mean a very database-specific simplification that does not carry over to other child-robot interaction scenarios. Thus, every training and test instance was processed in isolation and a classifier not modeling 'emotional history' was applied.

The left half of Table 4.13 presents the unweighted accuracies (UA) for emotion recognition by BoW linguistic features obtained from single-stream HMM ASR, both with and without acoustic features. For reference, also the results by acoustic features only are shown. For CT, CTRV, and multi-condition training, these are similar to the ones obtained by SVMs in [262]; for RM training, however, the SimpleLogistic classifier yields a significant ($p < 0.005$) performance gain over SVM in the CT (66.32 vs. 61.61 % UA) and RM (64.96 vs. 62.72 % UA) test cases. Best average performance is achieved by multi-condition training (64.95 % UA).

Furthermore, linguistic features on their own result in a remarkable performance: When using ASR features from CT data for training, 64.76 % and 64.92 % UA are achieved in the CT and CTRV test conditions, respectively. Overall, a strong correlation with the word accuracies from Table 4.12 can be seen, with multi-condition training showing best average performance (62.22 % UA) once more.

Finally, by fusion of acoustic and linguistic information a significant ($p < 0.005$) performance improvement over acoustic features, from 67.90 % to 70.08 % UA is observed for matched condition CT training and testing. While for RM testing, the clean acoustic-linguistic classifier prevails over both pure acoustic and linguistic analysis (60.94 % UA vs. 59.83 % and 54.67 %, respectively), this is not the case for CTRV testing, where a drop in performance (59.27 % vs. 64.92 % UA) compared to linguistic features is observed, which is arguably caused by the poor performance of acoustic features in that particular setup (53.99 % UA). Remarkably, on average over

Table 4.13: Unweighted accuracies (UA) for acoustic, linguistic, and combined acoustic-linguistic classification of the test set by feature-level fusion with BoW vectors. The best result per test condition is highlighted.

UA [%]	**test condition**							
training condition	CT	CTRV	RM	**mean**	CT	CTRV	RM	**mean**
			acoustic					
CT	67.90	53.99	59.83	60.57				
CTRV	59.97	67.22	60.27	62.48				
RM	66.32	63.03	64.96	64.77				
CT + CTRV + RM	68.20	66.24	60.40	64.95				
	single-stream HMM				**multi-stream BLSTM-HMM**			
			linguistic					
CT	64.76	64.92	54.67	61.45	65.21	64.53	56.54	62.10
CTRV	63.59	63.15	58.05	61.59	63.90	63.58	58.74	62.07
RM	55.47	58.06	60.20	57.91	56.44	59.96	60.64	59.01
CT + CTRV + RM	63.38	62.99	60.29	62.22	64.07	63.28	60.44	62.60
			acoustic + linguistic					
CT	**70.08**	59.27	60.94	63.43	**70.32**	59.34	62.19	63.95
CTRV	60.28	**68.55**	62.44	63.76	60.34	**68.61**	63.05	64.00
RM	65.86	63.58	**65.41**	64.95	65.80	64.05	**65.43**	65.09
CT + CTRV + RM	68.92	67.96	62.48	**66.46**	69.16	67.84	62.96	**66.65**

all test conditions, fused acoustic-linguistic analysis using multi-condition training (66.46 % UA) considerably outperforms linguistic (62.22 %) and acoustic analysis (64.95 % UA). The best performance on RM, i. e., realistically reverberated, data is obtained by fused acoustic-linguistic analysis trained on RM (65.41 % UA) – note that this is not matched condition training in a strict sense, since the training and test set were recorded in different acoustic settings. This suggests that whenever the acoustic conditions that the emotion classifier has to face are known to a certain degree (corresponding to CT and CTRV testing), multi-condition training is most promising; for unknown conditions (RM testing), training on realistically reverberated data is to be preferred, even if that data does not exactly match the acoustic conditions to be faced.

The right half of Table 4.13 shows the results for linguistic and acoustic-linguistic AER when applying the multi-stream BLSTM-HMM speech recognizer for linguistic feature generation. For almost all training and test conditions, we observe higher accuracies than for the recognition engine using conventional HMM ASR. Trends are similar to those for the single-stream HMM, i. e., matched condition training performs best while multi-condition training leads to the best average accuracy.

4.2 Audio-Visual Affect Recognition

Humans express and perceive emotion through the complex interplay of multiple modalities [126, 157]. Thus, considering multiple modalities when trying to automatically assess the emotional state of a user can give a more complete description of the expressed emotion and generally tends to lead to more accurate results than unimodal techniques [160]. Since most of today's computer systems are equipped with microphones and cameras, audio and video are the most important non-obtrusive modalities based on which affect recognition can be performed. Audio and video channels can provide complementary information and tend to improve recognition performance if they are used in a combined multimodal setup [228]. This led to a large number of studies investigating audiovisual non-verbal behavior analysis (e. g., [221]).

Similar to Section 4.1, which concentrates on purely speech-based approaches towards affect recognition, this section shows how multimodal emotion recognition can be improved via temporal context modeling applying Long Short-Term Memory networks. Now, we focus on *audio-visual* AER systems by integrating information from the video channel. We investigate both, feature-level (Section 4.2.1) and decision-level fusion (Section 4.2.3) of audio and video. In Section 4.2.1, various classification approaches such as Support Vector Machines, Hidden Markov Models, and Long Short-Term Memory networks are compared and evaluated with respect to their performance in assessing human affect based on speech and facial marker information [161, 289]. Next, in Section 4.2.2, we analyze the so-called *sequential Jacobian* of trained BLSTM networks for emotion recognition in order to determine the *amount* of context that is modeled by BLSTM networks used for context-sensitive AER in Section 4.2.1 [290]. Finally, Section 4.2.3 shows how acoustic, linguistic, and facial movement features can be exploited to recognize affect in an audio-visual LSTM-based classification framework [284].

4.2.1 Emotion Recognition from Speech and Facial Marker Information

This section describes a multimodal emotion recognition framework that merges audio-visual information at the feature level and uses LSTM networks to model long-range temporal dependencies [289]. Again, we focus on the recognition of dimensional emotional labels, valence and arousal, instead of categorical emotional tags, such as 'anger' or 'happiness'. The applied database for system training and testing also includes non-prototypical data; meaning utterances that are labeled differently by different annotators and may not have a categorical label. We classify a variety of emotional manifestations, which may include ambiguous emotions, subtle emotions, or mixtures of emotions. As discussed in Section 4.1, this allows for a more realistic AER performance assessment, since a real-life system has to classify

all data that is recorded. The acoustic and facial feature extraction applied in this section is based on the technique introduced in [160]. Yet, in contrast to [160], the considered approach does not use phoneme-dependent models or viseme information and thus does not rely on the correct phoneme transcription.

In the following experiments, a large multimodal and multisubject acted database [31] is used. It was collected so as to contain emotional manifestations that are non-prototypical and resemble as much as possible real-life emotional expression. In addition to classifying the degree of valence and arousal separately, we also investigate the modeling of clusters in the emotional space (as in Section 4.1.1). We compare the recognition performance of bidirectional LSTM networks to a conventional SVM approach and to fully-connected HMMs. Short-term context is incorporated into the HMM framework using a first-order 'language model', based on emotional state transition probabilities as observed in the training set.

Database

The Interactive Emotional Dyadic Motion Capture (IEMOCAP) database [31] contains approximately 12 hours of audio-visual data from five mixed gender pairs of actors, male and female (ten subjects in total). It includes detailed face information obtained from motion capture as well as video and audio of each session. Two techniques of actor training were used; scripts and improvisation of hypothetical scenarios. The goal was to elicit emotional displays that resemble natural emotional expression. Dyadic sessions of approximately five minute length were recorded and were later manually segmented into utterances. Each utterance was annotated into nine categorical (such as anger, happiness, or neutrality) as well as dimensional tags (valence, arousal, dominance), by multiple human annotators. In contrast to the SAL database (see Section 4.1.2), the dimensional tags in the IEMOCAP database are not in the range from -1 to 1 but take integer values that range from one to five. The dimensional tag of an utterance is the average of the tags given by at least two annotators. In the following, we focus on the classification of valence and arousal, so that all the available data can be used – even utterances for which there was no inter-annotator agreement and no categorical label, respectively. Such data are a relatively large portion of the database (approximately 17 % of the total utterances).

Feature Extraction and Selection

The IEMOCAP database comprises detailed facial marker information, as illustrated in Figure 4.6. Face markers are normalized for head rotation and translation and the marker at the tip of the nose is defined as the local coordinate center of each frame. The (x,y,z) coordinates from 46 facial markers are used. In order to obtain a low-dimensional representation of the facial marker information, principal feature analysis (PFA) [148] is applied. This method performs principal component analy-

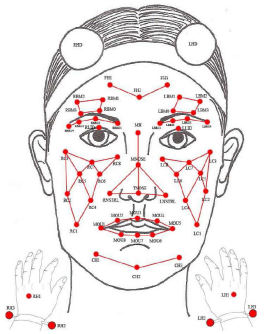


Figure 4.6: Facial marker positions.

sis as a first step and selects features (here marker coordinates) so as to minimize the correlations between them. 30 features are selected because the PCA transformation explains more than 95 % of the total variability, and the first derivatives are appended, resulting in a 60-dimensional representation. In addition, the facial features are normalized per speaker in order to smooth out individual facial characteristics that are unrelated to emotion. The speaker normalization approach consists of finding a mapping from the individual average face to the general average face. This is achieved by shifting the mean value of each marker coordinate of each subject to the mean value of that marker coordinate across all subjects. The feature selection and normalization framework is described in detail in [159].

From the speech signal, a variety of low-level features are extracted: 12 MFCC coefficients, 27 Mel-Frequency Band coefficients, pitch, and energy. In addition, their first derivatives are computed. All the audio features are normalized using mean and variance standardization (the statistics are computed from the corresponding training set). The audio and visual features are extracted at the same framerate of 25 ms, with a window size of 50 ms. Since the evaluation experiments are organized in a cyclic leave-one-speaker-out (LOSO) cross validation, all the normalization constants for the audio and video features, as well as the PCA transforms, are computed in a subject-independent way from the training set of each fold.

For LSTM and SVM classification, a set of utterance-level statistical functionals computed from the low-level acoustic and visual features are used. These functionals include means, standard deviations, linear and quadratic regression parameters (slope, offset, linear/quadratic approximation error), maximum and minimum positions, skewness, kurtosis, quartiles, inter-quartile ranges, and percentiles. In order to reduce the size of the resulting feature space, a cyclic CFS feature selection is performed, using the training set of each fold. This results in an automatic selection of between 66 and 224 features, depending on the classification task and the fold. For the valence classification task, on average 84 % of the selected features are facial features, whereas for classification of the degree of arousal, only 44 % of the features

Table 4.14: Distribution of the features selected via CFS for the classification of valence and arousal as well as for the discrimination of 3, 4, and 5 clusters in emotional space.

feature group	valence	arousal	3 clusters	4 clusters	5 clusters
pitch	5 %	4 %	3 %	4 %	3 %
energy	0 %	1 %	1 %	1 %	1 %
MFCC	4 %	21 %	11 %	11 %	10 %
MFB	7 %	30 %	18 %	19 %	21 %
lower face	63 %	32 %	50 %	49 %	48 %
upper face	21 %	12 %	17 %	16 %	17 %

selected via CFS are facial features. This underlines the fact that visual features tend to be well-suited for determining valence while acoustic features rather reveal the degree of arousal. For a detailed analysis of the selected features see Table 4.14.

Experiments and Results

The valence and arousal annotations in the IEMOCAP database range from one to five and can be non-integer, since the decisions of two evaluators are averaged for each utterance label. In the following experiments, we examine the classification of three levels of valence (negative, neutral, and positive, corresponding to ratings {1,1.5,2}, {2.5,3,3.5}, and {4,4.5,5}, respectively) and arousal (low, medium, and high, again corresponding to {1,1.5,2}, {2.5,3,3.5}, and {4,4.5,5}). The class sizes are not balanced since medium values of labels are more common than extreme values. We also consider the joint classification of the emotional dimensions by building three, four, and five clusters in the valence-arousal space. As in Section 4.1.1, the cluster midpoints in the emotional space are determined by applying the k-means algorithm on the annotations of the respective training sets. The ground truth of every utterance is assigned to one of the clusters using the minimum Euclidean distance between its annotation and the cluster midpoints. The intuition for clustering the valence-arousal space is to build classifiers that provide richer and more complete emotional information, that can correspond to generic emotional tags. For example, as can be seen in Figure 4.7, the coordinates of the cluster midpoints are interpretable: When considering three clusters, the midpoints roughly correspond to the affective states 'angry', 'neutral/sad', and 'happy'. The average standard deviation of the cluster centroid coordinates across the ten folds is as low as 0.05.

The applied LSTM networks consist of 128 memory blocks with one memory cell per block. The number of input nodes corresponds to the number of different features per utterance whereas the number of output nodes corresponds to the number of target classes. Zero mean Gaussian noise with standard deviation 0.6 was added to the inputs during training to improve generalization. All networks are trained using

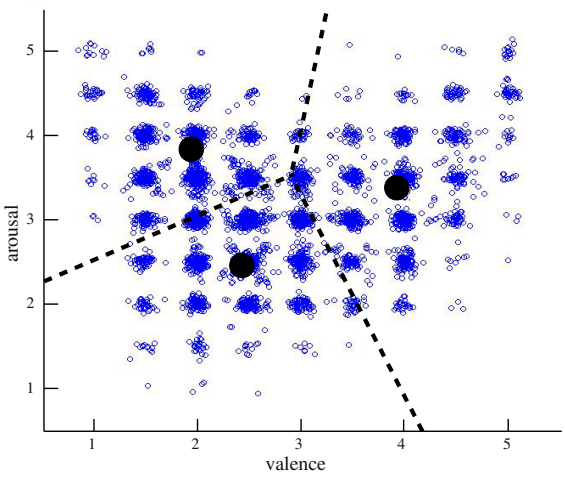


Figure 4.7: Annotations of the IEMOCAP training set for fold 1 with cluster midpoints (black circles) and resulting class borders (dotted lines) for the 3-class task; a small amount of random noise is added to the annotations for visualization purposes.

a learning rate of 10^{-5}. The bidirectional networks contain 128 memory blocks per input direction. As abort criterion for training, the classification performance on a validation set was evaluated. The validation set consisted of the utterances of two randomly selected speakers from the training split.

As an alternative classification approach, a dynamic, generative classification framework using Hidden Markov Models was examined. The motivation is to model the underlying dynamics of audio-visual emotional expression. Fully-connected 3-state HMMs were trained for the facial and vocal modality, as well as for the audio-visual setup. For each classification task, one HMM was trained for each class using the training utterances and during the test stage, the most probable class was recognized. Here, frame-level features were used, as opposed to the BLSTM experiments where statistical functionals of features were processed. For the facial HMMs, a 60-dimensional feature vector was used, containing 30 normalized PFA features and their first derivatives. For the vocal HMMs, a 58-dimensional feature vector containing 27 normalized MFBs, normalized pitch and energy values and their first derivatives was applied. Audio-visual HMMs were built by combining the synchronous face and speech features at the feature level (118 dimensions). In order to have a rough, local description of the past emotional context, a first-order 'language model' (LM) was incorporated into the HMM classification framework. Specifically, from the training set of each fold, the number of transitions for each pair of the classes was counted. In that way, an estimate of the transition probabilities from one class to the other can be obtained. During the test stage, the class that

Table 4.15: Recognition performances in [%] for discriminating three levels of valence and arousal using audio (A) and visual (V) features: accuracy (acc.), unweighted recall (rec.), precision (prec.), and F1-measure (F1).

classifier	features	acc.	rec.	prec.	F1
		valence			
HMM	A	47.08	47.11	48.20	47.62
HMM	V	55.53	60.07	56.77	58.29
HMM	A+V	59.27	58.81	61.68	60.17
HMM+LM	A+V	61.07	62.85	61.11	61.91
SVM	A+V	61.49	61.50	63.59	61.45
LSTM	A+V	62.35	63.77	63.80	63.66
BLSTM	A+V	63.92	64.71	65.87	**65.18**
		arousal			
HMM	A	55.06	61.68	50.93	55.77
HMM	V	43.87	51.86	47.48	49.30
HMM	A+V	51.33	52.56	60.16	55.90
HMM+LM	A+V	57.65	57.62	57.75	**56.89**
SVM	A+V	70.53	50.39	60.30	51.30
LSTM	A+V	68.84	50.58	58.45	53.89
BLSTM	A+V	67.31	52.53	58.46	55.18

maximizes the product of the class probability for the current utterance, and the transition probability from the previous class to the current class was selected.

Furthermore, we compare the performance of the BLSTM networks to static classification of utterance level feature functionals via Support Vector Machines. The SVMs have a polynomial kernel (degree 1) and are trained using the Sequential Minimal Optimization algorithm.

The experiments are organized in a cyclic leave-one-speaker-out cross validation. The mean and standard deviation of the number of test and training utterances across the folds is 498±60 and 4475±61, respectively. For each fold, the accuracy and the (unweighted) precision, recall, and F1 measure were computed. The presented recognition results are the subject-independent averages over the ten folds.

Table 4.15 shows the recognition performances for discriminating three levels of valence and arousal, respectively. The unimodal HMM results confirm the general experience that facial features tend to be more important for valence classification while acoustic features are well-suited for arousal classification. Generally, multimodal classification outperforms unimodal AER. The best F1-measure for valence can be obtained using a BLSTM network (65.18 %), and the performance for unidirectional LSTM networks is only slightly lower (F1-measure of 63.66 %). This indicates that modeling the long-range context between successive utterances is very important. Incorporating a bigram language model into the HMM recognition framework, also leads to a performance gain, which again underlines the importance of context modeling. For arousal, we observe a lower performance of LSTM modeling. A major

Table 4.16: Recognition performances in [%] for discriminating three, four, and five clusters in emotional space using audio (A) and visual (V) features: accuracy (acc.), unweighted recall (rec.), precision (prec.), and F1-measure (F1).

classifier	features	acc.	rec.	prec.	F1
3 clusters					
HMM	A+V	67.03	66.87	67.99	67.37
HMM+LM	A+V	67.03	66.89	68.04	67.41
SVM	A+V	68.91	68.58	69.20	67.95
LSTM	A+V	70.17	69.54	71.20	70.33
BLSTM	A+V	72.31	71.88	72.84	**72.34**
4 clusters					
HMM	A+V	55.70	55.93	55.69	55.73
HMM+LM	A+V	56.87	56.33	56.44	56.31
SVM	A+V	60.77	58.36	59.12	57.10
LSTM	A+V	63.69	61.00	62.86	61.87
BLSTM	A+V	64.30	61.92	63.85	**62.78**
5 clusters					
HMM	A+V	49.94	50.94	48.87	49.76
HMM+LM	A+V	50.81	50.99	50.17	50.41
SVM	A+V	51.49	49.52	50.99	48.55
LSTM	A+V	56.19	53.89	56.25	**55.00**
BLSTM	A+V	56.31	53.76	56.13	54.84

reason for this is the imbalance of the class distribution: The majority of utterances are labeled as 'medium arousal' so that the amount of training data for the remaining two arousal classes is insufficient (also see Figure 4.7). For the arousal task, the HMM+LM framework handles this class imbalance better and achieves the highest performance (F1-measure of 56.89 %).

A more balanced class distribution and a better class separability can be obtained when jointly classifying valence and arousal by assigning the utterances to clusters that are learned in a data-driven way: For the distinction between three clusters, BLSTM networks achieve an F1-measure of 72.34 % (see Table 4.16). For four and five clusters they achieve F1-measures of 62.78 % and 55.00 % respectively. For all cluster prediction tasks, we observe similar trends: LSTM modeling prevails over HMM and SVM classification and bidirectional context outperforms unidirectional context (except for the five-cluster task, where there is no significant difference between LSTM and BLSTM). The HMM+LM and SVM classification frameworks achieve comparable, and lower, results.

In general, the BLSTM framework which is able to incorporate long-range bidirectional context information, prevails over other classification frameworks which use no or limited contextual emotional information, such as the SVM and the HMM+LM respectively.

4.2.2 Sequential Jacobian Analysis

Since BLSTM neural networks can make use of an arbitrary, self-learned amount of past and future contextual information (see Section 2.3.9), they seem well suited for emotion recognition applications where modeling the *emotional history* during a conversation is of interest [161]. As shown in the last sections, the application of BLSTM networks for speech-based [32, 294] and audio-visual [167, 289] emotion recognition enables performance gains in context-sensitive AER when compared to systems that do not make use of context information, such as context-free HMM or SVM-based approaches. Yet, the actual *amount* of contextual information that is exploited within a BLSTM network for emotion classification has not been investigated so far and networks are often seen as a 'black box' being less transparent than, e. g., HMM systems. This section presents a methodology first, to systematically determine the amount of context that is used by BLSTM networks to classify utterances of a speaker during a conversation and, second, to examine the extent that this available context contributes to the overall BLSTM performance [290]. The goal is to better understand the effect of BLSTM modeling of human emotions and to gain insights supporting future AER system design. For the analyses, the same audio-visual recognition framework and database as introduced in Section 4.2.1 is applied.

Experiments and Results

Again, we aim to assess speaker independent AER performance of BLSTM networks when carrying out a cyclic leave-one-speaker-out cross validation on the IEMOCAP database. Both, the recognition task and the emotion recognition system are the same as in Section 4.2.1. To investigate the importance of having meaningful available context information during BLSTM network training and decoding, all BLSTM classification experiments were repeated using randomly shuffled data. Specifically, the utterances of a given conversation are processed in arbitrary order so that the network is not able to make use of meaningful context information. As can be seen in Table 4.17, this downgrades recognition performance (average F1-measure) for all classification tasks. To test the statistical significance of this result, a paired t-tests was performed to compare the average F1-measures, leading to the result that BLSTM networks perform significantly worse ($p=0.05$) when the input utterances are randomly shuffled. The performance gap suggests that the good performance of the BLSTM classifiers is to a large extent due to their ability to effectively learn an adequate amount of relevant emotional context from past and future observations. It can also be interpreted as evidence that learning to incorporate temporal context information is relevant for human emotion modeling.

An impression of the amount of contextual information that is used by the BLSTM network can be gained by measuring the sensitivity of the network out-

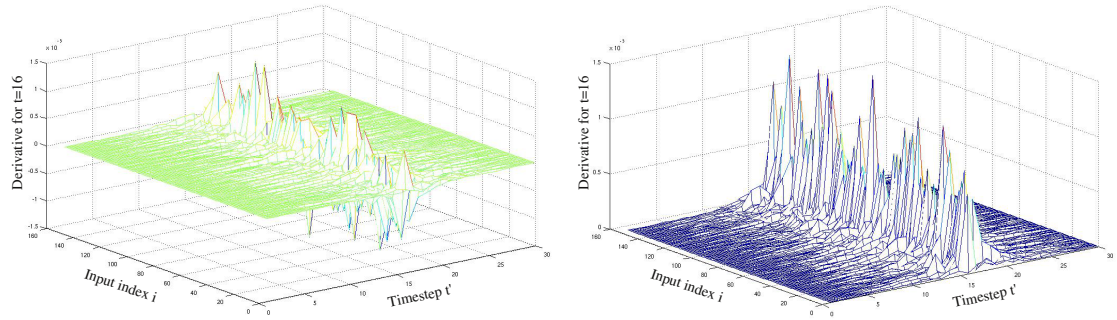
Table 4.17: Recognition performances in [%] for BLSTM networks and five different classification tasks. BLSTM networks trained on the original sequence of utterances and on utterances that are randomly shuffled, using audio (A) and visual (V) features: accuracy (acc.), unweighted recall (rec.), precision (prec.), and F1-measure (F1).

classifier	features	acc.	rec.	prec.	F1
valence					
BLSTM	A+V	63.92	64.71	65.87	65.18
BLSTM(shuffled)	A+V	59.80	58.97	60.46	59.63
arousal					
BLSTM	A+V	67.31	52.53	58.46	55.18
BLSTM(shuffled)	A+V	69.18	46.39	60.20	52.15
3 clusters					
BLSTM	A+V	72.31	71.88	72.84	72.34
BLSTM(shuffled)	A+V	68.02	66.69	69.08	67.84
4 clusters					
BLSTM	A+V	64.30	61.92	63.85	62.78
BLSTM(shuffled)	A+V	61.51	57.95	60.72	59.24
5 clusters					
BLSTM	A+V	56.31	53.76	56.13	54.84
BLSTM(shuffled)	A+V	53.25	50.95	53.17	51.94

puts to the network inputs. When using feedforward neural networks, this can be done by calculating the Jacobian matrix J whose elements J^{ki} correspond to the derivatives of the network outputs o^k with respect to the network inputs x^i. To extend the Jacobian to recurrent neural networks, we have to specify the timesteps (representing utterances) at which the input and output variables are measured. Thus, we calculate a four-dimensional matrix called the *sequential Jacobian* [89] to determine the sensitivity of the network outputs at time t to the inputs at time t':

$$J_{tt'}^{ki} = \frac{\partial o_t^k}{\partial x_{t'}^i}. \tag{4.2}$$

Figure 4.8(a) shows the derivatives of the network outputs at time $t = 16$ with respect to the different network inputs (i. e., features) at different timesteps t' for a randomly selected session consisting of 30 utterances when using a BLSTM network for the discrimination of five emotional clusters. Since we use BLSTM networks for utterance-level prediction, each timestep corresponds to one utterance. Note that the absolute magnitude of the derivatives is not important. We are rather interested in the relative magnitudes of the derivatives to each other, since this determines the sensitivity of outputs with respect to inputs at different timesteps. Of course the highest sensitivity can be detected at timestep $t' = 16$, which means that the current input has the most significant influence on the current output. However,



(a) Derivatives at time $t = 16$.

(b) Absolute values of the derivatives.

Figure 4.8: Derivatives of the network outputs at time $t = 16$ with respect to the different network inputs at different timesteps t'; randomly selected session consisting of 30 utterances (BLSTM network for the discrimination of five emotional clusters).

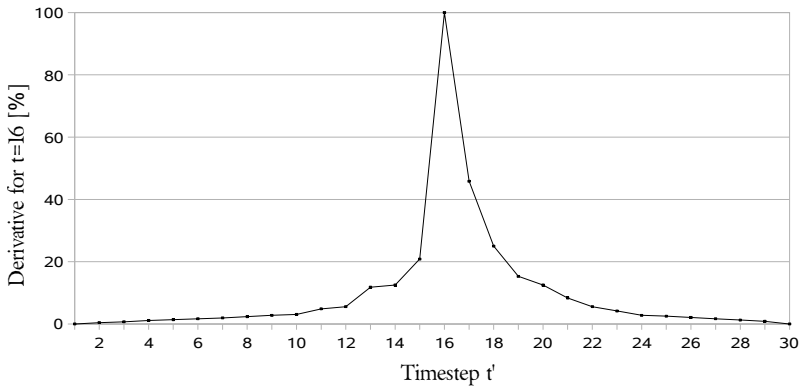
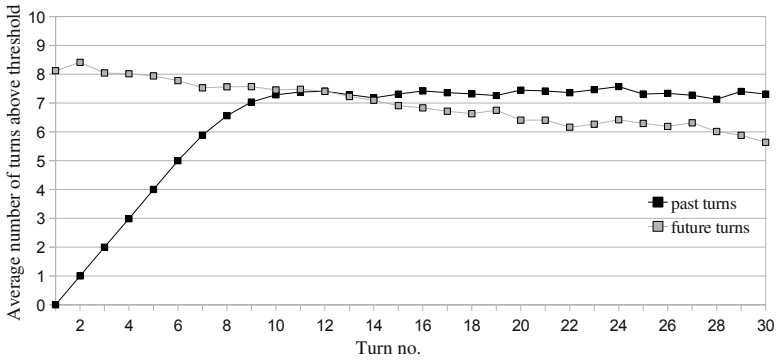


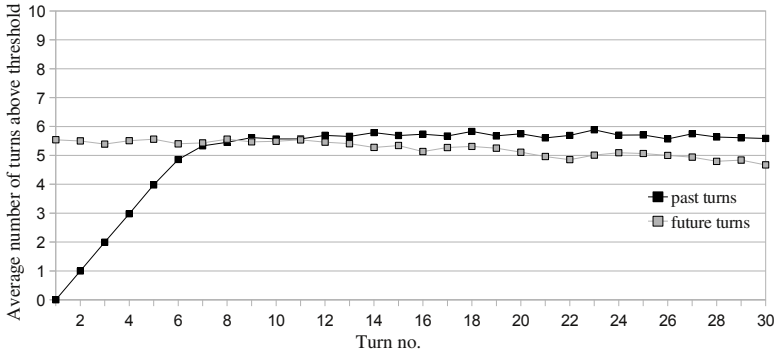
Figure 4.9: Derivatives summed up over all inputs and normalized.

also for timesteps smaller or greater than 16, derivatives different from zero can be found. This indicates that also past and future utterances affect the current prediction. As positive and negative derivatives are of equal importance, Figure 4.8(b) shows the absolute values of the derivatives in Figure 4.8(a). Finally, Figure 4.9 displays the corresponding derivatives summed up over all inputs and normalized to the magnitude of the derivative at $t' = 16$.

In order to systematically evaluate how many past and future inputs are relevant for the current prediction, we determine how many utterances before and after the current utterance (e. g., utterance 16 in the example given in Figure 4.9) have a sensitivity greater or equal to 3 % of the maximum sensitivity. To this end, we calculate projections of the sequential Jacobian as in Figure 4.9 for each timestep t in each session and each fold. Figure 4.10(a) shows the number of relevant past



(a) BLSTM network trained on utterances in the correct order.



(b) BLSTM network trained on randomly shuffled data.

Figure 4.10: Average number of relevant past and future utterances dependent on the position in the sequence when using a BLSTM network for the discrimination of five emotional clusters (3 % sensitivity-threshold).

and future utterances dependent on the position in the sequence (i. e., dependent on the utterance number within a session) when using a BLSTM network for the discrimination of five clusters in the emotional space (the corresponding figures for the other classification tasks are very similar and are omitted). The number of past utterances for which the sensitivity lies above the 3 % threshold increases approximately until the eighth utterance in a session. As more and more past utterances become available, the graph converges to a value of between seven and eight, meaning that roughly seven to eight utterances of past context are used for a prediction. For the first few emotion predictions the network uses about eight utterances of future context. The slight decrease of the number of used future utterances for higher utterance numbers (i. e., for utterances occurring later in a session) is simply due to the fact that some sessions consist of less than 30 utterances, which means that towards the end of a session, less future utterances are available on average. Figure

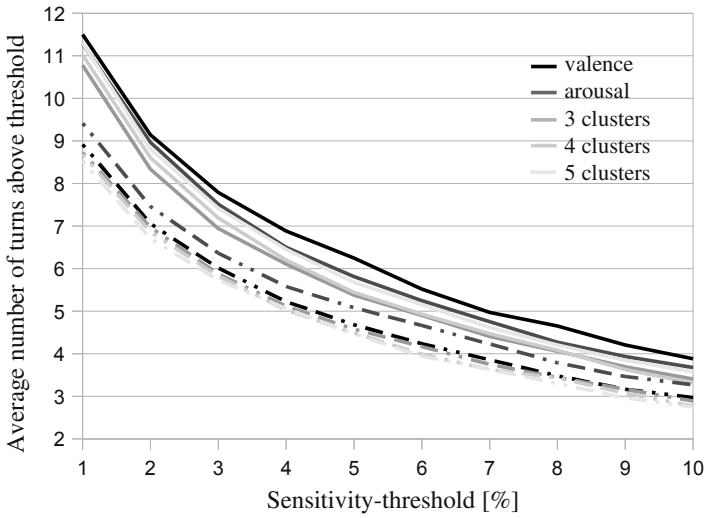


Figure 4.11: Average number of relevant past utterances dependent on the sensitivity-threshold; straight lines: utterances in correct order; dashed lines: randomly shuffled data.

4.10(b) shows the number of relevant preceding and successive utterances for the BLSTM network trained on randomly shuffled data. As can be seen, the amount of used context is less than for the BLSTM trained on correctly aligned utterances. Even though no reasonable emotional context can be learned when training on arbitrarily shuffled data, the network still uses context. One reason for this could be that BLSTM attempts to learn other session-specific characteristics, such as speaker characteristics.

Figure 4.11 shows the number of relevant past utterances when considering different classification tasks and sensitivity-thresholds from 1 to 10 %. Again, we can see that networks trained on randomly shuffled data use less context (see dashed lines in Figure 4.11) while the amount of context exploited for the different classification tasks is relatively similar.

4.2.3 Emotion Recognition from Acoustic, Linguistic, and Facial Movement Features

According to [99], 'second generation' AER systems have to focus on realistic human behavior data and need to model the complexity, subtlety, continuity, and dynamics of human emotions. As discussed in Section 4.1.1, we are currently observing a shift from modeling prototypical emotional categories such as *anger* or *happiness* to viewing human affect in a continuous orthogonal way by defining *emotional dimen-*

sions including for example *arousal* and *valence*. This allows researchers to model emotions either in a fully value-continuous way (e. g., via regression approaches as in [96, 276]) or by using discretized emotional dimensions, for example for the discrimination of high vs. low arousal or positive vs. negative valence (see Section 4.1.2). Systems applying the latter approach have the advantage of detecting a defined set of user states which can be easily used as input for automatic dialog managers that have to decide for an appropriate system response given a certain affective state of the user [206].

The 2011 Audio/Visual Emotion Challenge [221] focuses on exactly these kinds of discretized emotional dimensions. More specifically, this challenge was organized to provide research teams with unified training, development and test data sets that can be used to compare individual approaches applying a defined test scenario and defined performance measures. The task was to classify two levels of *arousal*, *expectation*, *power*, and *valence* from audio-visual data as contained in the SEMAINE database [155]. Compared to rather 'friendly' test conditions as considered in the early days of emotion recognition research [217], this scenario is exceedingly challenging and typically leads to results from below chance-level accuracies to around 70 % accuracy for a two-class task.

In this section, an LSTM-based AER framework exploiting acoustic, linguistic, and visual information is introduced. In contrast to the system proposed in Section 4.2.1, which is based on facial marker information, we now consider a fully automatic audio-visual recognition framework in which facial movement features are extracted without the need for facial markers [284, 285]. Focusing on the Audiovisual Sub-Challenge of the 2011 Audio/Visual Emotion Challenge, we investigate which modalities contribute to the discrimination between high and low levels of arousal, expectation, power, and valence. Furthermore, we analyze which emotional dimensions benefit the most from unidirectional and bidirectional Long Short-Term Memory modeling. By comparing the obtained results with all other contributions to the Audiovisual Sub-Challenge task, an overview over recent approaches towards audiovisual emotion recognition is provided, including an analysis of their strengths and weaknesses with respect to the modeling of the different emotional dimensions.

The audio-visual LSTM technique is evaluated on both, the development set

Table 4.18: Overview of the SEMAINE database as used for the 2011 Audio/Visual Emotion Challenge [221].

	Train	**Develop**	**Test**	**Total**
# Sessions	31	32	32	95
# Frames	501 277	449 074	407 772	1 358 123
# Words	20 183	16 311	13 856	50 350
Avg. word duration [ms]	262	276	249	263

and the official test set of the Audiovisual Sub-Challenge. This allows a comparison with various other methods proposed for this task so far, including Support Vector Machines [201, 221], extreme learning machine based feedforward neural networks (ELM-NN) [36], AdaBoost [169], Latent-Dynamic Conditional Random Fields (LD-CRF) [187], Gaussian Mixture Models [127], and a combined system consisting of multilayer perceptrons and HMMs [86].

Database

The freely available audio-visual SEMAINE corpus[1] [155] was collected to investigate social signals that typically occur during interactions between humans and virtual agents (see also Section 3.1.6). For the recordings, the participants were asked to speak to the four different emotionally stereotyped characters introduced in Section 2.1.1. The data used for the 2011 Audio/Visual Emotion Challenge[2] is based on the 'Solid-SAL' recordings, i. e., human operators imitated the behavior of artificial agents. Further details on the interaction scenario can be found in [221].

Video was recorded at 49.979 frames per second at a spatial resolution of 780 x 580 pixels and 8 bits per sample, while audio was recorded at 48 kHz with 24 bits per sample. Both, the user and the operator were recorded from a frontal view by both a greyscale camera and a color camera. In addition, the user is recorded by a greyscale camera positioned on one side of the user to capture a profile view of the whole scene, including their face and body. Audio and video signals were synchronized with an accuracy of 25 μs.

The 24 recordings considered in the Audio/Visual Emotion Challenge consisted of three to four character conversation sessions each and were split into three speaker independent partitions: a training, development, and test partition each consisting of eight recordings. As the number of character conversations varies between recordings, the number of sessions is different per set: The training partition contains 31 sessions, while the development and test partitions contain 32 sessions. Table 4.18 shows the distribution of data in sessions, video frames, and words for each partition.

In what follows, we exclusively focus on the *Audiovisual Sub-Challenge* of the emotion challenge. Thus, the applied test set consists only of the sessions that are intended for this sub-challenge, meaning only 10 out of the 32 test sessions. For the challenge, the originally continuous affective dimensions *arousal*, *expectation*, *power*, and *valence* were redefined as binary classification tasks by testing at every frame whether they are above or below average. As argued in [76], these four dimensions account for most of the distinctions between everyday emotion categories. Arousal is the individual's global feeling of dynamism or lethargy and subsumes mental activity as well as physical, preparedness to act as well as overt activity. Expectation also subsumes various concepts that can be separated as expecting, anticipating, being

[1] www.semaine-db.eu

[2] www.avec2011-db.sspnet.eu

Figure 4.12: Examples for low and high arousal, expectation, power, and valence.

Figure 4.13: Series of word-level screenshots of a user together with the corresponding valence annotation.

taken unaware. Power subsumes two related concepts, power and control. Valence subsumes whether the person rated feels positive or negative about the things, people, or situations at the focus of his/her emotional state. Figure 4.12 shows example screenshots for low and high arousal, expectation, power, and valence. In Figure 4.13, a series of word-level screenshots of a user and the corresponding valence annotation can be seen. A detailed description on the annotation process can be found in [221].

The word timings were obtained by running an HMM-based speech recognizer in forced alignment mode on the manual transcripts of the interactions. The recognizer used tied-state cross-word triphone left-right (linear) HMM models with three emitting states and 16 Gaussian mixture components per state.

Audio Feature Extraction

The applied acoustic feature extraction approach is based on a large set of low-level descriptors and derivatives of LLD combined with suited statistical functionals to

capture speech dynamics within a word. All features and functionals are computed using the openSMILE toolkit [73]. The audio feature set is identical to the 2011 Audio/Visual Emotion Challenge baseline acoustic feature set applied in [221] and consists of 1 941 features, composed of 25 energy and spectral related low-level descriptors x 42 functionals, 6 voicing related LLD x 32 functionals, 25 delta coefficients of the energy/spectral LLD x 23 functionals, 6 delta coefficients of the voicing related LLD x 19 functionals, and 10 voiced/unvoiced durational features. Details on the LLD and functionals are given in [221].

Linguistic and Non-Linguistic Feature Extraction

Linguistic features are extracted using the SEMAINE 3.0 ASR system [206]. It applies openSMILE as front-end to extract 13 MFCCs together with first and second order temporal derivatives every 10 ms (window size 25 ms). The HMM back-end is based on the open-source Julius decoder [135]. A back-off bigram language model as well as the tied-state triphone acoustic models were trained on the COSINE corpus [241], the SAL database [64], and the training set of the SEMAINE database [155]. All of these corpora contain spontaneous, conversational, and partly emotional speech. The phoneme HMMs consist of three states with 16 Gaussian mixtures per state. Models for non-linguistic vocalizations (laughing, breathing, sighing) consist of nine emitting states.

Typically, one (key)word is detected for every audio chunk (which correspond to single words), however the recognizer is not restricted to detect exactly one word, thus, insertions and deletions are possible. From the detected sequence of words, a Bag of Words vector is computed. The general procedure is as follows:

- a word list (also including non-linguistic vocalizations) is built from all the recognized words in the training and development set,
- words that occur less than 10 times in the union of training and development set are removed from the word list,
- the dimensionality of the Bag of Words vector equals the size of the remaining word list (141 words),
- for the current chunk a Bag of Words vector is built by setting each element corresponding to a detected word to the word confidence score; all other elements in the vector are set to zero; if the recognizer output for one word is empty, all elements of the vector are set to zero.

Visual Feature Extraction

In order to compute the visual low-level features applied in the proposed LSTM-based audio-visual emotion recognition framework we go through the steps depicted

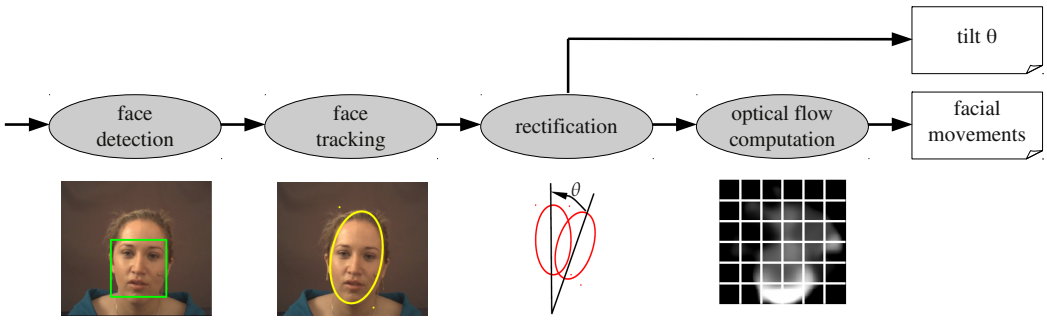


Figure 4.14: Basic steps for the computation of the low-level visual features.

in the block diagram in Figure 4.14. Note that only data from the frontal view color camera is used. In Block 1, the face is detected by a Viola Jones face detector [250]. From the detected face a histogram is built for tracking (Block 2 in Figure 4.14). The face detected in the first frame is cut out and transformed into the hue-saturation-value (HSV) color space and the entries of the histogram M are computed:

$$M(h,s,v) = \sum_{x,y} \begin{cases} 1 & \text{if } T_H(x,y) = h \cap T_S(x,y) = s \\ & \quad \cap \ T_V(x,y) = v \\ 0 & \text{else} \end{cases} \tag{4.3}$$

where T is the detected face region that is taken as template. The indices h, s, and v denote hue, saturation, and value, respectively. Each of the three components of the HSV color model has 20 bins in the histogram. For each pixel $I(x,y)$ in the current image the probability of a facial pixel can be approximated by

$$p_f(x,y) = \frac{M(I_H(x,y), I_S(x,y), I_V(x,y))}{N}, \tag{4.4}$$

with N being the number of template pixels that have been used to create the histogram. The face is considered as detected when there is a sufficiently large amount of facial pixels in the upper half of the image. Subsequently, the face is tracked with a camshift tracker [27] which takes the probability image as input. The location, the size, and the orientation of the face are computed according to [27]. One advantage of the camshift tracker is that it is comparatively robust which is important for a reliable facial movement feature extraction. Furthermore, it operates fast and also computes the tilt of the head, as can be seen in Figure 4.14.

Subsequently, the face is cut out and the tilt is undone (Block 3). The face in the up-right pose is compared to the previous frame. Note that the tilt θ itself is used as one facial low-level feature. In Block 4, 98 facial movement features are extracted as follows. The optical flow between the rectified face and the face of

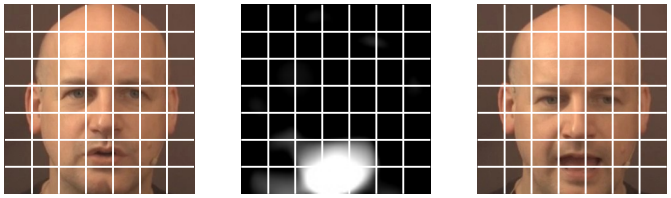


Figure 4.15: Example for optical flow computation: Between the frames there is a substantial change in the mouth region.

the previous frame is computed. As an example, Figure 4.15 depicts a subject that opens its mouth. In this case the y-values of the rectangles of the lip region are high. The cut out face is then subdivided into $7 \times 7 = 49$ rectangles. For each of these rectangles the average movement in x- and y-direction is computed. These movements are further features in addition to the tilt θ, so that a total of 99 visual low-level features are extracted per frame.

In order to map the sequence of frame-based video features to a single vector describing the word-unit, statistical functionals are applied to the frame-based video features and their first order delta coefficients. This step is conceptually the same as for the audio features, except that different functionals are used, considering the different properties of the video features. Note that words shorter than 250 ms are expanded to 250 ms which means that the time windows containing very short words can contain (fractions of) other words. The following functionals are applied to frame-based video features: arithmetic mean (for delta coefficients: arithmetic mean of absolute values), standard deviation, 5% percentile, 95% percentile, and range of 5% and 95% percentile. Fewer functionals as for audio features are used to ensure a similar dimensionality of the video feature vector and the audio feature vector. The resulting per-word video feature vector has $5 \times 2 \times 99 = 990$ features.

Figure 4.16 shows the importance of the subregions of the face for the video-based discrimination between high and low arousal, expectation, power, and valence. Importance was evaluated employing the ranking-based information gain attribute evaluation algorithm implemented in the Weka toolkit [102]. As input for the ranking algorithm, all 990 features extracted from each instance in the training set were used together with the ground truth annotation of the respective emotional dimension. In Figure 4.16, the shading of the facial regions indicates the importance of the features corresponding to the respective region. As expected, the small remaining background parts are less important than the subregions containing facial information. Within the face, the eye regions contain slightly more information. Overall, we observe that relevant information about a subject's emotional state can be found in multiple regions of the face and not just in the upper or lower face, corresponding to the eye and mouth region, respectively.

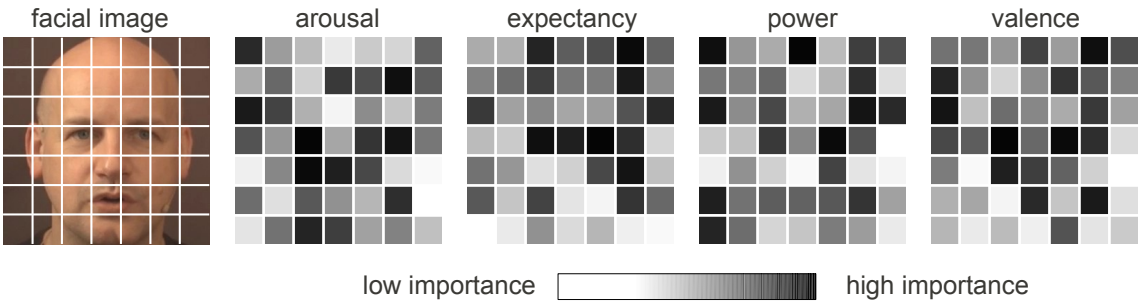


Figure 4.16: Importance of facial regions for video feature extraction according to the ranking-based information gain attribute evaluation algorithm implemented in the Weka toolkit [102]. Information gain is evaluated for each emotional dimension. The shading of the facial regions indicates the importance of the features corresponding to the respective region.

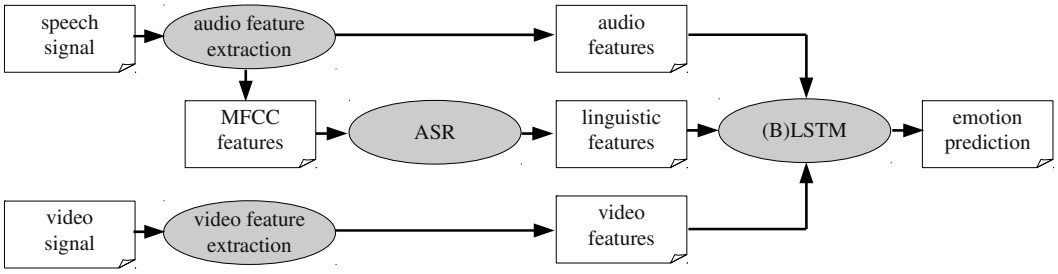


Figure 4.17: System architecture for early fusion of acoustic, linguistic, and video features.

System Architecture

Figure 4.17 shows the overall system architecture of the LSTM-based audio-visual emotion recognition framework applying early (i. e., feature-level) fusion. The openSMILE audio feature extractor provides framewise MFCC features for the speech recognition module as well as statistical functionals of acoustic features for the LSTM network. In addition to audio features, the network also processes the linguistic feature vector provided by the ASR system and video features computed by the facial feature extractor to generate the current emotion prediction.

Experiments and Results

All experiments were carried out on the Audiovisual Sub-Challenge task as described in [221]. To gain first insights concerning the optimal combination of modalities (i. e., acoustic, linguistic, and visual features) and the number of training epochs needed for LSTM network training, initial experiments were performed using the training set for network training and the development set for testing, before the actual challenge task was considered which consists in training on the union of the training and the development set and testing on the test set. The task is to discriminate between high and low arousal, expectation, power, and valence. As the class distribution in the training set is relatively well balanced, the official challenge measure is weighted accuracy, i. e., the recognition rates of the individual classes weighted by the class distribution. However, since the instances of the development and test sets are partly unbalanced with respect to the class distributions, unweighted accuracies (equivalent to unweighted average recall) are also reported. This imbalance holds in particular for the Audio and Audio-Visual Sub-Challenge as they consider word-level modeling rather than frame-based recognition.

We investigate the performance of both, bidirectional LSTMs and unidirectional LSTM networks for fully incremental on-line audio-visual affect recognition. Separate networks were trained for each emotional dimension. The following modality combinations were considered: acoustic features only, video features only, acoustic and linguistic features (including non-linguistic vocalizations), acoustic and video features, as well as acoustic, (non-)linguistic, and video features.

All LSTM networks consist of 128 memory blocks and each memory block contains one memory cell. Again, the number of input nodes corresponds to the number of different features per speech segment and the number of output nodes corresponds to the number of target classes, i. e., two output nodes were used, representing high and low arousal, expectation, power, and valence, respectively. To prevent overfitting of the neural networks to the training data, a small amount of noise (Gaussian noise with standard deviation 0.6) was added to the inputs at each training epoch. As in previous experiments, all networks were trained using a learning rate of 10^{-5}. The bidirectional networks consist of two hidden layers (one for forward and one for backward processing) with 128 memory blocks per input direction. Parameters such as learning rate and the number of memory blocks were configured according to experience with similar recognition tasks (see [289], for example). To validate whether better recognition performance can be obtained when changing the number of memory blocks, hidden layer sizes of between 80 and 160 memory blocks were evaluated on the development set. Yet, for none of the modality combinations a modified hidden layer size could significantly outperform networks using the default setting of 128 memory blocks. The resulting number of variables that need to be estimated during network training is equivalent to the number of weights in the network, e. g., an LSTM network that processes the full feature set consisting of

Table 4.19: Development set of the Audiovisual Sub-Challenge; **no feature selection**: weighted accuracies (WA) and unweighted accuracies (UA) for the discrimination of high and low arousal, expectation, power, and valence using acoustic (A), linguistic (L), and video (V) features combined with different classifiers. LF: late fusion; the best weighted accuracies for each emotional dimension are highlighted.

		arousal		expectation		power		valence		**mean**
classifier	**features**	**WA**	UA	**WA**	UA	**WA**	UA	**WA**	UA	**WA**
BLSTM	A	**68.5**	69.3	64.3	53.5	66.1	53.3	66.3	56.1	66.3
BLSTM	A+L	67.8	69.0	64.8	52.0	65.5	53.9	66.3	56.2	66.1
LSTM	A	**68.5**	68.6	66.1	55.9	64.7	56.1	65.6	55.2	66.2
LSTM	A+L	68.2	68.8	65.2	51.9	**66.2**	55.0	63.8	55.9	65.9
SVM [221]	A	63.7	64.0	63.2	52.7	65.6	55.8	58.1	52.9	62.7
BLSTM	V	62.3	62.9	62.3	51.8	55.2	53.0	63.3	60.5	60.8
LSTM	V	60.3	61.3	60.4	57.7	57.0	50.4	64.0	57.9	60.4
SVM [221]	V	60.2	57.9	58.3	56.7	56.0	52.8	63.6	60.9	59.5
BLSTM	A+V	67.7	68.0	63.1	53.4	60.6	55.0	67.2	61.8	64.7
BLSTM	A+L+V	66.9	67.0	66.2	57.3	63.4	52.3	65.9	61.5	65.6
LSTM	A+V	68.0	67.5	65.7	57.7	63.8	54.7	65.5	59.5	65.8
LSTM	A+L+V	67.4	66.8	65.3	56.7	61.7	54.2	67.6	62.8	65.5
BLSTM (LF)	A+V	67.9	69.3	65.0	53.2	64.0	55.5	**69.8**	61.3	**66.7**
BLSTM (LF)	A+L+V	67.0	68.6	65.7	51.6	63.6	55.7	**69.8**	61.2	66.5
LSTM (LF)	A+V	62.6	64.3	**67.6**	57.6	65.1	56.0	68.2	57.7	65.9
LSTM (LF)	A+L+V	66.3	67.4	63.9	58.1	66.0	53.9	66.4	58.2	65.7

acoustic, linguistic, and video information has 2 094 210 weights.

As abort criterion for training, the classification performance on the development set was periodically evaluated and the network which achieved the best results on the development set was used. The number of training epochs needed until the best performance was reached was around 30 epochs for recognition of expectation, power, and valence, and 60 epochs for arousal classification. All input features were mean- and variance normalized with means and variances computed from the training set.

Alternatively to early fusion of modalities on the feature level, also a simple late fusion (LF) technique was considered. The late fusion approach consisted in training separate networks for each modality and summing up the output activations of the respective networks before deciding about the estimated class that can be inferred from the highest (overall) output activation.

Table 4.19 shows both, weighted accuracies (WA) and unweighted accuracies (UA) obtained when training on the training set of the 2011 Audio/Visual Emotion Challenge and testing on the development set. Results are shown for BLSTMs, LSTM networks, and for the SVM approach applied in [221] and various modality combinations are considered. Note that the results for SVMs processing audio and video data are missing as they are not reported in [221]. For the Audiovisual Sub-Challenge, the development set data has been used in [221] only to train the fusion

Table 4.20: Development set of the Audiovisual Sub-Challenge; **CFS feature selection**: weighted accuracies (WA) and unweighted accuracies (UA) for the discrimination of high and low arousal, expectation, power, and valence using acoustic (A), linguistic (L), and video (V) features combined with different classifiers. LF: late fusion; the best weighted accuracies for each emotional dimension are highlighted.

		arousal		expectation		power		valence		**mean**
classifier	**features**	**WA**	UA	**WA**	UA	**WA**	UA	**WA**	UA	**WA**
BLSTM	A	71.3	70.2	66.2	51.0	66.0	56.4	65.9	60.6	67.4
BLSTM	A+L	**73.7**	74.4	66.1	53.1	64.6	55.7	65.8	57.2	67.6
LSTM	A	70.4	69.8	**67.7**	54.6	64.9	58.8	63.1	55.3	66.5
LSTM	A+L	71.9	71.1	63.1	55.5	66.6	56.3	64.7	56.9	66.6
BLSTM	V	59.8	58.8	66.2	50.1	64.1	57.5	63.3	56.0	63.4
LSTM	V	62.7	61.5	66.0	50.1	**70.2**	62.4	64.3	52.7	65.8
BLSTM	A+V	67.8	69.5	64.3	52.3	60.1	57.0	64.7	58.8	64.2
BLSTM	A+L+V	69.9	70.7	63.3	50.4	61.9	56.1	61.4	55.9	64.1
LSTM	A+V	69.7	70.8	64.5	52.0	63.5	56.8	62.4	53.0	65.0
LSTM	A+L+V	70.4	71.3	65.7	53.3	63.5	55.9	62.9	53.2	65.6
BLSTM (LF)	A+V	68.5	67.5	66.7	50.4	64.2	52.7	**69.1**	60.6	67.1
BLSTM (LF)	A+L+V	72.3	72.3	66.6	50.9	64.4	54.0	67.9	58.5	**67.8**
LSTM (LF)	A+V	65.7	63.7	67.4	52.1	68.0	58.6	66.8	54.8	67.0
LSTM (LF)	A+L+V	64.8	63.5	67.1	54.9	68.1	57.3	65.7	56.4	66.4

engine – this, however, is not necessary for the proposed LSTM-based recognition engine since it uses either early fusion or a simple late fusion scheme that does not require training.

The performance difference between unidirectional and bidirectional LSTM networks is comparatively small. In some cases (e. g., classification of arousal using acoustic and linguistic features), LSTM networks perform even slightly, but not significantly better than BLSTM nets. This means that modeling only past context does not necessarily downgrade recognition results compared to bidirectional modeling, which is important for incremental on-line applications in which future context is not available due to real-time constraints. The performance of the different feature groups (acoustic, linguistic, video) heavily depends on the considered emotional dimension. For arousal, the best WA of 68.5 % is obtained for acoustic features only, which is in line with previous studies showing that audio is the most important modality for assessing arousal [289]. However, the classification of expectation seems to benefit from including visual information as the best WA (67.6 %) is reached for LSTM networks applying late fusion of audio and video modalities. Similar to arousal, power is best classified via speech-based features. Bidirectional networks for classifying power cannot be enhanced by linguistic features, however, for unidirectional modeling WA significantly increases from 64.7 % to 66.2 % when using linguistics in addition to audio features. For valence, the inclusion of video

information helps, leading to a WA of 69.8 % when using BLSTM networks and audio-visual data. The effectiveness of the emotion recognition approaches using only video information also depends on the emotional dimension. For arousal and expectation, BLSTM modeling of facial movement features prevails, while for power and valence, we observe slightly, but not significantly better results for SVM-based classification of local appearance descriptors as proposed in [221] and for unidirectional LSTM modeling. On average the best performance on the development set is obtained for bidirectional processing and acoustic and visual features (mean WA of 66.7 %). Yet, in this case there is no significant difference between bi- and unidirectional processing, as LSTM networks achieve almost the same WA on average (66.5 %). For each emotional dimension, context modeling via LSTM increases accuracies compared to the static SVM-based technique applied in [221]. Furthermore, late fusion tends to prevail over early fusion.

To investigate whether a smaller feature space leads to better recognition performance, all evaluations on the development set were repeated applying a Correlation-based Feature Subset Selection [269] for each modality combination. The corresponding results can be seen in Table 4.20. For most settings, CFS does not significantly improve the average weighted accuracy. However, for recognition based on video only, CFS leads to a remarkable performance gain, increasing the average WA from 60.4 % to 65.8 % for unidirectional LSTM networks.

The results for the official Audiovisual Sub-Challenge test set can be seen in Table 4.21. Networks were trained on the training and development set. According to optimizations on the development set, the number of training epochs was 60 for networks classifying arousal and 30 for all other networks. Networks processing video data only are based on a video feature set reduced via CFS, whereas for all other networks, no CFS was applied. All network parameters (number of memory blocks, learning rate, etc.) were identical to the previous set of experiments on the development set. BLSTM and LSTM modeling was compared to all other approaches proposed for the Audiovisual Sub-Challenge, including Support Vector Machines [201, 221], extreme learning machine based feedforward neural networks [36], AdaBoost [169], Latent-Dynamic Conditional Random Fields [187], Gaussian Mixture Models [127], and a combined system consisting of MLPs and HMMs [86]. Note, however, that these classification techniques do not necessarily use the same set of audio (and video) features, thus, Table 4.21 compares the overall approaches of different research groups rather than the effectiveness of the different classifiers. Similar to our experiments on the development set, audio features lead to the best result for arousal classification. When applying LSTM modeling we reach a WA of 71.2 % which is the best result reported for this task so far. Also for BLSTM-based classification of expectation using facial movement features, the obtained WA of 68.6 % is higher than what is reported for other techniques. For power, audio-visual classification with Latent-Dynamic Conditional Random Fields as proposed in [187] could not be outperformed. For valence, the audio features used in [201] lead to

Table 4.21: Test set of the Audiovisual Sub-Challenge: weighted accuracies (WA) and unweighted accuracies (UA) for the discrimination of high and low arousal, expectation, power, and valence using acoustic (A), linguistic (L), and video (V) features combined with different classifiers. LF: late fusion; the best weighted accuracies for each emotional dimension are highlighted.

		arousal		expectation		power		valence		**mean**
classifier	**features**	**WA**	UA	**WA**	UA	**WA**	UA	**WA**	UA	**WA**
BLSTM	A	69.2	69.1	63.1	54.6	59.6	52.9	68.7	57.4	**65.2**
LSTM	A	**71.2**	71.2	57.6	48.7	57.4	50.4	68.7	59.5	63.7
SVM [201]	A	59.8	59.7	63.6	50.0	57.9	48.4	**70.2**	54.9	62.9
ELM-NN [36]	A	52.0	52.3	63.7	50.1	62.2	50.7	69.1	50.0	61.8
AdaBoost [169]	A	57.6	57.5	62.2	49.6	54.2	47.9	60.3	47.6	58.6
LDCRF [187]	A	60.9	60.4	53.2	44.1	56.8	45.7	60.9	45.8	57.9
GMM [127]	A	55.3	55.2	56.1	50.7	49.1	45.3	50.9	48.4	52.9
BLSTM	V	43.1	42.9	**68.6**	62.0	44.8	41.0	51.7	52.4	52.1
LSTM	V	48.6	48.7	65.6	60.2	37.6	35.8	60.8	52.2	53.1
SVM [52]	V	47.8	47.4	62.0	54.8	57.9	47.4	69.6	50.2	59.3
LDCRF [187]	V	53.2	53.1	46.8	43.2	57.3	50.5	59.3	50.7	54.1
BLSTM	A+V	58.3	58.1	64.1	59.5	46.9	45.4	51.1	45.4	55.1
BLSTM	A+L+V	58.8	58.6	60.8	54.8	46.9	44.0	57.1	50.2	55.9
LSTM	A+V	56.3	56.2	61.6	54.1	46.7	45.8	61.2	53.9	56.5
LSTM	A+L+V	57.9	57.8	64.0	58.6	47.6	44.8	55.7	47.9	56.3
SVM [221]	A+V	67.2	67.2	36.3	48.5	62.2	50.0	66.0	49.2	57.9
LDCRF [187]	A+V	65.6	65.3	53.4	49.2	**62.9**	58.3	59.5	49.6	60.3
MLP [86]	A+V	54.1	54.3	58.5	57.8	42.7	40.0	44.8	35.9	50.0
BLSTM (LF)	A+V	69.5	69.4	63.6	54.5	55.8	49.3	69.6	59.2	64.6
BLSTM (LF)	A+L+V	63.3	63.2	62.9	53.0	53.1	48.4	57.6	46.9	59.2
LSTM (LF)	A+V	67.8	67.8	58.3	48.9	57.5	50.3	68.7	59.3	63.1
LSTM (LF)	A+L+V	70.3	70.3	62.5	52.2	56.0	50.4	69.2	58.5	64.5

the highest accuracy (70.2 %). When computing the average WA, we observe that a remarkable average performance can be obtained for systems exclusively processing audio data (for an overview over the statistical significance of the performance difference between the audio-based approaches, see Table 4.22). This suggests that even though video information helps for some emotional dimensions (such as expectation), on average acoustic features contribute the most to the assessment of affective states in the SEMAINE scenario. Interestingly, in the evaluations on the test set, the performance gap between early and late fusion of modalities via LSTM networks is significantly more pronounced than in the initial experiments on the development set. The average WA values we obtain for BLSTMs (65.2 %) and LSTMs (63.7 %) processing acoustic features prevail over all other approaches applied for this task by the challenge participants. Thus, similar to Section 4.2.1, we can conclude that the LSTM architecture is well suited for modeling affect in human conversations and that the exploitation of long-range temporal context not only helps humans to judge a conversational partner's emotional state but also increases the accuracy of

Table 4.22: Statistical significance of the average performance difference between the audio-based classification approaches denoted in the column and the approaches in the table header (evaluations on test set of the Audiovisual Sub-Challenge); '-': not significant; 'o' significant at 0.1 level; '+': significant at 0.05 level; '++': significant at 0.001 level. Significance levels are computed according to the z-test described in [235].

	LSTM	SVM	ELM-NN	AdaBoost	LDCRF	GMM [127]
BLSTM	o	+	++	++	++	++
LSTM		-	+	++	++	++
SVM [201]			-	++	++	++
ELM-NN [36]				+	++	++
AdaBoost [169]					-	++
LDCRF [187]						++

automatic affect sensing in human-computer interaction.

4.3 Summary and Outlook

Automatic non-verbal human behavior analysis is needed for intelligent systems that take into account the affective state of the user in order to enable natural and emotion-sensitive human-computer interaction. This chapter provided insights into novel speech-based and audio-visual machine learning techniques that exploit acoustic, linguistic, visual, and context information to recognize human emotions in spontaneous interactions. Rather than modeling discrete categorical emotions like 'happiness' or 'anger', this chapter focused on dimensional representations of affect by considering emotional dimensions such as *valence* and *arousal*. In Section 4.1.1, data-driven clustering of the valence-arousal space was investigated as an alternative to quadrant-based quantization. Depending on the application scenario, such 'clusters' in the emotional space can represent typical affective states in a more appropriate way [277].

Next, in Section 4.1.2 a novel technique for speech-based emotion estimation as needed for the SEMAINE system (see Section 2.1) was introduced [294]. In contrast to many other studies that report recognition results for the static classification of acted speech turns representing *emotional prototypes*, the experiments presented in Section 4.1.2 can be seen as a realistic evaluation of recognition accuracy under real-life conditions, where non-prototypical speech has to be classified using powerful techniques of dynamic speech modeling. The considered approach combines acoustic features obtained by the openSMILE on-line feature extractor with binary linguistic features produced by a Tandem BLSTM-DBN (see Section 3.1.3), which are then classified by a Long Short-Term Memory recurrent neural net. As outlined in Section 2.3.9, the LSTM architecture allows for the modeling of long-range contextual

information and thus enables a new technique of incremental affect recognition that does not require the computation of statistical functionals of features but captures the temporal evolution indirectly through LSTM memory cells. As an alternative to regression-based AER, a discriminatively trained LSTM system was created to distinguish quadrants of the emotional space. The prediction quality of the proposed system was shown to be comparable to the degree of consistency between different human labelers.

Section 4.1.3 introduced a speech-based framework for the assessment of a user's level of interest based on acoustic and linguistic information. Again, we exploit contextual knowledge via bidirectional Long Short-Term Memory networks to model how the user's interest evolves over time. Combining the BLSTM technique with the idea of bottleneck nets by designing LSTM networks with multiple hidden layers (including a narrow bottleneck layer in the middle) was shown to enable the generation of a compact low-dimensional feature representation within the network and to lead to improved interest recognition results. The Bottleneck-BLSTM strategy achieved remarkable results on the Interspeech 2010 Paralinguistic Challenge task [220], outperforming all other methods which have been proposed for this task so far [299].

In Section 4.1.4, we analyzed the effect of reverberation on automatic speech and emotion recognition in a child-robot interaction scenario involving spontaneous speech and non-prototypical emotions [302]. As expected, reverberation tends to degrade acoustic, linguistic, and combined acoustic-linguistic emotion recognition performance, however, the usage of reverberated training material can largely compensate the decrease of both, speech and emotion recognition accuracy. Multi-condition training leads to good performance for all reverberation conditions and reaches accuracies comparable to matched condition training. This shows that including reverberated data in the training set leads to more robust models – even if the training conditions do not exactly match the acoustic conditions during testing. Applying the multi-stream BLSTM-HMM ASR system detailed in Section 3.2.2, acoustic-linguistic AER accuracies of up to 70.3 % can be obtained for the recognition of negative emotions, which corresponds to results that were previously only reported for the fusion of multiple recognition engines [212].

As also facial features derived from the video signal can contain valuable information about the affective state of a user, Section 4.2 was devoted to audio-visual approaches towards non-verbal behavior analysis – again focusing on LSTM-based techniques for context-sensitive learning. First, in Section 4.2.1, a multi-modal framework for affect recognition from acoustic and facial marker features was investigated [289]. Various challenging subject-independent classification tasks revealed that BLSTM modeling prevails over conventional dynamic or static classification strategies.

In the light of Sections 4.1 and 4.2.1, which showed that context modeling via BLSTM networks is well-suited for emotion recognition applications, Section 4.2.2

introduced a methodology to analyze the amount of past and future context that is used by a BLSTM network to predict the emotional expression of a spoken utterance. In addition, we investigated the contribution of contextual information to the overall BLSTM performance, by randomly shuffling the order of utterances within a conversation so that the network fails to learn and exploit meaningful context. Systematic evaluations of the sequential Jacobian of trained BLSTM networks revealed that approximately eight past (and if available, also future) utterances are considered by the network as contextual information, when using a 3 % sensitivity-threshold as defined in [290]. When the input utterances are randomly shuffled, the BLSTM network uses fewer past and future utterances (around six). Emotion recognition results showed that performance significantly decreases when networks are trained on randomly shuffled data. This suggests that good performance of BLSTM-based approaches is due to the networks' ability to learn an adequate amount of relevant emotional context around the current observation. When such meaningful context is not present, performance degrades.

Finally, in Section 4.2.3, we considered an automatic emotion recognition framework exploiting acoustic, linguistic, and visual information in affective interactions [284]. LSTM context modeling was exploited to discriminate between high and low levels of arousal, expectation, power, and valence using statistical functionals of a large set of acoustic low-level descriptors, linguistic information (including non-linguistic vocalizations), and facial movement features. To get an impression of the effectiveness of context-sensitive LSTM-based audio-visual emotion recognition compared to other recently published approaches, the system was trained and evaluated on data sets defined in the 2011 Audio/Visual Emotion Challenge [221]. For the emotional dimensions *arousal* and *expectation*, the proposed framework led to the best accuracies reported so far (71.2 % and 68.6 %, respectively). Averaged over all four emotional dimensions, we obtained a (weighted) accuracy of 65.2 % via bidirectional LSTM modeling of acoustic features, which is higher than all other average accuracies reported for this task in literature up to now. The absolute values of the reported accuracies seem low in comparison to easier scenarios, such as the discrimination of acted, prototypical emotions. However, the considered scenario reflects realistic conditions in natural interactions and thus highlights the need for further research in the area of affective computing in order to get closer to the human performance in judging emotions.

Future studies should consider to examine the potential of multi-task learning, i. e., learning phonemes and the affective state simultaneously. Furthermore, the context analysis method used in Section 4.2.2 to analyze BLSTM modeling of affective human-human conversations should also be applied to other databases and scenarios, such as human-computer interactions, human-robot dialogues, and call-center data. This could help to gain insights regarding the flexibility and adaptiveness of LSTM context modeling, as well as the characteristics of different emotion recognition use-cases. Future research in the area of video feature extraction should include

the application of multi-camera input to be more robust to head rotations. The facial movements captured by multiple 2D cameras can be combined to predict 3D movement via deformable 3D models. Concerning the 2011 Audio/Visual Emotion Challenge, it would be interesting to fuse the results of all challenge participants to make use of the potentially complementary information generated by the individual techniques. To obtain the best possible recognition performance, future studies should also investigate which feature-classifier combinations lead to the best results, e. g., by combining the LSTM framework outlined in Section 4.2.3 with other audio or video features proposed for the 2011 Audio/Visual Emotion Challenge.

In addition to the mentioned approaches for future improvements, there will be a lot more aspects to consider before emotion-sensitive systems show a degree of naturalness that is comparable to human-human communication. Yet, even though the amount of social competence an advanced emotion recognition framework can incorporate into a virtual agent remains limited and cannot fully compete with human affect recognition quality, the principle of integrating long-range context information can be seen as a further step towards making virtual agents more human-like.

5

Driving Behavior Analysis

So far, we have focussed on (mostly speech-based) verbal and non-verbal behavior analysis in human-computer interaction scenarios. This chapter shows that the context-sensitive machine learning techniques considered and advanced in this thesis can also be successfully applied in other domains of human behavior analysis that involve signals which strongly differ from the speech signals processed by algorithms and models introduced in Chapters 3 and 4. As pattern recognition is increasingly used in the automotive domain, we will now concentrate on driving behavior analysis and transfer the methods established in the field of affective computing (see Chapter 4) to driver distraction detection. Recognizing whether a driver is distracted or not plays an important role for the design of lane-keeping assistance systems which may be more acceptable to users if the assistance was adaptive to the driver's state. Thus, this chapter introduces a novel technique for on-line detection of driver's distraction, modeling the long-range temporal context of driving and head tracking data. Again, Long Short-Term Memory recurrent neural networks (see Section 2.3.9) are applied as an efficient technique to capture the dynamics of successive pattern vectors. Following a strategy similar to the emotion recognition engines proposed in Chapter 4, LSTM modeling is combined with large-scale feature functional computation by the openSMILE toolkit [73]. The following sections show that this approach enables a reliable, subject-independent detection of driver inattention with an accuracy of up to 96.6 %, outperforming conventional approaches such as Support Vector Machines [271].

5.1 Driver Distraction Detection

Driver inattention is one of the major factors in traffic accidents. The National Highway Traffic Safety Administration estimates that in 25 % of all crashes some form of inattention is involved [256]. Distraction (besides drowsiness), as one form of driver inattention, may be characterized as: "any activity that takes a driver's

attention away from the task of driving" [188]. Causes for driver inattention are for example the use of wireless devices or passenger related distractions [61]. Although over the last few years many European countries have prohibited, for instance, the use of wireless devices while driving, it should not be expected that the amount of distraction in driving will necessarily decrease. Even without the distractions caused by mobile devices, the amount of distraction due to in-car information systems will increase. Thus, original equipment manufacturers and automotive suppliers will need to find a way to deal with this problem.

One method that aims to minimize crashes rather than distractions is the development of new driver assistant systems [78, 242]. With the evolution of adequate lane tracking, lane-keeping assistance systems were introduced into the market recently. These systems track the lane markings in front of the vehicle and compute the time until the vehicle will cross the marking. If the driver does not show an intention of leaving the lane by using the indicator, the systems will use directed steering torques on the steering wheel to guide the car to the middle of the lane. Authors of several studies reported overall effects of lane departure warning systems on lane-keeping performance [2, 128, 194]. Even though different kinds of warnings can be helpful, participants in [2] judged the lane departure warning system to be annoying in some circumstances. The reason why those systems are annoying for some drivers is easy to explain. That is, lane-keeping assistance aims at preventing the driver from making unintended lane departures. However, these systems do not yet respond to the driver's state or his intent but to lane markings and the car's speed. This implies that warnings can be triggered if attentive drivers intentionally change lanes but forget to use the indicator or if certain maneuvers that are executed with full attention require lane crossings. Thus, if it was possible to recognize a driver's state reliably, the system would give just as much assistance as the driver needed. This would allow for a greater safety margin without irritating the driver with false alarms in normal driving situations.

In [25] three main approaches to such a recognition are discussed: monitoring of driver's perception, monitoring of driver steering and lane-keeping behavior, and the recognition of the driver's involvement in a secondary task itself. In recent years, several techniques trying to estimate the driver to be distracted have been published. However, the majority of approaches are developed and evaluated using data that was captured in a driving simulator and not in a real vehicle, where data is much more noisy and complex than it is in a simulator scenario [54, 140, 141, 246, 308]. A considerable number of studies concentrate on the detection and modeling of fatigue or stress as important causes for inattention (e. g., [63, 104, 117, 118]). However, as shown in [308], also visual distraction downgrades driving performance.

In order to detect distraction or inattention while driving, different classification techniques can be found in literature. The predominant approach is to use static classifiers such as Support Vector Machines [140, 142] (see Section 2.3.1). A promising approach can be found in [130] where SVM are used to detect driver dis-

traction based on data captured in real traffic conditions, resulting in accuracies of 65 - 80 %. In this study, features are computed from fixed-length time windows, i. e., the amount of context that is incorporated into the classification decision is predefined. In [141], the authors show that time-dependencies are highly relevant when predicting the current state of a driver: Modeling the *dynamics* of driver behavior by using a Dynamic Bayesian Network (see Section 2.3.2) rather than a static network led to accuracies of around 80 %. Similar approaches towards driver behavior or driver state estimation that model contextual information via DBNs or Markov models can also be found in [129] and [172]. Other popular classification strategies include the application of fuzzy logic [181], multiple adaptive regression trees [246], or neural networks [54, 63].

This section introduces a framework for on-line driver distraction detection based on modeling contextual information in driving and head tracking data captured during test drives in real traffic. Similar to the emotion recognition systems outlined in Chapter 4, the approach is based on Long Short-Term Memory RNNs, exploiting their ability to capture the long-range temporal evolution of data sequences (see Section 2.3.9). We investigate both, 'sample-wise' classification based on low-level signals and 'frame-wise' classification using statistical functionals of the signals. Evaluations in Section 5.1.3 show that using low-level signals for driver distraction detection is hardly feasible with conventional recurrent neural networks where the amount of accessible context information is limited.

5.1.1 Driving Data and Signals

In order to collect data that represents a distracted drivers' behavior in realistic driving situations, 30 participants (12 female and 18 male) were recruited [271]. The subjects were 23 to 59 years old and had driven at least 10 000 kilometers in the last 12 months. An Audi A6 was used as the experimental car. The car was equipped with the Audi Multimedia System (see Figure 5.1) and an interface to measure Controller Area Network (CAN)-Bus data. Additionally, a head tracking system [25] was installed, which was able to measure head position and head rotation. This data was also sent on CAN-Bus. Head tracking systems are not common in vehicles today, but promising research in systems for driver state detection will lead to a higher installation rate in serial cars in the near future.

Eight typical tasks on the Multimedia Interface were chosen as distraction conditions:

- radio: adjust the radio sound settings
- CD: skip to a specific song
- phonebook: search for a name in the phonebook

Figure 5.1: Audi A6 Cockpit.

- navigation-point of interest: search for a nearby gas station
- phone: dial a specific phone number
- navigation: enter a city in the navigation device
- TV: switch the TV mode to 'PAL'
- navigation-sound: adjust the volume of navigation announcements

These kinds of visual and manual distractions that are typical when operating in-vehicle information systems. Purely mental forms of distraction or inattention (such as 'being lost in thought') were excluded since they are comparably hard to elicit and detect. Also tasks leading to auditory distraction (e. g. talking to a passenger) were not included in the database as they are generally considered as lower-risk activities [305].

The main functions (e. g., navigation, CD/TV, and radio) are available through eight so-called hardkeys which are located on the right- and left-hand side of the control button (see Figure 5.1). In each main menu, special functions (e. g. sound settings in the radio menu) can be selected by the four so-called softkeys which surround the control button. These special functions differ between the main menus. The functions assigned to the softkeys are shown in the corners of the display which is located in the middle console. Most inputs are done using the control button. By turning the control button left or right it is possible to scroll up and down in lists while pushing the button selects highlighted items. For typing letters (navigation) or digits (phone) the so-called speller is used, whereas symbols are arranged in a circle and can be selected by turning and pushing the control button.

The procedure for the experiment was as follows: After a training to become familiar with the car each participant drove down the same country road eight times (one time per task) while performing secondary tasks on the in-vehicle information system. Each task was performed only once per drive and only the time from the

beginning of the task to the end of the task was recorded as a 'distracted drive'. On another two runs the drivers had to drive down the road with full attention on the roadway ('baseline' runs). In order to account for sequential effects, the order in which the conditions were presented was randomized for each participant. During each drive CAN-Bus data (including head tracking data) were logged. The experiments were performed on a German country road with an average road width of 3.37 m and continuous road marking. The route is straight (apart from two slight turns), consists of one lane per direction, and leads through a forest. During the experiments oncoming traffic was present, however, the overall traffic density was moderate. Participants drove during the daytime under different weather conditions (mostly dry). Overall, 53 runs while driving attentively and 220 runs while the drivers were distracted could be measured (some runs had to be excluded due to logging problems). The 'attentive' runs lasted 3 134.6 seconds altogether, while 9 145.8 seconds of 'distracted' driving were logged. Thus, the average duration of attentive and distracted runs was 59.2 seconds and 41.6 seconds, respectively. At an average speed of roughly 100 km/h, this corresponds to distances of 1.64 km and 1.16 km, respectively. An analysis of the influence on lane-keeping of the different in-vehicle information system interaction tasks [25] indicated that the tasks can be characterized as distracting in general.

Three different classification tasks are considered for the estimation of distraction: the binary decision whether a driver is distracted or not ('two-class problem'), the discrimination between no, medium, and a high degree of distraction ('three-class problem'), and the discrimination between six levels of distraction ('six-class problem'). For the binary problem, all tasks (i. e., runs during which the tasks were performed) were labeled as 'distracted' compared to driving down the road with full attention ('attentive'). Since all participants were asked to judge the level of distraction of a certain task (meaning the difficulty of the task) on a scale between 1 (easy) and 5 (difficult), these individual judgments were used to model also the *degree* of distraction as a six-class problem ('attentive' plus five levels of distraction). For the three-class problem, difficulties 1 to 3 as well as difficulties 4 and 5 were clustered together. Thus, the system for driver distraction detection is trained to predict the subjective ratings of distraction assigned by the participants using different levels of granularity. Even though the system outputs an estimate of the subjective level of distraction every few milliseconds, the level of distraction is defined *by drive*, meaning that we assign the same level of distraction to each time step of a certain drive. This has the effect that the classifier considers long-term context and predicts the driver state according to the overall difficulty of the task and the resulting level of distraction. It is assumed that during the 'distracted' runs the driver is continuously engaged in the task, even if there are short periods of attention which are of course necessary while driving. By characterizing distraction on a *per-drive* basis, these short intervals of attention are smoothed out in order to model the driver state on a long-term basis, which in turn is desired when using driver state estimations for

adaptive lane-keeping assistance. Six signals were chosen:

- steering wheel angle (SA)
- throttle position (TP)
- speed (SP)
- heading angle (HA, angle between the longitudinal axis of the vehicle and the tangent on the center line of the street)
- lateral deviation (LD, deviation of the center of the car from the middle of the traffic lane)
- head rotation (HR, rotation around the vertical axis of the car)

The first three (SA, TP, and SP) are direct indicators of the driver behavior. Many studies prove the fact that visually distracted drivers steer their car in a different way than attentive drivers do. The same applies for throttle use and speed (an overview can be found in [305]). The car's heading angle and its lateral deviation in the lane rely on the amount of attention the driver is allocating to the roadway and may hence give useful information about distraction. Head rotation of the driver is an indicator of the driver's visual focus [234]. While using the Multimedia Interface, which is located in the middle console just below the dashboard, the main rotation of the head is to the right. Thus, the heading angle of head rotation is the most promising indicator of the head tracking signals.

5.1.2 Distraction Detection from Driving and Head Tracking Data

The main architecture of the proposed system for driver distraction classification can be seen in Figure 5.2. In the following we will denote all signals prior to statistical functional computation as *low-level signals* with synchronized time index n (and time index n' prior to synchronization) whereas t is the frame index referring to the time windows over which statistical functionals are calculated. In Section 5.1.3 we investigate both, the direct modeling of low-level signals s_n (including the first and second derivatives) and the modeling of statistical functionals of those signals (x_t). In other words, we examine the performance of driver distraction detection with and without the processing unit represented by the dotted box in Figure 5.2.

A camera capturing the road in front of the vehicle provides a video signal $v^1_{n'}$ which is processed by the lane departure warning system to compute the current lateral deviation $s^{LD}_{n'}$ and heading angle $s^{HA}_{n'}$. The head rotation $s^{HR}_{n'}$ is determined by a head tracking system that processes the signal $v^2_{n'}$ recorded by a second camera facing the driver. Steering wheel angle $s^{SA}_{n'}$, throttle position $s^{TP}_{n'}$, and speed $s^{SP}_{n'}$ are

captured by the corresponding sensors and sent to the CAN-Bus together with $s_{n'}^{LD}$, $s_{n'}^{HA}$, and $s_{n'}^{HR}$.

The sample frequencies of the six signals represented by $s_{n'}^{c}$ range from 10 to 100 Hz. Thus, the data sequences are linearly intrapolated in order to obtain a uniform frequency of 100 Hz before being synchronized. From the resulting interpolated and synchronized signal vector s_n^i first and second order regression coefficients (i. e. first and second temporal derivatives δs_n^i and $\delta\delta s_n^i$) are calculated for every time step n and each component of the low-level signal vector s_n^i. Thus, together with δs_n^i and $\delta\delta s_n^i$, we have $3 \times 6 = 18$ low-level data sequences at this stage.

As mentioned before, an alternative to directly using the low-level signals $s_n = [s_n^i, \delta s_n^i, \delta\delta s_n^i]$ as inputs for LSTM-based driver state classification every 10 ms is to compute a set of statistical functionals over longer time windows and use those functionals x_t as a basis for classification. Here, t refers to the index of the frame which contains functionals extracted from a time window of three seconds. As frame rate 500 ms are used resulting in a frame overlap of 2.5 seconds. Depending on whether or not this kind of framewise processing is used, either x_t or s_n is normalized to have zero mean and variance one. All means and variances are determined from the training set.

The normalized signals x_t' or s_n' are then used as inputs for the LSTM network, meaning that the individual components of the vectors x_t' / s_n' represent the activations of the input nodes of the network at a given time step n or frame t. Consequently, the LSTM network has as many input nodes as there are components in the vectors x_t' and s_n', respectively. The number of output nodes of the network corresponds to the number of distinct classes in the classification task. Three different classification tasks are investigated: the discrimination between two, three, and six different levels of distraction. Thus, the LSTM network has either two, three, or six output nodes. The activation of the output nodes o_t / o_n corresponds to the likelihood that the respective class (or distraction level) is observed at a given time step. To obtain an estimate z_t or z_n of the level of driver distraction at each frame or time step, we simply take the class corresponding to the maximum network output activation.

As mentioned before, two different strategies for driver distraction detection are considered: Firstly, the low-level signals, together with their first and second temporal derivatives are used for *samplewise* classification every 10 ms. Secondly, *frame-wise* classification is applied by computing statistical functionals every 500 ms from both, the low-level signals and their derivatives (55 functionals per input signal, see Tables 5.1 and 5.3) with one frame spanning three seconds. Temporal derivatives of the low-level signals are calculated according to the following formula:

$$\delta s_n^i = \frac{\sum_{d=1}^{D} d \cdot (s_{n+d}^i - s_{n-d}^i)}{2 \cdot \sum_{d=1}^{D} d^2}. \tag{5.1}$$

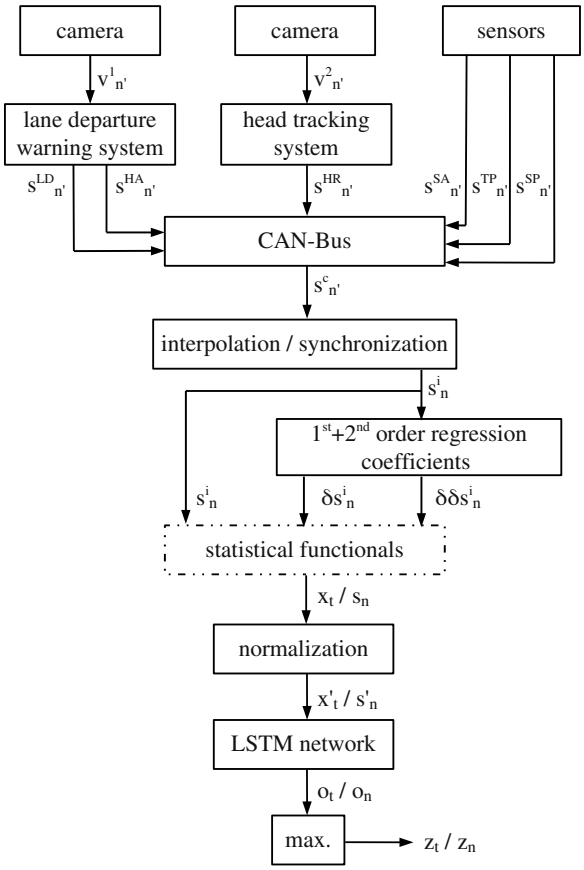


Figure 5.2: System architecture of the driver distraction detection system.

The parameter D is set to one. For the calculation of the second derivative $\delta\delta s_n^i$ we simply applied Equation 5.1 to δs_n^i. Applying the openSMILE toolkit [73], a set of 55 statistical functionals is computed for each of the 18 low-level signals as a basis for the framewise classification task. Thus, we obtain a 990-dimensional feature vector for each 500 ms frame.

Using the validation partitions (see Section 5.1.3), each, CFS feature selection is applied to these functionals in order to reduce the dimensionality of the feature space by focussing on the most relevant features [269]. Since the driver distraction estimation experiments are arranged in a 30-fold cyclic leave-one-driver-out cross-validation, feature selection has to be conducted 30 times for each classification task (two- three- and six-class problem). On average, 33.8 features are selected for a given classification task and fold (see Table 5.3). Insights into the usefulness of

Table 5.1: Statistical functionals grouped into categories with abbreviations as used in Table 5.2.

functionals	abbreviation
Extremes	
maximum, minimum	max, min
range (max - min)	range
distance between maximum and mean	distmax
distance between minimum and mean	distmin
Regression	
linear regression coefficients 1 and 2	lregc1/2
arithmetic mean of linear regression error	mlrege
quadratic mean of linear regression error	qmlrege
quadratic regression coefficients 1, 2, and 3	qregc1/2/3
arithmetic mean of quadratic regression error	mqrege
quadratic mean of quadratic regression error	qmqrege
Means	
arithmetic mean	mean
arithmetic mean of non-zero values	nzmean
arithmetic mean of absolute non-zero values	nzmeanabs
geometric mean of non-zero values	nzgmean
Percentiles	
quartiles 1, 2, and 3 (25 %, 50 %, and 75 %)	q1, q2, q3
interquartile range 1-2, 2-3, and 1-3	iqr1-2/2-3/1-3
Peaks	
mean of peaks	pkmean
distance between mean of peaks and mean	pkmmd
others	
number of non-zero values (normalized)	nnz
zero crossing rate	zcr
mean crossing rate	mcr

the computed signal-functional combinations can be gained by ranking the features according to the number of folds in which they are selected via CFS. Such a ranking can be found in Table 5.2 where the 30 most frequently selected features are listed for each classification task. As assumed, functionals computed from the head rotation signal provide the most reliable features for the detection of driver distraction caused by the operation of the Multimedia Interface. According to Table 5.2, several different functionals such as minimum, mean, distance between the mean of the peaks and the mean, quartiles, interquartile ranges, or linear and quadratic regression coefficients are suited to extract useful information from the head rotation signal. Other frequently selected features are based on the second temporal derivative of the steering wheel angle ($\delta\delta$SA). This indicates that sudden abrupt movements of the steering wheel – which are necessary to correct the orientation of the car in case the driver does not continuously focus on the street – are a good indicator for distraction. Features computed from the heading angle are mostly selected for the two-class problem and seem less relevant as soon as a finer level of granu-

Table 5.2: Ranking of the 30 most frequently selected signal-functional combinations for the discrimination of two, three, and six levels of distraction. Numbers display the number of folds in which the corresponding feature was selected via CFS. δ and $\delta\delta$ indicate first and second temporal derivatives, respectively. Abbreviations in capital letters indicate the underlying low-level signal: steering wheel angle (SA), throttle position (TP), speed (SP), heading angle (HA), lateral deviation (LD), or head rotation (HR). Abbreviations in lower case letters represent the functionals (see Table 5.1).

2 classes		**3 classes**		**6 classes**	
feature	#	**feature**	#	**feature**	#
HR-min	30	HR-min	30	HR-min	30
HR-pkmmd	30	HR-pkmmd	30	SA-max	30
HR-q1	30	HR-q1	30	HR-q1	30
HR-iqr1-2	30	HR-iqr1-2	30	HR-iqr1-2	30
HR-iqr2-3	30	HR-iqr2-3	30	HR-iqr2-3	30
HR-iqr1-3	30	HR-iqr1-3	30	$\delta\delta$SA-max	30
HR-lregc2	30	HR-lregc2	30	$\delta\delta$SA-min	30
HR-qregc3	30	HR-qregc3	30	HR-mqrege	30
HR-mqrege	30	HR-mqrege	30	SA-min	29
$\delta\delta$SA-nzgmean	30	$\delta\delta$SA-nzgmean	30	$\delta\delta$SA-nzgmean	29
LD-max	28	LD-max	30	HR-iqr1-3	29
HR-q2	27	HR-mlrege	30	HR-lregc2	29
HR-mlrege	26	HR-q2	29	SP-pkmean	29
$\delta\delta$SA-distmax	26	$\delta\delta$SA-min	29	HR-q2	28
HR-mcr	23	$\delta\delta$SA-pkmean	29	HR-mlrege	28
$\delta\delta$SA-pkmmd	23	$\delta\delta$SA-pkmmd	29	HR-qregc3	28
HR-pkmean	22	SA-min	29	$\delta\delta$SA-pkmmd	27
δHR-nzgmean	22	HR-mcr	28	SA-pkmean	26
δHA-pkmean	20	HR-qmqrege	28	HR-mcr	24
HR-qmqrege	19	δHR-nzgmean	28	δHR-nzgmean	24
$\delta\delta$SA-distmin	19	HR-nzmean	25	$\delta\delta$LD-min	24
HR-nzmean	18	SA-max	24	LD-max	23
HR-distmin	17	SP-pkmean	24	δLD-min	23
HA-nzmeanabs	16	SA-pkmean	23	HR-qmqrege	22
HR-qmlrege	16	HR-pkmean	23	δSA-min	22
$\delta\delta$SA-pkmean	16	HR-distmin	22	δSA-max	20
$\delta\delta$SA-range	14	HR-mean	21	δLD-max	19
HR-mean	13	HR-qmlrege	21	$\delta\delta$SA-range	19
$\delta\delta$SA-zcr	13	$\delta\delta$SA-max	21	HR-mean	18
SA-max	12	$\delta\delta$SA-nnz	21	HR-nzmean	18

larity is to be modeled for driver state estimation. By contrast, features based on the lateral deviation signal tend to be rather suited for the six-class task: Four out of the 30 most frequently selected features are based on the lateral deviation when modeling six classes, whereas for the two- and three-class task only the maximum lateral deviation (LD-max) is frequently selected. Speed and throttle position are only rarely selected as can also be seen in Table 5.3.

Table 5.3: Left-hand side: functional categories and number of calculated functionals per data stream (each stream consists of the low-level signal, first, and second order regression coefficients); right-hand side: average number of features selected via CFS for the individual data streams: steering wheel angle (SA), throttle position (TP), speed (SP), heading angle (HA), lateral deviation (LD), and head rotation (HR). All numbers are averaged over all 30 leave-one-subject-out folds and all classification tasks.

number of functionals		**average number of selected features**						
type	**total**	**SA**	**TP**	**SP**	**HA**	**LD**	**HR**	**total**
Extremes	3×7	3.4	0.5	0.3	0.5	1.0	1.7	**7.4**
Regression	3×9	0.1	0.1	0.6	0.1	0.2	5.6	**6.7**
Means	3×7	2.3	0.1	0.1	1.2	0.0	2.6	**6.3**
Percentiles	3×6	0.1	0.0	0.3	0.1	0.6	5.0	**6.2**
Peaks	3×4	1.9	0.2	0.4	0.7	0.2	1.7	**5.1**
others	3×22	0.6	0.1	0.1	0.1	0.1	1.1	**2.0**
sum	**3×55**	**8.4**	**1.1**	**1.8**	**2.7**	**2.0**	**17.8**	**33.8**

5.1.3 Evaluation and Discussion

For all driver distraction detection experiments in this section, a driver independent cross-validation approach was used. The number of folds was equal to the number of drivers in the database (see Section 5.1.1). In each fold the test set consisted of a single driver (that is, all runs recorded for this person; up to two baseline runs and eight runs with task) while six other drivers were chosen randomly to form a validation set (containing nine to twelve baseline runs and 41 to 47 runs with tasks). The data of the remaining persons made up the training set (39 to 42 baseline runs, 166 to 172 runs with task).

Three different class distributions were evaluated. In each of these distributions, the baseline runs are treated as a single class. The runs with distracting tasks either make up another single class (two-class problem) or are split into two or five classes, based upon the individual, subjective rating of the difficulty of the respective task (three-class and six-class problem). In case of the three-class problem, one class consists of all runs rated with difficulties one to three (easy to medium), another one of all runs with difficulties four or five (difficult). In the six-class problem, each class corresponds to a single level of difficulty.

In order to investigate the effect of long-range contextual information modeling by using a hidden layer with LSTM architecture (i. e. using memory blocks instead of hidden cells, see Section 2.3.9), both, LSTM networks and conventional RNNs were trained and evaluated using the same configuration. LSTMs and RNNs had an input layer with as many nodes as there are features and a hidden layer with 100 memory blocks or neurons, respectively. Each memory block consisted of one cell. The number of output nodes is equal to the number of classes. Each network

Table 5.4: Classification of driver distraction using LSTM networks, standard RNNs, and SVMs that process either low-level signals with first and second order regression coefficients or statistical functionals of the signals and regression coefficients: accuracy (acc.), unweighted recall (rec.), unweighted precision (prec.), and F1-measure (F1) for the subject-independent discrimination of two, three, and six levels of distraction.

classifier	features	acc.	rec.	prec.	F1
	two classes				
LSTM	low-level signals	91.6	89.7	90.8	90.1
LSTM	functionals	96.6	95.0	97.2	**96.0**
RNN	low-level signals	74.6	60.0	68.3	63.2
RNN	functionals	94.9	92.9	95.0	93.8
SVM	functionals	91.8	88.0	90.6	89.1
	three classes				
LSTM	low-level signals	54.4	62.1	63.0	62.0
LSTM	functionals	60.4	70.2	70.1	**70.1**
RNN	low-level signals	42.1	46.6	46.4	45.6
RNN	functionals	62.5	67.9	65.7	66.5
SVM	functionals	61.6	65.8	64.6	64.9
	six classes				
LSTM	low-level signals	43.3	39.0	38.7	38.1
LSTM	functionals	45.4	42.6	41.0	**40.7**
RNN	low-level signals	37.8	30.9	30.6	29.5
RNN	functionals	44.7	41.4	36.4	38.0
SVM	functionals	43.5	39.2	35.2	36.7

is trained for up to 50 training iterations, applying an early stopping method. That is, training is instantly terminated if no improvement on the validation set could be achieved within the last ten iterations. To improve generalization, zero mean Gaussian noise with standard deviation 0.4 was added to the inputs during training. The networks were trained with on-line gradient descent, using a learning rate of 10^{-5} and a momentum of 0.9.

For comparison, all experiments employing the computed functionals as input data were repeated using Support Vector Machines with Sequential Minimal Optimization. The best results were achieved with a radial basis function as kernel. Table 5.4 shows the results for samplewise classification of driver distraction every 10 ms using the low-level signals together with regression coefficients and for classification every 500 ms applying functionals computed over 3000 ms time windows. Note that due to the imbalance in the class distribution, the F1-measure (harmonic mean of precision and recall) is a more adequate performance measure than accuracy. When using the low-level data, LSTM networks achieve an average F1-measure of 90.1 % for the two-class task and clearly outperform standard RNNs (63.2 %). The major reason for this is the inability of standard RNNs to model long-range time dependencies, which in turn is essential when using the low-level signal as a basis for

samplewise classification. When applying statistical functionals, the temporal evolution of the data streams is captured by the features (to a certain extent), leading to an acceptable performance of RNNs and SVMs (93.8 % and 89.1 %, respectively). Still, the best F1-measure is obtained with LSTM networks (96.0 %). The same holds for the three- and six-class problem, where Long Short-Term Memory modeling leads to an F1-measure of 70.1 % and 40.7 %, respectively, which is remarkable when considering that the participants' ratings of the level of distraction are highly subjective. The performance gap between SVM and LSTM classification can most likely be attributed to the fact that LSTM networks are able to model a flexible and self-learned amount of contextual information which seems to be beneficial for driver state estimation, while the context that is modeled by SVMs is limited to 3000 ms and is exclusively captured by the *features* via statistical functionals and not by the *classifier*.

5.2 Summary and Outlook

This section introduced a technique for on-line driver distraction detection that uses Long Short-Term Memory recurrent neural nets to continuously predict the driver's state based on driving and head tracking data. The considered recognition framework is able to model the long-range temporal evolution of either low-level signals or statistical functionals in order to reliably detect inattention, and can be seen as a basis for adaptive lane-keeping assistance. Experiments in Section 5.1.3 revealed that the proposed technique detects inattention with an accuracy of up to 96.6 %, corresponding to an F1-measure of 96.0 %. LSTM modeling prevails over conventional RNN networks and Support Vector Machines. From this point of view, an adaption of lane-keeping assistance systems which is based on driver state estimation seems to be a viable and promising approach.

In spite of the high accuracies obtained when operating the proposed driver distraction detection system in defined conditions, such as driving down a relatively straight country road or highway, the output of driver state estimation will of course be less accurate as soon as the driving behavior gets more complex, as for example when changing lanes or turning while driving in a city. Thus, a system for distraction detection as the one presented in this chapter can only be used if the current driving scenario roughly matches the training data, as it would be the case for most country roads. Similarly, a strong mismatch between the distraction characteristics observed during training and other potential sources of distraction that are not covered by the evaluation experiments might degrade the system performance and limit the applicability of distraction detection. However, even though negative performance offsets have to be expected under some circumstances and will, e. g., justify the additional usage of GPS information as a further indicator of when to activate and deactivate lane-keeping assistance, the experiments show that modeling contextual

information is beneficial for driver distraction detection and that the principle of Long Short-Term Memory is an elegant way to cope with this finding.

Future experiments should include the incorporation of bidirectional context for incremental refinement of driver state estimations. Moreover, alternative network topologies such as the bottleneck architecture (see Section 4.1.3) should be considered to gain further improvements.

6

Summary

The aim of this thesis was to create and evaluate novel machine learning architectures in order to enhance the accuracy of systems for automatic verbal and non-verbal behavior analysis. Such systems can be applied for speech and affective state recognition, e. g., within conversational agents to enable natural, emotion-sensitive human-machine interaction. An important requirement for human behavior recognition techniques that are designed for real-life application is robustness with respect to various challenging but realistic conditions such as conversational, spontaneous, disfluent, and emotional speaking styles, reverberation and background noise, as well as non-prototypical ambiguous emotions as they typically occur in natural interactions. To cope with these challenges, powerful modeling architectures are needed, which motivates the transfer of effective solutions developed by the machine learning community to the domain of intelligent human behavior analysis. One key strategy to improve verbal and non-verbal behavior analysis is the efficient exploitation of contextual information. Thus, the focus of this thesis was on context-sensitive machine learning techniques such as Long Short-Term Memory RNNs [84, 93, 111] which allow for enhanced long-range temporal context modeling within neural networks. To enable the best possible recognition performance for various behavior analysis tasks, including keyword detection, continuous speech recognition, (audio-visual) emotion recognition, interest recognition, etc., this thesis showed how the LSTM principle can be combined with front-ends supporting large-scale on-line speech feature extraction [73], statistical functional computation, Non-Negative Matrix Factorization [261], and facial movement feature extraction [284] and with recognition back-ends comprising Dynamic Bayesian Networks [278], discriminative learning strategies [123], Connectionist Temporal Classification [90], and multi-stream models [281].

After an introduction of the theoretical background the developed human behavior analysis components are based on (Chapter 2), the goal of Chapter 3 was to advance the state-of-the-art in keyword spotting, continuous speech recognition, and noise robust speech recognition – i. e., *verbal* behavior analysis. Five different keyword detection techniques based on discriminative learning, hierarchical Graphical

Models, and Long Short-Term Memory were proposed and evaluated [273, 275, 278, 280, 293]. Various experiments showed that the integration of phoneme modeling via LSTM networks increases keyword detection accuracies. For read speech, best performance could be obtained with the Tandem CTC-DBN outlined in Section 3.1.5 while for spontaneous speech a combination of discriminative keyword spotting and bidirectional Long Short-Term Memory as proposed in Section 3.1.1 led to the best results. Next, different methods to integrate BLSTM context-modeling into HMM systems for conversational speech recognition were investigated [279, 281, 291, 296]. Evaluations on the COSINE database and on the Buckeye corpus revealed that the most effective technique for BLSTM-based continuous recognition of spontaneous speech is the Bottleneck-BLSTM front-end introduced in Section 3.2.4. It unites the principles of LSTM context exploitation, bottleneck networks, and bidirectional speech modeling and increases word accuracies by 6.6 % and 7.2 % (absolute) on the COSINE and the Buckeye task, respectively, when compared to a standard MFCC-HMM system. Finally, various approaches towards enhancing the noise robustness of ASR systems were examined, including Switching Linear Dynamic Models [286], multi-condition training [287], Tandem BLSTM-HMM systems [298], and Non-Negative Matrix Factorization [301]. Impressive noisy speech recognition accuracies could be obtained with a novel triple-stream system [301] featuring NMF speech enhancement [260], Non-Negative Sparse Classification [81], and multi-stream BLSTM-HMM modeling [281]: On the CHiME Challenge 2011 task [39], which consists in noisy speech recognition in multisource environments, the system introduced in Section 3.3.4 achieved an average word accuracy of 91.86 % which is the best result reported for the challenge task so far [301].

In Chapter 4, we concentrated on *non-verbal* behavior analysis, meaning recognition of paralinguistic and affective states (emotion, interest, etc.). Similar to Chapter 3, a major goal was to improve emotion recognition performance in challenging conditions via appropriate LSTM context modeling. Extensive experiments were devoted to the investigation of different aspects of affect modeling and recognition from speech, including emotion representation (continuous emotional dimensions vs. clusters in the emotional space), the unit of analysis (frame- vs. turn-level), the gain of incorporating linguistic information in addition to acoustic features, and the effect of context modeling. It was shown that (B)LSTM-based recognition engines prevail over systems that use no or limited contextual information, i. e., SVMs, SVR, or conventional recurrent neural networks [294]. Similar observations could be made for the task of automatically estimating a user's level of interest from acoustic and linguistic features: BLSTM modeling as used within the recognition engine introduced in Section 4.1.3 led to the best interest recognition results ever reported for the Interspeech 2010 Paralinguistic Challenge task [299, 300]. The proposed context-sensitive speech-based emotion recognition framework can be extended to a multi-modal system processing speech and facial marker information for enhanced, audio-visual emotion recognition as shown in Section 4.2.1. Again, BLSTM mod-

eling of statistical functionals of low-level voice and face features outperformed alternative approaches using RNNs, BRNNs, SVMs, or HMMs [161, 289]. In Section 4.2.2, a methodology to investigate the *amount* of context information used within an audio-visual BLSTM emotion recognition system was explained [290]. Finally, a fully automatic emotion recognition system exploiting acoustic, linguistic, facial movement, and BLSTM context information was proposed in Section 4.2.3. The system achieved the best average recognition accuracy that has been reported so far for the Audiovisual Sub-Challenge of the 2011 Audio/Visual Emotion Challenge [284].

To illustrate that the proposed methodology used for affective computing can be successfully transferred to other pattern recognition disciplines which are not based on speech signal processing but on completely different time-continuous signals, a system using segment-wise statistical functional computation via the openSMILE toolkit [73] as well as context-sensitive sequence labeling with LSTM networks was created for the task of driver distraction detection from head-tracking and driving data (Chapter 5). The system was able to detect driver distraction with an accuracy of up to 96.6 % and outperforms methods applying RNNs or SVMs [271].

In summary, it can be observed that replacing or enhancing widely-used static or dynamic machine learning techniques such as SVMs or HMMs with advanced context-sensitive techniques like LSTM networks seems beneficial for a wide range of different pattern recognition disciplines in which the consideration of past (and possibly future) temporal context helps to infer the class encoded in the current observation. The dynamics or the temporal evolution of observed feature vectors that are part of a continuous stream of data plays an essential role in all of the human behavior analysis tasks considered in this thesis. In speech recognition, context in the form of language information / word transition likelihoods, co-articulation effects, and phoneme or phoneme state transitions has to be exploited to reach acceptable recognition accuracies. Similarly, when designing emotion recognition engines that continuously predict the user's emotional state, e. g., during a conversation, the temporal evolution of affect has to be modeled to reliably assess the user's emotion. The LSTM architecture is a very effective technique for modeling long-range context and to learn the amount of relevant context from training data. Even though the recognition systems and experiments detailed in Chapters 3 to 5 cover only selected pattern recognition tasks, the results show that LSTM-based sequence modeling clearly outperforms state-of-the-art techniques as the outlined recognition frameworks led to the best results in various international research challenges (Interspeech 2010 Paralinguistic Challenge [220], 2011 PASCAL CHiME Challenge [39], 2011 Audio/Visual Emotion Challenge [221]) [284, 299, 301]. This should motivate researchers working on various related pattern recognition systems to transfer the proposed context-sensitive machine learning approaches and model architectures to their domains in order to benefit from the findings contained in this thesis.

In addition to the application of the proposed techniques for other tasks and

scenarios, there are a lot of thinkable and promising possibilities to enhance human behavior analysis in the future. For example, in the field of speech recognition, future studies should consider language modeling with LSTM networks. Context-sensitive neural networks might also be employed for speech feature enhancement, e. g., by training a Regression-LSTM network to map from noisy speech feature vectors to clean features. Furthermore, it seems promising to investigate CTC-based phoneme modeling also for large-vocabulary continuous speech recognition. Finally, context-sensitive multi-task learning of phonemes and emotional or paralinguistic speaker states or traits could increase the overall recognition performance compared to modeling phonemes only. An interesting approach towards further improving emotion recognition accuracies is to combine different front- end back-ends, e. g., as proposed by the various participants of the 2011 Audio/Visual Emotion Challenge. This allows for investigations concerning the complementarity of different ideas to enhance features and recognition engines for the automatic assessment of human affect from speech and video information. When applying context-sensitive emotion recognition systems, it might also be possible to move from time-continuous emotion recognition to emotion detection (or 'spotting') and to use, e. g., Graphical Model architectures developed for keyword spotting for the task of emotion detection.

Acronyms

3D-DTW ..... three-dimensional Dynamic Time Warping
ACF ..... autocorrelation function
AER ..... automatic emotion recognition
AFE ..... advanced front-end feature extraction
AHMM ..... Asynchonous Hidden Markov Model
ANN ..... artificial neural network
AR-SLDS ..... Autoregressive Switching Linear Dynamical Systems
ASR ..... automatic speech recognition
AUC ..... area under the ROC curve
BLSTM ..... bidirectional Long Short-Term Memory
BN ..... Bayesian Network
BoW ..... Bag of Words
BRIR ..... binaural room impulse response
BRNN ..... bidirectional recurrent neural network
CAN ..... Controller Area Network
CC ..... cross correlation
CFS ..... Correlation-based Feature Subset selection
CMS ..... Cepstral Mean Subtraction
CPF ..... conditional probability function
CT ..... close-talk
CTC ..... Connectionist Temporal Classification
CTRV ..... close-talk reverberated
DBN ..... Dynamic Bayesian Network
DCT ..... Discrete Cosine Transform
DISC ..... discriminative keyword spotter
DTW ..... Dynamic Time Warping
ECA ..... Embodied Conversational Agent
ELM-NN ..... extreme learning machine based feedforward neural network
EM ..... expectation maximization

FER ..... framewise phoneme error rate
FIR ..... finite impulse response
FPA ..... framewise phoneme accuracy
fpr ..... false positive rate
GM ..... Graphical Model
GMM ..... Gaussian Mixture Model
GPB ..... generalized pseudo-Bayesian
HA ..... heading angle
HCRF ..... Hidden Conditional Random Fields
HEQ ..... Histogram Equalization
HMM ..... Hidden Markov Model
HNR ..... Harmonics-to-Noise Ratio
HR ..... head rotation
HSV ..... hue-saturation-value
IDL ..... idle / neutral emotional state
IEMOCAP ..... Interactive Emotional Dyadic Motion Capture
ILA ..... inter-labeler agreement
lAUC ..... local AUC
LD ..... lateral deviation
LDCRF ..... Latent-Dynamic Conditional Random Fields
LDM ..... Linear Dynamic Model
LF ..... late fusion
LLD ..... low-level descriptor
LM ..... language model
LOI ..... level of interest
LOSO ..... leave-one-speaker-out
LPC ..... Linear Prediction Coding
LSTM ..... Long Short-Term Memory
LVCSR ..... large vocabulary continuous speech recognition
MAP ..... maximum a priori
MCT ..... multi-condition training
MDDTW ..... Multi-Dimensional Dynamic Time Warping
MFB ..... Mel-frequency bands
MFCC ..... Mel-Frequency Cepstral Coefficients
ML ..... maximum likelihood
MLE ..... mean linear error
MLP ..... multilayer perceptron
MMSE ..... Minimum Mean Square Error
MVN ..... Mean and Variance Normalization
NEG ..... negative valence
NMF ..... Non-Negative Matrix Factorization
NSC ..... Non-Negative Sparse Classification

OOV ..... out-of-vocabulary
PCA ..... principal componenet analysis
PFA ..... principal feature analysis
PLP ..... Perceptual Linear Prediction
RBF ..... radial basis function
RM ..... room microphone
RNN ..... recurrent neural network
ROC ..... Receiver Operating Characteristics
rProp ..... Resilient Propagation
SA ..... steering wheel angle
SAL ..... Sensitive Artificial Listener
SAR-HMM ..... Switching Autoregressive Hidden Markov Models
SLDM ..... Switching Linear Dynamic Model
SMO ..... Sequential Minimal Optimization
SMOTE ..... Synthetic Minority Oversampling Technique
SNR ..... signal to noise ratio
SP ..... speed
SVM ..... Support Vector Machine
SVR ..... Support Vector Regression
TP ..... throttle position
tpr ..... true positive rate
UA ..... unweighted accuracy
USS ..... Unsupervised Spectral Subtraction
VAM ..... Vera am Mittag
WA ..... word accuracy
WA ..... weighted accuracy

List of Symbols

Acoustic Feature Extraction

A_i ..... peak amplitude in the i^{th} time window
ACF_k^s ..... autocorrelation function of s_n
ACF_k^w ..... autocorrelation function of the window function
c_i ..... Mel-frequency cepstral coefficient
d ..... index for the computation of temporal derivatives
D ..... parameter for the computation of temporal derivatives
E ..... short-time energy of a speech signal frame
F_0 ..... fundamental frequency
f ..... frequency
f_s ..... sampling frequency
i ..... MFCC index
j ..... index counting the log filterbank amplitudes
J ..... jimmer
k ..... pre-emphasis coefficient
k ..... time-shift variable for the computation of the autocorrelation function
m_j ..... log filterbank amplitude
$MEL(f)$ ..... Mel-scale
n ..... index of a speech sample within the speech signal frame $s_{1:N}$
N ..... total number of speech samples in the speech signal frame $s_{1:N}$
N_{FB} ..... number of filterbank channels
N_V ..... number of voiced frames
s_n ..... speech sample within the speech signal frame $s_{1:N}$
s_n^{pre} ..... sample of the pre-emphasized speech signal
s_n^{raw} ..... sample of the raw speech signal

S ..... shimmer
T_0 ..... period of a periodic signal
T_i ..... duration of a pitched period

General Classification
C_k ..... class with index k
$h(x)$ ..... classifier output for a pattern vector x
i ..... index counting the training samples
I ..... number of instances in the training set
k ..... index representing the class C_k
K ..... number of different classes
l ..... target label
S ..... training set
S' ..... test set
V ..... length of the label sequence $l_{1:V}$
w ..... set of adjustable classifier parameters
w' ..... optimal parameter vector according to MAP approximation
$w^\star$ ..... optimal parameter vector according to ML approximation
x ..... input pattern vector

Support Vector Machines
α_i ..... weights of SVM training samples
b ..... bias of SVM hyperplane
C ..... SVM error weighting parameter
$d(x)$ ..... distance between input and SVM hyperplane
$i^\star$ ..... index of the training sample with largest coefficient α_i
j ..... secondary index counting the SVM training samples
$k(x_i, x_j)$ ..... kernel function
μ ..... margin of separation for SVM classification
p ..... polynomial order
$\Phi(x_i)$ ..... non-linear transformation for SVM kernel trick
w ..... normal vector of SVM hyperplane
x_i^{SV} ..... support vectors
ξ_i ..... slack variable

Dynamic Bayesian Networks and Hidden Markov Models
a_{ij} ..... HMM state transition probability from state i to state j

$\alpha_s(t)$ ..... forward probability
$b_s(x_t)$ ..... HMM emission probability
$\beta_s(t)$ ..... backward probability
c_{sm} ..... weight of a Gaussian mixture component
L_{st} ..... likelihood of being in state s at time t
m ..... index counting Gaussian mixture components
M ..... number of Gaussian mixture components
μ ..... mean vector
N ..... number of random variables in a Bayesian Network
$\mathcal{N}(\cdot)$ ..... Gaussian distribution
$p(\cdot)$ ..... random conditional probability function
π_i ..... set of parents of a node x_i
$\phi_s(t)$ ..... maximum likelihood of observing vectors $x_{1:T}$ while being in state s at time t
S ..... number of emitting states
$s_{1:T}$ ..... state sequence
σ ..... standard deviation
Σ ..... covariance matrix
t ..... time step
T ..... length of the input vector sequence $x_{1:T}$
x_i ..... ith random variable in a Bayesian Network

Multimodal Data Fusion

$\alpha_{s,\tau}(t)$ ..... forward variable for AHMM
$d(i,t,\tau)$ ..... 3D-DTW distance
$D(i,t,\tau)$ ..... accumulated 3D-DTW distance
e_t ..... binary variable indicating whether an AHMM observation of stream $y_{1:T'}$ is emitted or not
ϵ_s ..... probability of emitting a secondary observation of the feature vector sequence $y_{1:T'}$ in state s
g ..... parameter weighting the modalities for bimodal classification
k ..... constant indicating the maximum stretching between AHMM observation streams
m ..... index representing feature vector component for second modality
M ..... number of feature vector components (second modality)
n ..... index representing feature vector component for first modality
N ..... number of feature vector components (first modality)
o_t ..... potentially bimodal observation of an AHMM

r_i . . . . . reference sequence for DTW
$r^A_{1:I}$ . . . . . 3D-DTW reference sequence corresponding to the first modality
$r^B_{1:I}$ . . . . . 3D-DTW reference sequence corresponding to the second modality
T' . . . . . length of the secondary input vector sequence $y_{1:T'}$
τ . . . . . secondary time variable indicating the alignment for AHMM modeling of bimodal data streams
y . . . . . secondary input pattern vector for bimodal inputs

Neural Networks and Connectionist Temporal Classification

α . . . . . filter coefficient for filtering the LSTM outputs
$\alpha_t(v)$ . . . . . CTC forward variable
α^h . . . . . activation of hidden unit h
α^{in}_t . . . . . activation of input gate at time t
α^{for}_t . . . . . activation of forget gate at time t
α^{out}_t . . . . . activation of output gate at time t
$\mathcal{B}(\cdot)$ . . . . . operator that removes first the repeated labels and then the blanks from the output sequence
β^h . . . . . activation of hidden unit h after applying the activation function
β^{in}_t . . . . . activation of input gate at time t after applying the activation function
β^{for}_t . . . . . activation of forget gate at time t after applying the activation function
β^{out}_t . . . . . activation of output gate at time t after applying the activation function
$\beta_t(v)$ . . . . . CTC backward variable
c . . . . . index counting the LSTM memory cells
C . . . . . number of LSTM memory cells
δ_{ij} . . . . . Kronecker delta
f_h . . . . . neural network activation function
f_i . . . . . LSTM activation function for inputs
f_g . . . . . LSTM activation function for gates
f_o . . . . . LSTM activation function for outputs
h . . . . . index referring to a specific hidden cell in a neural network
H^q . . . . . number of neurons in hidden layer q
I . . . . . number of input nodes in a neural network
η^{ij} . . . . . neural network weight from unit i to unit j
J . . . . . sequential Jacobian

k . . . . . index referring to a specific output unit in a neural network
$l'_{1:V'}$ . . . . . modified label sequence with the 'blank' label added to the beginning and end, and between each pair of labels
L . . . . . number of hidden layers in a neural network
$lab(l_{1:V}, k)$ . . . . . set of positions in $l_{1:V}$ where the label k occurs
m . . . . . ANN momentum parameter
n . . . . . number of stacked feature frames
o . . . . . output vector of a neural network
o_t^k . . . . . activation of RNN output unit k at time t
o_t^s . . . . . filtered LSTM output at time t
O . . . . . objective function
O^{CTC} . . . . . CTC objective function
q . . . . . index referring to a specific hidden layer in a neural network
r . . . . . ANN learning rate
s_t^c . . . . . state of an LSTM memory cell c at time t
σ . . . . . logistic sigmoid
v . . . . . index indicating position in label sequence $l'_{1:V'}$
V' . . . . . length of the modified label sequence $l'_{1:V'}$
$w(n)$ . . . . . ANN weight vector after the nth update
x' . . . . . stacked feature vector
$z_{1:T}$ . . . . . framewise label sequence for CTC

Keyword Detection and ASR Systems

a . . . . . parameter to adjust the trade-off between true positives and false positives
A_k . . . . . AUC for keyword k
α_i . . . . . parameter for update rule of ω
b_t . . . . . discrete phoneme prediction feature
c_τ . . . . . binary 'cut' variable used for decoding with a hybrid CTC-DBN
C^u . . . . . parameter contolling the aggressiveness of the update rule for ω
d_τ . . . . . binary random variable indicating a phoneme deletion
δ . . . . . keyword spotter confidence threshold
ϵ . . . . . phoneme end time
ϵ . . . . . number of feature frames that lie between the CTC outputs $l_{\tau-1}$ and l_τ
ϵ' . . . . . number of frames between that lie between the CTC outputsl_τ and $l_{\tau+1}$

ε_l . . . . . lower fpr boundary for calculation of the local AUC
ε_u . . . . . upper fpr boundary for calculation of the local AUC
f . . . . . keyword spotter
g_t . . . . . hidden 'garbage' variable
$h_q(x_{1:T})$ . . . . . output of the hierarchical phoneme classifier
i . . . . . iteration of update of ω
i_τ^q . . . . . binary random variable indicating a phoneme insertion
I . . . . . identity matrix
$\mathbb{I}_{\{\cdot\}}$ . . . . . indicator function
k . . . . . index referring to a specific keyword
$\mathcal{K}$ . . . . . lexicon of keywords
κ . . . . . phoneme start time
l_τ . . . . . CTC phoneme prediction
l_t^{sp} . . . . . spike indicator variable used for Tandem CTC-DBN keyword spotting
L . . . . . length of a phoneme sequence
λ_h . . . . . weight of the hierarchical phoneme classifier
λ_o . . . . . weight of the BLSTM phoneme classifier
λ . . . . . stream weight variable for multi-stream HMM
λ_1 . . . . . stream weight variable for MFCC feature stream
λ_2 . . . . . stream weight variable for BLSTM feature stream
λ_3 . . . . . stream weight variable for NSC feature stream
n . . . . . number of non-linear feature functions
n_{ij} . . . . . number of phoneme transitions from phoneme i to phoneme j
N . . . . . matrix containing the number of phoneme transitions
$o_q(x_{1:T})$ . . . . . output of the BLSTM phoneme classifier
ω . . . . . weight vector for the discriminative keyword spotter
P . . . . . phoneme bigram
P . . . . . number of different phonemes
$\mathcal{P}$ . . . . . phoneme inventory
$\{\phi_j\}_{j=1}^n$ . . . . . non-linear feature functions for discriminative keyword spotting
$\boldsymbol{\phi}$ . . . . . vector of feature functions
$q_{1:L}^k$ . . . . . phoneme sequence of length L
q_t^c . . . . . count variable determining the current position in the phoneme sequence
q_t . . . . . phoneme identity variable
q_t^{ps} . . . . . random vairable determining the position within the phoneme

q_t^{tr} ..... binary variable indicating a phoneme transition
s_t^{tr} ..... binary variable indicating a state transition
τ ..... phoneme counter variable
V ..... number of words in the vocabulary
w_t ..... word identity variable
w_t^{ps} ..... random vairable determining the position within the word
w_t^{tr} ..... binary variable indicating a word transition
$x_{1:T}^{+}$ ..... utterance containing the keyword k
$x_{1:T}^{-}$ ..... utterance not containing the keyword k
$\mathcal{X}$ ..... domain of all possible feature vectors
$\mathcal{X}_k^{+}$ ..... set of utterances containing the keyword k
$\mathcal{X}_k^{-}$ ..... set of utterances not containing the keyword k
ξ ..... floor value for phoneme transition matrix
y_t ..... joint feature vector consisting of MFCC and BLSTM observations

Non-Negative Matrix Factorization

d_β ..... β-divergence between the observed spectrogram V and the approximation Λ
H ..... non-negative activations for NMF
k ..... speaker index
M ..... spectral resolution of magnitude spectrogram V
n_t ..... word identity feature obtained via NSC
N ..... temporal resolution of magnitude spectrogram V
P ..... temporal resolution of base spectrogram X
R ..... number of base spectograms for NMF
$\mathbb{R}_+$ ..... positive real numbers
$T^{(s,k,w)}$ ..... concatenated magnitude spectra
V ..... matrix representing the magnitude spectogram
W ..... concatenated base spectograms X for NMF
X ..... base spectogram for NMF
Λ ..... approximation of magnitude spectogram V
$\omega^{(s,k,w)}$ ..... convolutive base of magnitude spectra

Facial Movement Feature Extraction

h ..... 'hue' dimension of the HSV color space
$I(x, y)$ ..... pixel in an image

M . . . . . histogram for face tracking
N . . . . . number of template pixels for histogram
$p_f(x, y)$ . . . . . probability of a facial pixel
s . . . . . ‘saturation’ dimension of the HSV color space
T . . . . . detected face region taken as a template
θ . . . . . tilt
v . . . . . ‘value’ dimension of the HSV color space
x . . . . . x-coordinate in an image
y . . . . . y-coordinate in an image

Driver Distraction Detection

n . . . . . index referring to a sample of the driving data sequence s_n
n' . . . . . index referring to a sample of the driving data sequence $s_{n'}$ prior to synchronization
s_n . . . . . low-level driving data signal
$s_{n'}^{LD}$ . . . . . lateral deviation signal
$s_{n'}^{HA}$ . . . . . heading angle signal
$s_{n'}^{HR}$ . . . . . head rotation signal
$s_{n'}^{SA}$ . . . . . steering wheel angle signal
$s_{n'}^{SP}$ . . . . . speed signal
$s_{n'}^{TP}$ . . . . . throttle position signal
$v_{n'}^{1}$ . . . . . (first) camera signal for driver distraction detection
$v_{n'}^{2}$ . . . . . (second) camera signal for driver distraction detection
z_t . . . . . output prediction of driver distraction detection system

Bibliography

[1] M. Al-Hames and G. Rigoll, "Reduced complexity and scaling for asynchronous HMMs in a bimodal input fusion application," in *Proc. of ICASSP*, Toulouse, Fance, 2006, pp. 757–760.

[2] T. Alkim, G. Bootsma, and S. Hoogendoorn, "Field operational test 'The assisted driver'," in *Proc. of Intelligent Vehicles Symposium*, Instanbul, Turkey, 2007, pp. 1198–1203.

[3] J. Allen, G. Ferguson, and A. Stent, "An architecture for more realistic conversational systems," in *Proc. of Intelligent User Interfaces*, Santa Fe, USA, 2001, pp. 1–8.

[4] E. André, "Natural language in multimedia/multimodal systems," in *Handbook of Computational Linguistics*, R. Mitkov, Ed. Oxford University Press, 2003, pp. 650–669.

[5] E. André and C. Pelachaud, "Interacting with embodied conversational agents," in *Speech technology*, F. Chen and K. Jokinen, Eds. Springer, New York, 2010, pp. 123–149.

[6] E. André, M. Rehm, W. Minker, and D. Bühler, "Endowing spoken language dialogue system with emotional intelligence," in *Affective Dialogue Systems*, E. André, L. Dybkjaer, W. Minker, and P. Heisterkamp, Eds. Springer, 2004, pp. 178–187.

[7] J. Ang, R. Dhillon, E. Shriberg, and A. Stolcke, "Prosody-based automatic detection of annoyance and frustration in human-computer dialog," in *Proc. of Interspeech*, Denver, Colorado, 2002, pp. 2037–2040.

[8] S. Arunachalam, D. Gould, E. Anderson, D. Byrd, and S. Narayanan, "Politeness and frustration language in child-machine interactions," in *Proc. of Eurospeech*, Aalborg, Denmark, 2001, pp. 2675–2678.

[9] Y. Bar-Shalom and X. R. Li, *Estimation and tracking: principles, techniques, and software.* Artech House, Norwood, MA, 1993.

[10] L. F. Barrett and E. A. Kensinger, "Context is routinely encoded during emotion perception," *Psychological Science*, vol. 21, pp. 595–599, 2010.

[11] A. Batliner, C. Hacker, S. Steidl, E. Nöth, S. D'Arcy, M. Russel, and M. Wong, "You stupid tin box - children interacting with the Aibo robot: a cross-linguistic emotional speech corpus," in *Proc. of LREC*, Lisbon, Portugal, 2004, pp. 171–174.

[12] A. Batliner, S. Steidl, and E. Nöth, "Releasing a thoroughly annotated and processed spontaneous emotional database: the FAU Aibo Emotion Corpus," in *Proc. of a Satellite Workshop of LREC 2008 on Corpora for Research on Emotion and Affect*, L. Devillers, J. C. Martin, R. Cowie, E. Douglas-Cowie, and A. Batliner, Eds., Marrakesh, 2008, pp. 28–31.

[13] A. Batliner, S. Steidl, B. Schuller, D. Seppi, K. Laskowski, T. Vogt, L. Devillers, L. Vidrascu, N. Amir, L. Kessous, and V. Aharonson, "Combining efforts for improving automatic classification of emotional user states," in *Proc. of the 5th Slovenian and 1st International Language Technologies Conference*, Ljubljana, Slovenia, 2006, pp. 240–245.

[14] A. Batliner, S. Steidl, B. Schuller, D. Seppi, T. Vogt, J. Wagner, L. Devillers, L. Vidrascu, V. Aharonson, and N. Amir, "Whodunnit - searching for the most important feature types signalling emotional user states in speech," *Computer Speech and Language, Special Issue on Affective Speech in real-life interactions*, vol. 25, no. 1, 2011.

[15] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *The Annals of Mathematical Statistics*, vol. 41, no. 1, pp. 164–171, 1970.

[16] Y. Benayed, D. Fohr, J. P. Haton, and G. Chollet, "Confidence measure for keyword spotting using support vector machines," in *Proc. of ICASSP*, Hong Kong, 2003, pp. 588–591.

[17] S. Bengio, "An asynchronous hidden markov model for audio-visual speech recognition," *Advances in NIPS 15*, pp. 1–8, 2003.

[18] S. Bengio, "Multimodal authentication using asynchronous HMMs," in *Proc. of AVBPA*, Guildford, UK, 2003, pp. 770–777.

[19] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[20] E. Bevacqua, E. de Sevin, C. Pelachaud, M. McRorie, and I. Sneddon, "Building credible agents: behaviour influenced by personality and emotional traits," in *Proc. of Kansei Engineering and Emotion Research*, Paris, France, 2010.

[21] J. A. Bilmes, "Graphical models and automatic speech recognition," in *Mathematical Foundations of Speech and Language Processing*, R. Rosenfeld, M. Ostendorf, S. Khudanpur, and M. Johnson, Eds. New York: Springer Verlag, 2003, pp. 191–246.

[22] J. A. Bilmes and C. Bartels, "Graphical model architectures for speech recognition," *IEEE Signal Processing Magazine*, vol. 22, no. 5, pp. 89–100, 2005.

[23] F. Biocca, J. Burgoon, C. Harms, and M. Stoner, "Criteria and scope conditions for a theory and measure of social presence," in *Presence 2001*, Philadelphia, USA, 2001.

[24] C. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[25] C. Blaschke, F. Breyer, B. Färber, J. Freyer, and R. Limbacher, "Driver distraction based lane-keeping assistance," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 12, no. 4, pp. 288–299, 2009.

[26] P. Boersma, "Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound," in *Proceedings of the Institute of Phonetic Sciences*, Amsterdam, 1993, vol. 17, pp. 97–110.

[27] G. R. Bradski, "Computer vision face tracking for use in a perceptual user interface," Intel Technology Journal, Tech. Rep. Q2, 1998.

[28] J. Breese and G. Ball, "Modeling emotional state and personality for conversational agents," Microsoft, Tech. Rep., 1998.

[29] F. Burkhardt, A. Paeschke, M. Rolfes, W. Sendlmeier, and B. Weiss, "A database of German emotional speech," in *Proc. of Interspeech*, Lisbon, Portugal, 2005, pp. 1517–1520.

[30] F. Burkhardt, M. van Ballegooy, R. Englert, and R. Huber, "An emotion-aware voice portal," in *Proc. of Electronic Speech Signal Processing ESSP*, Prague, Czech Republic, 2005, pp. 123–131.

[31] C. Busso, M. Bulut, C.-C. Lee, A.Kazemzadeh, E. Mower, S. Kim, J. Chang, S.Lee, and S.Narayanan, "IEMOCAP: interactive emotional dyadic motion capture database," *Language Resources and Evaluation*, vol. 42, pp. 335–359, 2008.

[32] L. Caponetti, C. A. Buscicchio, and G. Castellano, "Biologically inspired emotion recognition from speech," *EURASIP Journal on Advances in Signal Processing*, 2011.

[33] S. Casale, A. Russo, G. Scebba, and S. Serrano, "Speech emotion classification using machine learning algorithms," in *Proc. of the IEEE International Conference on Semantic Computing*, Santa Clara, California, 2008, pp. 158–165.

[34] J. Cassell, "Nudge nudge wink wink: elements of face-to-face conversation for embodied conversational agents," in *Embodied conversational agents*. MIT Press, 2000, pp. 1–27.

[35] J. Cassell, T. Bickmore, L. Campbell, H. Vilhjalmsson, and H. Yan, "Human conversation as a system framework: designing embodied conversational agents," in *Embodied conversational agents*. MIT Press, 2000, pp. 29–63.

[36] L. Cen, Z. L. Yu, and M. H. Dong, "Speech emotion recognition system based on L1 regularized linear regression and decision fusion," in *Proc. of First International Audio/Visual Emotion Challenge and Workshop (AVEC 2011) held in conjunction with ACII*, Memphis, Tennessee, USA, 2011, pp. 332–340.

[37] F. Charles, S. Lemercier, T. Vogt, N. Bee, M. Mancini, J. Urbain, M. Price, E. André, C. Pelachaud, and M. Cavazza, "Affective interaction narrative in the CALLAS project," in *Proc. of ICVS*, Saint-Malo, 2007, pp. 210–213.

[38] B. Chen, Q. Zhu, and N. Morgan, "Learning long-term temporal features in LVCSR using neural networks," in *Proc. of ICSLP*, Jeju, Korea, 2004, pp. 612–615.

[39] H. Christensen, J. Barker, N. Ma, and P. Green, "The CHiME corpus: a resource and a challenge for Computational Hearing in Multisource Environments," in *Proc. of Interspeech*, Makuhari, Japan, 2010, pp. 1918–1921.

[40] Z. J. Chuang and C. H. Wu, "Emotion recognition using acoustic features and textual content," in *Proc. of ICME*, Taipei, Taiwan, 2004, pp. 53–56.

[41] I. Cohen, N. Sebe, A. Garg, L. Chen, and T. Huang, "Facial expression recognition from video sequences: Temporal and static modeling," *Computer Vision and Image Understanding*, vol. 91, no. 1, pp. 160–187, 2003.

[42] M. Cooke, J. R. Hershey, and S. J. Rennie, "Monaural speech separation and recognition challenge," *Computer Speech and Language*, vol. 24, pp. 1–15, 2010.

[43] M. Cooke, J. Barker, S. Cunningham, and X. Shao, "An audio-visual corpus for speech perception and automatic speech recognition," *The Journal of the Acoustical Society of America*, vol. 120, no. 5, pp. 2421–2424, 2006.

[44] C. Cortes and M. Mohri, "Confidence intervals for the area under the ROC curve," in *Advances in Neural Information Processing Systems 17*, 2004.

[45] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[46] R. Cowie, E. Douglas-Cowie, B. Apolloni, J. Taylor, A. Romano, and W. Fellenz, "What a neural net needs to know about emotion words," in *Computational Intelligence and Applications*, N. Mastorakis, Ed., 1999, pp. 109–114.

[47] 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 *Proc. of the ISCA Workshop on Speech and Emotion*, Newcastle, Northern Ireland, UK, 2000, pp. 19–24.

[48] R. Cowie, E. Douglas-Cowie, J. G. Taylor, S. Ioannou, M. Wallace, and S. Kollias, "An intelligent system for facial emotion recognition," in *Proc. of ICME*, Amsterdam, The Netherlands, 2005, pp. 1–4.

[49] R. Cowie, E. Douglas-Cowie, N. Tsapatsoulis, G. Votsis, S. Kollias, W. Fellenz, and J. G. Taylor, "Emotion recognition in human-computer interaction," *IEEE Signal Processing Magazine*, vol. 18, no. 1, pp. 32–80, 2001.

[50] R. Cowie, "Describing the forms of emotional colouring that pervade everyday life," in *The Oxford Handbook of Philosophy of Emotion*, P. Goldie, Ed. Oxford University Press, 2010, pp. 63–94.

[51] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive aggressive algorithms," *Journal of Machine Learning Research*, vol. 7, pp. 551–585, 2006.

[52] A. Cruz, B. Bhanu, and S. Yang, "A psychologically-inspired match-score fusion model for video-based facial expression recognition," in *Proc. of First International Audio/Visual Emotion Challenge and Workshop (AVEC 2011) held in conjunction with ACII*, Memphis, Tennessee, USA, 2011, pp. 341–350.

[53] A. de la Torre, A. M. Peinado, J. C. Segura, J. L. Perez-Cordoba, M. C. Benitez, and A. J. Rubio, "Histogram equalization of speech representation for robust speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 3, pp. 355–366, 2005.

[54] D. de Waard, K. A. Brookhuis, and N. Hernandez-Gress, "The feasibility of detecting phone-use related driver distraction," *International Journal of Vehicle Design*, vol. 26, no. 1, pp. 85–95, 2001.

[55] O. Dekel, J. Keshet, and Y. Singer, "Online algorithm for hierarchical phoneme classification," in *Workshop on Multimodal Interaction and Related Machine Learning Algorithms*, Martigny, Switzerland, 2004, pp. 146–159.

[56] M. Delcroix, K. Kinoshita, T. Nakatani, S. Araki, A. Ogawa, T. Hori, S. Watanabe, M. Fujimoto, T. Yoshioka, T. Oba, Y. Kubo, M. Souden, S. J. Hahm, and A. Nakamura, "Speech recognition in the presence of highly non-stationary noise based on spatial, spectral and temporal speech/noise modeling combined with dynamic variance adaptation," in *Proc. of Machine Listening in Multisource Environments (CHiME 2011), satellite workshop of Interspeech 2011*, Florence, Italy, 2011, pp. 12–17.

[57] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of Royal Statistical Society Series B*, vol. 39, pp. 185–197, 1977.

[58] J. Deng, M. Bouchard, and T. H. Yeap, "Noisy speech feature estimation on the Aurora2 database using a switching linear dynamic model," *Journal of Multimedia*, vol. 2, no. 2, pp. 47–52, 2007.

[59] L. Devillers, L. Lamel, and I. Vasilescu, "Emotion detection in task-oriented spoken dialogs," in *Proc. of ICME*, Baltimore, USA, 2003, pp. 549–552.

[60] L. Devillers, L. Vidrascu, and L. Lamel, "Challenges in real-life emotion annotation and machine learning based detection," *Neural Networks*, vol. 18, no. 4, pp. 407–422, 2005.

[61] T. Dingus, S. Klauer, V. Neale, A. Petersen, S. Lee, J. Sudweeks, M. Perez, J. Hankey, D. Ramsey, S. Gupta, C. Bucher, Z. Doerzaph, J. Jermeland, and R. Knipling, "The 100-car naturalistic driving study, phase II - results of the 100-car field experiment," Transportation Research Board of the National Academies, Tech. Rep., 2006.

[62] G. R. Doddington and T. B. Schalk, "Speech recognition: turning theory to practice," *IEEE Spectrum*, pp. 26–32, September 1981.

[63] T. D'Orazio, M. Leo, C. Guaragnella, and A. Distante, "A visual approach for driver inattention detection," *Pattern Recognition*, vol. 40, no. 8, pp. 2341–2355, 2007.

[64] E. Douglas-Cowie, R. Cowie, I. Sneddon, C. Cox, O. Lowry, M. McRorie, J. C. Martin, L. Devillers, S. Abrilian, A. Batliner, N. Amir, and K. Karpouzis, "The HUMAINE database: addressing the collection and annotation of naturalistic and induced emotional data," in *Affective Computing and Intelligent Interaction*. Springer, 2007, vol. 4738/2007, pp. 488–500.

[65] J. Droppo and A. Acero, "Noise robust speech recognition with a switching linear dynamic model," in *Proc. of ICASSP*, Montreal, Canada, 2004, pp. 953–956.

[66] J. Droppo, L. Deng, and A. Acero, "A comparison of three non-linear observation models for noisy speech features," in *Proc. of Eurospeech*, Geneva, Switzerland, 2003, pp. 681–684.

[67] K. Dupuis and K. Pichora-Fuller, "Use of lexical and affective prosodic cues to emotion by younger and older adults," in *Proc. of Interspeech*, Antwerp, Belgium, 2007, pp. 2237–2240.

[68] C. Elliott, "The affective reasoner: A process model of emotions in a multi-agent system," Ph.D. dissertation, Northwestern University, 1992.

[69] I. S. Engberg, A. V. Hansen, O. Andersen, and P. Dalsgaard, "Design, recording and verification of a Danish emotional speech database," in *Proc. of Eurospeech*, Rhodes, 1997, pp. 1695–1698.

[70] Y. Ephraim and W. J. J. Roberts, "Revisiting autoregressive hidden Markov modeling of speech signals," *IEEE Signal Processing Letters*, vol. 12, pp. 166–169, 2005.

[71] ETSI ES 202 050 V1.1.5, *Speech processing, Transmission and Quality aspects (STQ); Distributed speech recognition; Advanced front-end feature extraction algorithm; Compression algorithms*, 2007.

[72] G. Evangelista, S. Marchand, M. Plumbley, and E. Vincent, "Sound source separation," in *DAFX - Digital Audio Effects, 2nd Edition*, U. Zölzer, Ed. Wiley, 2011.

[73] F. Eyben, M. Wöllmer, and B. Schuller, "openSMILE - the Munich versatile and fast open-source audio feature extractor," in *Proc. of ACM Multimedia*, Firenze, Italy, 2010, pp. 1459–1462.

[74] F. Eyben, M. Wöllmer, M. F. Valstar, H. Gunes, B. Schuller, and M. Pantic, "String-based audiovisual fusion of behavioural events for the assessment of dimensional affect," in *Proc. of FG*, Santa Barbara, CA, USA, 2011, pp. 322–329.

[75] S. Fernandez, A. Graves, and J. Schmidhuber, "An application of recurrent neural networks to discriminative keyword spotting," in *Proc. of ICANN*, Porto, Portugal, 2007, pp. 220–229.

[76] J. R. J. Fontaine, K. R. Scherer, E. B. Roesch, and P. Ellsworth, "The world of emotions is not two-dimensional," *Psychological science*, vol. 18, no. 2, pp. 1050 – 1057, 2007.

[77] G. D. Forney, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[78] R. Freymann, "The role of driver assistance systems in a future traffic scenario," in *Proc. of the 2006 IEEE International Conference on Control Applications*, Munich, Germany, 2006, pp. 2269–2274.

[79] R. Gajsek, J. Zibert, T. Justin, V. Struc, B. Vesnicer, and F. Mihelic, "Gender and Affect Recognition based on GMM and GMM-UBM modeling with relevance MAP estimation," in *Proc. of Interspeech*, Makuhari, Japan, 2010, pp. 2810–2813.

[80] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren, "DARPA TIMIT acoustic phonetic continuous speech corpus CDROM," 1993.

[81] J. Gemmeke, T. Virtanen, and A. Hurmalainen, "Exemplar-based sparse representations for noise robust automatic speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 2067–2080, 2011.

[82] J. Gemmeke, T. Virtanen, and A. Hurmalainen, "Exemplar-Based Speech Enhancement and its Application to Noise-Robust Automatic Speech Recognition," in *Proc. of CHiME Workshop*, Florence, Italy, 2011, pp. 53–57.

[83] M. Gerosa, S. Lee, D. Giuliani, and S. Narayanan, "Analyzing children's speech: an acoustic study of consonants and consonant-vowel transition," in *Proc. of ICASSP*, Toulouse, France, 2006, pp. 393–396.

[84] F. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[85] D. Giuliani and M. Gerosa, "Investigating recognition of children's speech," in *Proc. of ICASSP*, Hong Kong, 2003, pp. 137–140.

[86] 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 *Proc. of First International Audio/Visual Emotion Challenge and Workshop (AVEC 2011) held in conjunction with ACII*, Memphis, Tennessee, USA, 2011, pp. 359–368.

[87] D. Goddeau, E. Brill, J. R. Glass, C. Pao, M. Phillips, J. Polifroni, S. Seneff, and V. W. Zue, "Galaxy: A human-language interface to on-line travel information," in *Proc. of ICSLP*, Yokohama, Japan, 1994, pp. 707–710.

[88] B. Goertzel, K. Silverman, C. Hartley, S. Bugaj, and M. Ross, "The baby webmind project," in *Proc. of The Annual Conference of The Society for the Study of Artifcial Intelligence and the Simulation of Behaviour (AISB)*, 2000.

[89] A. Graves, "Supervised sequence labelling with recurrent neural networks," Ph.D. dissertation, Technische Universität München, 2008.

[90] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented data with recurrent neural networks," in *Proc. of ICML*, Pittsburgh, USA, 2006, pp. 369–376.

[91] A. Graves, S. Fernandez, and J. Schmidhuber, "Bidirectional LSTM networks for improved phoneme classification and recognition," in *Proc. of ICANN*, Warsaw, Poland, 2005, pp. 602–610.

[92] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 855–868, 2009.

[93] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

[94] F. Grezl and P. Fousek, "Optimizing bottle-neck features for LVCSR," in *Proc. of ICASSP*, Las Vegas, NV, 2008, pp. 4729–4732.

[95] F. Grezl, M. Karafiat, K. Stanislav, and J. Cernocky, "Probabilistic and bottleneck features for LVCSR of meetings," in *Proc. of ICASSP*, Honolulu, Hawaii, 2007, pp. 757–760.

[96] M. Grimm, K. Kroschel, and S. Narayanan, "Support vector regression for automatic recognition of spontaneous emotions in speech," in *Proc. of ICASSP*, Honolulu, Hawaii, 2007, pp. 1085–1088.

[97] M. Grimm, K. Kroschel, and S. Narayanan, "The vera am mittag german audio-visual emotional speech database," in *Proc. of ICME*, Hannover, Germany, 2008, pp. 865–868.

[98] H. Gunes and M. Pantic, "Dimensional emotion prediction from spontaneous head gestures for interaction with sensitive artificial listeners," in *Proc. of Intelligent Virtual Agents*, Philadelphia, USA, 2010, pp. 371–377.

[99] H. Gunes, B. Schuller, M. Pantic, and R. Cowie, "Emotion representation, analysis and synthesis in continuous space: A survey," in *Proc. of IEEE Conference on Face and Gesture Recognition*, Santa Barbara, CA, USA, 2011, pp. 827–834.

[100] J. Gustafson and K. Sjölander, "Voice transformations for improving children's speech recognition in a publicly available dialogue system," in *Proc. of ICSLP*, Denver, Colorado, 2002, pp. 297–300.

[101] M. Hall, "Correlation-based feature selection for machine learning," Ph.D. dissertation, University of Waikato, 1999.

[102] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," *SIGKDD Explorations*, vol. 11, no. 1, 2009.

[103] J. Hansen and S. Bou-Ghazale, "Getting started with SUSAS: A speech under simulated and actual stress database," in *Proc. of Eurospeech*, Rhodes, Greece, 1997, pp. 1743–1746.

[104] J. A. Healey and R. W. Picard, "Detecting stress during real-world driving tasks using physiological sensors," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 2, pp. 156–166, 2005.

[105] M. Helen and T. Virtanen, "Separation of drums from polyphonic music using non-negative matrix factorization and support vector machine," in *Proc. of EUSIPCO*, Antalya, Turkey, 2005.

[106] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *Journal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–1752, 1990.

[107] H. Hermansky, D. P. W. Ellis, and S. Sharma, "Tandem connectionist feature extraction for conventional HMM systems," in *Proc. of ICASSP*, Istanbul, Turkey, 2000, pp. 1635–1638.

[108] H. Hermansky and P. Fousek, "Multi-resolution RASTA filtering for TANDEM-based ASR," in *Proc. of European Conf. on Speech Communication and Technology*, Lisbon, Portugal, 2008, pp. 361–364.

[109] H. G. Hirsch and D. Pearce, "The AURORA experimental framework for the performance evaluations of speech recognition systems under noisy conditions," in *ISCA ITRW ASR2000: Automatic Speech Recognition: Challenges for the Next Millennium*, Paris, France, 2000.

[110] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," in *A Field Guide to Dynamical Recurrent Neural Networks*, S. C. Kremer and J. F. Kolen, Eds. IEEE Press, 2001, pp. 1–15.

[111] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[112] A. Hurmalainen, K. Mahkonen, J. F. Gemmeke, and T. Virtanen, "Exemplar-based Recognition of Speech in Highly Variable Noise," in *Proc. of Machine Listening in Multisource Environments (CHiME 2011), satellite workshop of Interspeech 2011*, Florence, Italy, 2011, pp. 1–5.

[113] F. Itakura, "Minimum prediction residual principle applied to speech recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 23, pp. 67–72, 1975.

[114] H. Jaeger, "The echo state approach to analyzing and training recurrent neural networks," Bremen: German National Research Center for Information Technology, Tech. Rep., 2001, (Tech. Rep. No. 148).

[115] F. V. Jensen, *An introduction to Bayesian Networks*. Springer, 1996.

[116] J. H. Jeon, R. Xia, and Y. Liu, "Level of interest sensing in spoken dialog using multi-level fusion of acoustic and lexical evidence," in *Proc. of Interspeech*, Makuhari, Japan, 2010, pp. 2802–2805.

[117] Q. Ji, P. Lan, and C. Looney, "A probabilistic framework for modeling and real-time monitoring human fatigue," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 36, no. 5, pp. 862–875, 2006.

[118] Q. Ji, Z. Zhu, and P. Lan, "Real-time nonintrusive monitoring and prediction of driver fatigue," *IEEE Transactions on Vehicle Technology*, vol. 53, no. 4, pp. 1052–1068, 2004.

[119] B. Jiang, M. F. Valstar, and M. Pantic, "Action unit detection using sparse appearance descriptors in space-time video volumes," in *Proc. of FG*, Santa Barbara, CA, USA, 2011, pp. 314–321.

[120] T. Joachims, "Text categorization with support vector machines: learning with many relevant features," in *Proc. of ECML*, Chemniz, Germany, 1998, pp. 137–142.

[121] J. Keshet, "Large margin algorithms for discriminative continuous speech recognition," Ph.D. dissertation, Hebrew University.

[122] J. Keshet, D. Grangier, and S. Bengio, "Discriminative keyword spotting," in *Proc. of NOLISP*, Paris, France, 2007, pp. 47–50.

[123] J. Keshet, D. Grangier, and S. Bengio, "Discriminative keyword spotting," *Speech Communication*, vol. 51, no. 4, pp. 317–329, 2009.

[124] H. Ketabdar, J. Vepa, S. Bengio, and H. Boulard, "Posterior based keyword spotting with a priori thresholds," in *IDAIP-RR*, 2006, pp. 1–8.

[125] D. S. Kim, S. Y. Lee, and R. M. Kil, "Auditory processing of speech signals for robust speech recognition in real-world noisy environments," *IEEE Transactions on Speech and Audio Processing*, vol. 7, 1999.

[126] J. Kim and E. André, "Emotion recognition based on physiological changes in listening music," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 12, pp. 2067–2083, 2008.

[127] J. C. Kim, H. Rao, and M. A. Clements, "Investigating the use of formant based features for detection of affective dimensions in speech," in *Proc. of First International Audio/Visual Emotion Challenge and Workshop (AVEC 2011) held in conjunction with ACII*, Memphis, Tennessee, USA, 2011, pp. 369–377.

[128] K. Kozak, J. Pohl, W. Birk, J. Greenberg, B. Artz, M. Blommer, L. Cathey, and R. Curry, "Evaluation of lane departure warnings for drowsy drivers," in *Proc. of Human Factors and Ergonomics Society 50th Annual Meeting*, San Francisco, USA, 2006.

[129] T. Kumagai and M. Akamatsu, "Prediction of human driving behavior using dynamic bayesian networks," *IEICE Transactions on Information Systems*, vol. E89D, no. 2, pp. 857–860, 2006.

[130] M. H. Kutila, M. Jokela, T. Mäkinen, J. Viitanen, G. Markkula, and T. W. Victor, "Driver cognitive distraction detection: Feature estimation and implementation," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 221, no. 9, pp. 1027–1040, 2007.

[131] N. Landwehr, M. Hall, and E. Frank, "Logistic Model Trees," *Machine Learning*, pp. 161–205, 2005.

[132] K. J. Lang, A. H. Waibel, and G. E. Hinton, "A time-delay neural network architecture for isolated word recognition," *Neural Networks*, vol. 3, no. 1, pp. 23–43, 1990.

[133] G. Lathoud, M. Magimia-Doss, B. Mesot, and H. Boulard, "Unsupervised spectral subtraction for noise-robust ASR," in *Proc. of ASRU*, San Juan, Puerto Rico, 2005, pp. 343–348.

[134] S. L. Lauritzen, *Graphical Models*, New York: Oxford, 1996.

[135] A. Lee and T. Kawahara, "Recent development of open-source speech recognition engine Julius," in *Proc. of APSIPA ASC*, Sapporo, Japan, 2009.

[136] C.-C. Lee, C. Busso, S. Lee, and S. Narayanan, "Modeling mutual influence of interlocutor emotion states in dyadic spoken interactions," in *Proc. of Interspeech*, Brighton, UK, 2009, pp. 1983–1986.

[137] C. M. Lee, S. Narayanan, and R. Pieraccini, "Combining acoustic and language information for emotion recognition," in *Proc. of ICSLP*, Denver, USA, 2002, pp. 873–376.

[138] M. Lee and S. Narayanan, "Toward detecting emotions in spoken dialogs," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 2, pp. 293–303, 2005.

[139] X. Li, J. Tao, M. T. Johnson, J. Soltis, A. Savage, K. M. Leong, and J. D. Newman, "Stress and emotion classification using jitter and shimmer features," in *Proc. of ICASSP*, Honolulu, Hawaii, 2007, pp. 1081–1084.

[140] Y. Liang and J. D. Lee, *Driver Cognitive Distraction Detection using Eye Movements*. Springer Berlin Heidelberg, 2008, pp. 285–300.

[141] Y. Liang, J. D. Lee, and M. L. Reyes, "Nonintrusive detection of driver cognitive distraction in real time using bayesian networks," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2018/2007, pp. 1–8, 2007.

[142] Y. Liang, M. L. Reyes, and J. D. Lee, "Real-time detection of driver cognitive distraction using support vector machines," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 2, pp. 340–350, 2007.

[143] H. Lin, J. A. Bilmes, D. Vergyri, and K. Kirchhoff, "OOV detection by joint word/phone lattice alignment," in *Proc. of ASRU*, Kyoto, Japan, 2007, pp. 478–483.

[144] H. Lin, A. Stupakov, and J. A. Bilmes, "Improving multi-lattice alignment based spoken keyword spotting," in *Proc. of ICASSP*, Taipei, Taiwan, 2009, pp. 4877–4880.

[145] T. Lin, B. G. Horne, P. Tino, and C. L. Giles, "Learning long-term dependencies in NARX recurrent neural networks," *IEEE Transactions on Neural Networks*, vol. 7, no. 6, pp. 1329–1338, 1996.

[146] D. Litman and K. Forbes, "Recognizing emotions from student speech in tutoring dialogues," in *Proc. of ASRU*, Virgin Island, 2003, pp. 25–30.

[147] H. Liu, H. Lieberman, and T. Selker, "A model of textual affect sensing using real-world knowledge," in *Proc. of the 8th international conference on intelligent user interfaces*, Miami, Florida, 2003, pp. 125–132.

[148] Y. Lu, I. Cohen, X. S. Zhou, and Q. Tian, "Feature selection using principal feature analysis," in *Proceedings of the 15th international conference on Multimedia*, Augsburg, Germany, 2007, pp. 301–304.

[149] N. Ma, J. Barker, H. Christensen, and P. Green, "Distant microphone speech recognition in a noisy indoor environment: combining soft missing data and speech fragment decoding," in *Proc. of ISCA Workshop on Statistical And Perceptual Audition (SAPA)*, Makuhari, Japan, 2010.

[150] A. Maier, C. Hacker, S. Steidl, E. Nöth, and H. Niemann, "Robust Parallel Speech Recognition in Multiple Energy Bands," in *Proc. of Pattern Recognition, DAGM Symposium*, Vienna, Austria, 2005, pp. 133–140.

[151] J. Mamou, B. Ramabhadran, and O. Siohan, "Vocabulary independent spoken term detection," in *Proc. of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, Amsterdam, The Netherlands, 2007, pp. 615–622.

[152] O. Martin, I. Kotsia, B. Macq, and I. Pitas, "The enterface'05 audio-visual emotion database," in *Proc. of IEEE Workshop on Multimedia Database Management*, Atlanta, 2006.

[153] C. Mayo, J. M. Scobbie, N. Hewlett, and D. Waters, "The influence of phonemic awareness development on acoustic cue weighting strategies in children's speech perception," *Journal of Speech, Language, and Hearing Research*, vol. 46, pp. 1184–1196, 2003.

[154] G. McKeown, M. Valstar, R. Cowie, M. Pantic, and M. Schröder, "The SEMAINE database: Annotated multimodal records of emotionally coloured conversations between a person and a limited agent," *IEEE Transactions on Affective Computing*, vol. 3, no. 1, pp. 5–17, 2011.

[155] G. McKeown, M. F. Valstar, M. Pantic, and R. Cowie, "The SEMAINE corpus of emotionally coloured character interactions," in *Proc. of ICME*, Singapore, 2010, pp. 1–6.

[156] M. F. McTear, "Spoken dialogue technology: enabling the conversational user interface," *ACM Computing Surverys*, vol. 34, no. 1, pp. 90–169, 2002.

[157] A. Mehrabian, "Communication without words," *Psychology today*, vol. 2, pp. 53–56, 1968.

[158] B. Mesot and D. Barber, "Switching linear dynamic systems for noise robust speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 6, pp. 1850–1858, 2007.

[159] A. Metallinou, C.Busso, S. Lee, and S. Narayanan, "Visual emotion recognition using compact facial representations and viseme information," in *Proc. of ICASSP*, Dallas, Texas, 2010, pp. 2474–2477.

[160] A. Metallinou, S. Lee, and S. Narayanan, "Decision level combination of multiple modalities for recognition and analysis of emotional expression," in *Proc. of ICASSP*, Dallas, Texas, 2010, pp. 2462–2465.

[161] 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*, vol. 3, no. 2, pp. 184–198, 2012.

[162] C. D. Mitchell and A. R. Setlur, "Improving spelling recognition using a tree-based fast lexical match," in *Proc. of ICASSP*, Phoenix, AZ, USA, 1999, pp. 597–600.

[163] P. J. Moreno, "Speech recognition in noisy environments," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, 1996.

[164] S. Mota and R. Picard, "Automated posture analysis for detecting learner's interest level," in *Proc. of Workshop on CVPR for HCI*, Madison, 2003, pp. 49–55.

[165] K. Murphy, "Dynamic bayesian networks: representation, inference and learning," Ph.D. dissertation, Dept. EECS, CS Division, Univ. California, Berkeley, 2002.

[166] J. Nicholson, K. Takahashi, and R. Nakatsu, "Emotion recognition in speech using neural networks," *Neural Computing and Applications*, vol. 9, pp. 290–296, 2000.

[167] M. A. Nicolaou, H. Gunes, and M. Pantic, "Continuous prediction of spontaneous affect from multiple cues and modalities in valence-arousal space," *IEEE Transactions on Affective Computing*, vol. 2, no. 2, pp. 92–105, 2011.

[168] A. Nijholt and J. Hulstijn, "Multimodal interactions with agents in virtual worlds," in *Future directions for intelligent information systems and information science*, N. Kasabov, Ed. Physica-Verlag, 2000, pp. 148–173.

[169] S. Pan, J. Tao, and Y. Li, "The CASIA audio emotion recognition method for audio/visual emotion challenge 2011," in *Proc. of First International Audio/Visual Emotion Challenge and Workshop (AVEC 2011) held in conjunction with ACII*, Memphis, Tennessee, USA, 2011, pp. 388–395.

[170] D. Pardo, B. L. Mencia, A. H. Trapote, and L. Hernandez, "Non-verbal communication strategies to improve robustness in dialogue systems: a comparative study," *Journal on Multimodal User Interfaces*, vol. 3, no. 4, pp. 285–297, 2010.

[171] S. Parveen and P. Green, "Speech enhancement with missing data techniques using recurrent neural networks," in *Proc. of ICASSP*, Montreal, Canada, 2004, pp. 733–736.

[172] A. Pentland and A. Liu, "Modeling and prediction of human behavior," *Neural Computation*, vol. 11, pp. 229–242, 1999.

[173] V. Petrushin, "Emotion in speech: Recognition and application to call centers," *Artif. Neu. Net. Engr. (ANNIE)*, 1999.

[174] R. Picard, *Affective Computing*. Cambridge, MA: MIT Press, 1997.

[175] M. A. Pitt, L. Dilley, K. Johnson, S. Kiesling, W. Raymond, E. Hume, and E. Fosler-Lussier, *Buckeye Corpus of Conversational Speech (2nd release)*. Columbus, OH, USA: Department of Psychology, Ohio State University (Distributor), 2007, [www.buckeyecorpus.osu.edu].

[176] J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods: Support Vector Learning*. Cambridge, MA: MIT Press, 1999, pp. 185–208.

[177] R. Plutchik, *Emotion: A psychoevolutionary synthesis*. NY, USA: Harper and Row, 1980.

[178] T. S. Polzin and A. Waibel, "Emotion-sensitive human-computer interfaces," in *Proc. of the ISCA ITRW on Speech and Emotion*, Newcastle, Northern Ireland, UK, 2000, pp. 201–206.

[179] A. Potamianos, S. Narayanan, and S. Lee, "Automatic speech recognition for children," in *Proc. of Eurospeech*, Rhodes, Greece, 1997, pp. 2371–2374.

[180] E. Principi, S. Cifani, C. Rocchi, S. Squartini, and F. Piazza, "Keyword spotting based system for conversation fostering in tabletop scenarios: preliminary evaluation," in *Proc. of HSI*, Catania, Italy, 2009, pp. 216–219.

[181] L. Qiao, M. Sato, and H. Takeda, "Learning algorithm of environmental recognition in driving vehicle," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, no. 6, pp. 917–925, 1995.

[182] A. Quattoni, S. Wang, L. P. Morency, M. Collins, and T. Darrell, "Hidden conditional random fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, pp. 1848–1853, 2007.

[183] L. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, pp. 257–286, 1989.

[184] M. G. Rahim, B. H. Juang, W. Chou, and E. Buhrke, "Signal conditioning techniques for robust speech recognition," *IEEE Signal Processing Letters*, vol. 3, pp. 107–109, 1996.

[185] B. Raj, R. Singh, and T. Virtanen, "Phoneme-dependent NMF for speech enhancement in monaural mixtures," in *Proc. of Interspeech*, Florence, Italy, 2011, pp. 1217–1220.

[186] B. Raj, T. Virtanen, S. Chaudhuri, and R. Singh, "Non-negative matrix factorization based compensation of music for automatic speech recognition," in *Proc. of Interspeech*, Makuhari, Japan, 2010, pp. 717–720.

[187] G. Ramirez, T. Baltrusaitis, and L. P. Morency, "Modeling latent discriminative dynamic of multi-dimensional affective signals," in *Proc. of First International Audio/Visual Emotion Challenge and Workshop (AVEC 2011) held in conjunction with ACII*, Memphis, Tennessee, USA, 2011, pp. 396–406.

[188] T. Ranney, E. Mazzae, R. Garrott, and M. Goodman, "NHTSA driver distraction research: past, present and future," Washington, DC: National Highway Traffic Safety Administration, Tech. Rep., 2000.

[189] S. J. Rennie, J. R. Hershey, and P. A. Olsen, "Efficient model-based speech separation and denoising using non-negative subspace analysis," in *Proc. of ICASSP*, Las Vegas, NV, USA, 2008, pp. 1833–1836.

[190] B. H. Repp, "Some observations on the development of anticipatory coarticulation," *Journal of the Acoustic Society of America*, vol. 79, no. 5, pp. 1616–1619, 1986.

[191] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: the RPROP algorithm," in *Proc. of IEEE International Conference on Neural Networks*, 1993, pp. 586–591.

[192] G. Rigoll, "Maximum mutual information neural networks for hybrid connectionist-hmm speech recognition systems," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 2, no. 1, 1994.

[193] G. Rigoll, R. Müller, and B. Schuller, "Speech emotion recognition exploiting acoustic and linguistic information sources," in *Proc. of SPECOM*, Patras, Greece, 2005, pp. 61–67.

[194] M. Rimini-Döring, T. Altmüller, U. Ladstätter, and M. Rossmeier, "Effects of lane departure warning on drowsy drivers' performance and state in a simulator," in *Proc. of 3. International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design*, Rockport, USA, 2005.

[195] R. C. Rose, "Keyword detection in conversational speech utterances using hidden markov model based continuous speech recognition," *Computer Speech and Language*, vol. 9, no. 4, pp. 309–333, 1995.

[196] R. C. Rose and D. B. Paul, "A hidden Markov model based keyword recognition system," in *Proc. of ICASSP*, Albuquerque, NM, USA, 1990, pp. 129–132.

[197] F. Rosenblatt, *Principles of Neurodynamics*. New York: Spartan, 1963.

[198] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning internal representations by error propagation*. Cambridge, MA, USA: MIT Press, 1986.

[199] J. A. Russell, "A circumplex model of affect," *Journal of Personality and Social Psychology*, vol. 39, pp. 1161–1178, 1980.

[200] H. Sacks, E. A. Schegloff, and G. Jefferson, "A simplest systematics for the organization of turn-taking for conversation," *Language*, vol. 50, no. 4, pp. 696–735, 1974.

[201] A. Sayedelahl, P. Fewzee, M. Kamel, and F. Karray, "Audio-based emotion recognition from natural conversations based on co-occurrence matrix and frequency domain energy distribution features," in *Proc. of First International Audio/Visual Emotion Challenge and Workshop (AVEC 2011) held in conjunction with ACII*, Memphis, Tennessee, USA, 2011, pp. 407–414.

[202] A. M. Schaefer, S. Udluft, and H. G. Zimmermann, "Learning long-term dependencies with recurrent neural networks," *Neurocomputing*, vol. 71, no. 13-15, pp. 2481–2488, 2008.

[203] J. Schmidhuber, "Learning complex extended sequences using the principle of history compression," *Neural Computing*, vol. 4, no. 2, pp. 234–242, 1992.

[204] M. N. Schmidt and R. K. Olsson, "Single-channel speech separation using sparse non-negative matrix factorization," in *Proc. of Interspeech*, Pittsburgh, PA, USA, 2006, pp. 2614–2617.

[205] B. Schölkopf and A. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning)*. Cambridge, MA: MIT Press, 2002.

[206] M. Schröder, E. Bevacqua, R. Cowie, F. Eyben, H. Gunes, D. Heylen, M. ter Maat, G. McKeown, S. Pammi, M. Pantic, C. Pelachaud, B. Schuller, E. de Sevin, M. Valstar, and M. Wöllmer, "Building autonomous sensitive artificial listeners," *IEEE Transactions on Affective Computing*, vol. 3, no. 2, pp. 165–183, 2012.

[207] M. Schröder, R. Cowie, D. Heylen, M. Pantic, C. Pelachaud, and B. Schuller, "Towards responsive sensitive artificial listeners," in *Proc. of 4th Intern. Workshop on Human-Computer Conversation*, Bellagio, Italy, 2008, pp. 1–6.

[208] M. Schröder and J. Trouvain, "The german text-to-speech synthesis system mary: A tool for research, development and teaching," *International Journal of Speech Technology*, vol. 6, no. 4, pp. 365–377, 2003.

[209] M. Schröder, "The SEMAINE API: towards a standards-based framework for building emotion-oriented systems," *Advances in Human-Computer Interaction*, vol. 2010, no. 319406, 2010.

[210] B. Schuller, A. Batliner, S. Steidl, and D. Seppi, "Emotion recognition from speech: Putting ASR in the loop," in *Proc. of ICASSP*, Taipei, Taiwan, 2009, pp. 4585–4588.

[211] B. Schuller, A. Batliner, S. Steidl, and D. Seppi, "Recognising realistic emotions and affect in speech: State of the art and lessons learnt from the first challenge," *Speech Communication, Special Issue on "Sensing Emotion and Affect – Facing Realism in Speech Processing"*, vol. 53, no. 9-10, pp. 1062–1087, 2011.

[212] B. Schuller, F. Metze, S. Steidl, A. Batliner, F. Eyben, and T. Polzehl, "Late fusion of individual engines for improved recognition of negative emotion in speech – learning vs. democratic vote," in *Proc. of ICASSP*, Dallas, Texas, 2010, pp. 5230–5233.

[213] B. Schuller, R. Müller, M. Lang, and G. Rigoll, "Speaker independent emotion recognition by early fusion of acoustic and linguistic features within ensemble," in *Proc. of Interspeech*, Lisbon, Portugal, 2005, pp. 805–808.

[214] B. Schuller, S. Reiter, and G. Rigoll, "Evolutionary feature generation in speech emotion recognition," in *Proc. of ICME*, Toronto, Canada, 2006, pp. 5–8.

[215] B. Schuller and G. Rigoll, "Timing levels in segment-based speech emotion recognition," in *Proc. of Interspeech*, Pittsburgh, USA, 2006, pp. 1818–1821.

[216] B. Schuller, G. Rigoll, S. Can, and H. Feussner, "Emotion sensitive speech control for human-robot interaction in minimal invasive surgery," in *Proc. of 17th Intern. Symposium on Robot and Human Interactive Communication, RO-MAN 2008*, Munich, Germany, 2008, pp. 453–458.

[217] B. Schuller, G. Rigoll, and M. Lang, "Hidden markov model-based speech emotion recognition," in *Proc. of ICASSP*, Hong Kong, China, 2003, pp. 1–4.

[218] B. Schuller, G. Rigoll, and M. Lang, "Speech emotion recognition combining acoustic features and linguistic information in a hybrid support vector machine-belief network architecture," in *Proc. of ICASSP*, Montreal, Canada, 2004, pp. 577–580.

[219] B. Schuller, S. Steidl, and A. Batliner, "The Interspeech 2009 emotion challenge," in *Proc. of Interspeech*, Brighton, UK, 2009, pp. 312–315.

[220] B. Schuller, S. Steidl, A. Batliner, F. Burkhardt, L. Devillers, C. Müller, and S. Narayanan, "The Interspeech 2010 Paralinguistic Challenge," in *Proc. of Interspeech*, Makuhari, Japan, 2010, pp. 2794–2797.

[221] B. Schuller, M. Valstar, F. Eyben, G. McKeown, R. Cowie, and M. Pantic, "AVEC - the first international Audio/Visual Emotion Challenge," in *Proc. of First International Audio/Visual Emotion Challenge and Workshop (AVEC 2011) held in conjunction with ACII*, Memphis, Tennessee, USA, 2011, pp. 415–424.

[222] B. Schuller, B. Vlasenko, R. Minguez, G. Rigoll, and A. Wendemuth, "Comparing one and two-stage acoustic modeling in the recognition of emotion in speech," in *Proc. of ASRU*, Kyoto, Japan, 2007, pp. 596–600.

[223] B. Schuller, M. Wimmer, L. Mösenlechner, D. Arsic, and G. Rigoll, "Brute-forcing hierarchical functionals for paralinguistics: A waste of feature space?" in *Proc. of ICASSP*, Las Vegas, NV, 2008, pp. 4501–4504.

[224] B. Schuller, M. Wöllmer, F. Eyben, and G. Rigoll, "Spectral or voice quality? feature type relevance for the discrimination of emotion pairs," in *The Role of Prosody in Affective Speech, Linguistic Insights, Studies in Language and Communication*, S. Hancil, Ed. Peter Lang Publishing Group, 2009, pp. 285–307.

[225] B. Schuller, M. Wöllmer, T. Moosmayr, and G. Rigoll, "Speech recognition in noisy environments using a switching linear dynamic model for feature enhancement," in *Proc. of Interspeech*, Brisbane, Australia, 2008, pp. 1789–1792.

[226] B. Schuller, M. Wöllmer, T. Moosmayr, and G. Rigoll, "Recognition of noisy speech: A comparative survey of robust model architecture and feature enhancement," *Journal on Audio, Speech, and Music Processing*, 2009, iD 942617.

[227] B. Schuller, M. Wöllmer, T. Moosmayr, G. Ruske, and G. Rigoll, "Switching linear dynamic models for noise robust in-car speech recognition," in *Proc. of 30th DAGM Symposium, Munich, Germany*, vol. LNCS 5096. Springer, 2008, pp. 244–253.

[228] B. Schuller, R. Müller, F. Eyben, J. Gast, B. Hörnler, M. Wöllmer, G. Rigoll, A. Höthker, and H. Konosu, "Being bored? recognising natural interest by extensive audiovisual integration for real-life application," *Image and Vision Computing Journal, Special Issue on Visual and Multimodal Analysis of Human Spontaneous Behavior*, vol. 27, no. 12, pp. 1760–1774, 2009.

[229] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, pp. 2673–2681, 1997.

[230] F. Seide, P. Vu, C. Ma, and E. Chang, "Vocabulary-independent search in spontaneous speech," in *Proc. of ICASSP*, Montreal, Canada, 2004, pp. 253–256.

[231] E. Shriberg, "Spontaneous speech: How peoply really talk and why engineers should care," in *Proc. of Interspeech*, Lisbon, Portugal, 2005, pp. 1781–1784.

[232] P. Smaragdis, "Discovering auditory objects through non-negativity constraints," in *Proc. of SAPA*, Jeju, Korea, 2004.

[233] P. Smaragdis, "Convolutive speech bases and their application to supervised speech separation," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 1, pp. 1–14, 2007.

[234] P. Smith, M. Shah, and N. da Vitoria Lobo, "Determining driver visual attention with one camera," *IEEE Transactions on Intelligent Transportation Systems*, vol. 4, no. 4, pp. 205–218, 2003.

[235] G. W. Snedecor and W. G. Cochran, *Statistical methods (8th ed.)*. Iowa State University Press, 1989.

[236] S. Steidl, *Automatic Classification of Emotion-Related User States in Spontaneous Speech*, Logos, Berlin, Germany, 2009.

[237] S. Steidl, B. Schuller, A. Batliner, and D. Seppi, "The hinterland of emotions: Facing the open-microphone challenge," in *Proc. of ACII*, Amsterdam, The Netherlands, 2009, pp. 690–697.

[238] S. Steininger, F. Schiel, O. Dioubina, and S. Raubold, "Development of user-state conventions for the multimodal corpus in smartkom," in *Workshop on Multimodal Resources and Multimodal Systems Evaluation*, Las Palmas, 2002, pp. 33–37.

[239] M. Streit, A. Batliner, and T. Portele, "Emotions Analysis and Emotion-Handling Subdialogues," in *SmartKom: Foundations of Multimodal Dialogue Systems*, W. Wahlster, Ed. Berlin: Springer, 2006, pp. 317–332.

[240] A. Stupakov, E. Hanusa, J. Bilmes, and D. Fox, "COSINE - a corpus of multi-party conversational speech in noisy environments," in *Proc. of ICASSP*, Taipei, Taiwan, 2009, pp. 4153–4156.

[241] A. Stupakov, E. Hanusa, D. Vijaywargi, D. Fox, and J. Bilmes, "The design and collection of COSINE, a multi-microphone in situ speech corpus recorded in noisy environments," *Computer Speech and Language*, vol. 26, no. 1, pp. 52–66, 2011.

[242] Y. Sugimoto and C. Sauer, "Effectiveness estimation method for advanced driver assistance system and its application to collision mitigation brake system," in *Proc. of 19th International Technical Conference on Enhanced Safety Vehicles*, 2005, pp. 1–8.

[243] A. Tawari and M. Trivedi, "Speech emotion analysis in noisy real world environment," in *Proc. of ICPR*, Istanbul, Turkey, 2010, pp. 4605–4608.

[244] M. ter Maat, K. P. Truong, and D. Heylen, "How Turn-Taking strategies influence users' impressions of an agent," in *Proc. of Intelligent Virtual Agents*, Philadelphia, USA, 2010, pp. 441–453.

[245] S. Thomas, S. Ganapathy, and H. Hermansky, "Phoneme recognition using spectral envelope and modulation frequency features," in *Proc. of ICASSP*, Taipei, Taiwan, 2009, pp. 4453–4456.

[246] K. Torkkola, N. Massey, and C. Wood, "Detecting driver inattention in the absence of driver monitoring sensors," in *Proc. of International Conference on Machine Learning and Applications*, Louisville, USA, 2004.

[247] E. Trentin and M. Gori, "A survery of hybrid ANN/HMM models for automatic speech recognition," *Neurocomputing*, vol. 37, no. 1-4, pp. 91–126, 2001.

[248] D. Vergyri, I. Shafran, A. Stolcke, R. R. Gadde, M. Akbacak, B. Roark, and W. Wang, "The SRI/OGI 2006 spoken term detection system," in *Proc. of Interspeech*, Antwerp, Belgium, 2007, pp. 2393–2396.

[249] O. Viikki and K. Laurila, "Cepstral domain segmental feature vector normalization for noise robust speech recognition," *Speech Communication*, vol. 25, pp. 133–147, 1998.

[250] P. A. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[251] B. Vlasenko, B. Schuller, A. Wendemuth, and G. Rigoll, "Frame vs. turn-level: Emotion recognition from speech considering static and dynamic processing," in *Proc. of ACII*, A. Paiva, Ed., vol. LNCS 4738. Lisbon, Portugal: Springer Berlin, Heidelberg, 2007, pp. 139–147.

[252] M. T. Vo and A. Waibel, "Multimodal human-computer interaction," in *Proc. of ISSD*, Waseda, Japan, 1993, pp. 95–101.

[253] T. Vogt and E. André, "Comparing feature sets for acted and spontaneous speech in view of automatic emotion recognition," in *Proc. of ICME*, Amsterdam, The Netherlands, 2005, pp. 474–477.

[254] W. Wahlster, "Smartkom: Symmetric multimodality in an adaptive and reusable dialogue shell," in *Proc. of Human Computer Interaction Status Conference*, vol. 3, 2003, pp. 47–62.

[255] H. C. Wang, J. F. Wang, and Y. N. Liu, "A conversational agent for food-ordering dialog based on VenusDictate," in *Proc. of ROCLING X International Conference*, 1997, pp. 325–334.

[256] J. Wang, R. Knipling, and M. Goodman, "The role of driver inattention in crashes; new statistics from the 1995 crashworthiness data system (CDS)," in *40th Annual Proc.: Association for the Advancement of Automotive Medicine*, 1996.

[257] W. Wang, A. Cichocki, and J. A. Chambers, "A multiplicative algorithm for convolutive non-negative matrix factorization based on squared Euclidean distance," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2858–2864, 2009.

[258] Y. Wang and I. H. Witten, "Modeling for optimal probability prediction," in *Proc. of the Nineteenth International Conference in Machine Learning*, Sydney, Australia, 2002, pp. 650–657.

[259] M. Weintraub, "Keyword-spotting using SRI's DECIPHER large vocabulary speech recognition system," in *Proc. of ICASSP*, Minneapolis, USA, 1993, pp. 463–466.

[260] F. Weninger, J. Geiger, M. Wöllmer, B. Schuller, and G. Rigoll, "The Munich 2011 CHiME Challenge Contribution: NMF-BLSTM Speech Enhancement and Recognition for Reverberated Multisource Environments," in *Proc. of Machine Listening in Multisource Environments (CHiME 2011), satellite workshop of Interspeech 2011*, Florence, Italy, 2011, pp. 24–29.

[261] F. Weninger, A. Lehmann, and B. Schuller, "openBliSSART: Design and Evaluation of a Research Toolkit for Blind Source Separation in Audio Recognition Tasks," in *Proc. of ICASSP*, Prague, Czech Republic, 2011, pp. 1625–1628.

[262] F. Weninger, B. Schuller, A. Batliner, S. Steidl, and D. Seppi, "Recognition of Nonprototypical Emotions in Reverberated and Noisy Speech by Nonnegative Matrix Factorization," *EURASIP Journal on Advances in Signal Processing*, 2011, article ID 838790.

[263] F. Weninger, B. Schuller, M. Wöllmer, and G. Rigoll, "Localization of non-linguistic events in spontaneous speech by non-negative matrix factorization and Long Short-Term Memory," in *Proc. of ICASSP*, Prague, Czech Republic, 2011, pp. 5840–5843.

[264] F. Weninger, M. Wöllmer, J. Geiger, B. Schuller, J. F. Gemmeke, A. Hurmalainen, T. Virtanen, and G. Rigoll, "Non-Negative Matrix Factorization for Highly Noise-Robust ASR: to Enhance or to Recognize?" in *Proc. of ICASSP*, Kyoto, Japan, 2012, pp. 4681–4684.

[265] P. Wik and A. Hjalmarsson, "Embodied conversational agents in computer assisted language learning," *Speech Communication*, vol. 51, no. 10, pp. 1024–1037, 2009.

[266] R. J. Williams and D. Zipser, "Gradient-based learning algorithms for recurrent neural networks and their computational complexity," in *Back-propagation:*

Theory, Architectures and Applications, Y. Chauvin and D. E. Rumelhart, Eds. Lawrence Erlbaum Publishers, Hillsdale, N.J., 1995, pp. 433–486.

[267] K. W. Wilson, B. Raj, P. Smaragdis, and A. Divakaran, "Speech denoising using nonnegative matrix factorization with priors," in *Proc. of ICASSP*, Las Vegas, NV, USA, 2008, pp. 4029–4032.

[268] T. Winograd, "Understanding natural language," *Cognitive Psychology*, vol. 3, no. 1, pp. 1–191, 1972.

[269] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*, 2nd ed. San Francisco: Morgan Kaufmann, 2005.

[270] M. Wöllmer, M. Al-Hames, F. Eyben, B. Schuller, and G. Rigoll, "A multidimensional dynamic time warping algorithm for efficient multimodal fusion of asynchronous data streams," *Neurocomputing*, vol. 73, no. 1-3, pp. 366–380, 2009.

[271] M. Wöllmer, C. Blaschke, T. Schindl, B. Schuller, B. Färber, S. Mayer, and B. Trefflich, "On-line driver distraction detection using long short-term memory," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 2, pp. 574–582, 2011.

[272] M. Wöllmer, F. Eyben, A. Graves, B. Schuller, and G. Rigoll, "A Tandem BLSTM-DBN architecture for keyword spotting with enhanced context modeling," in *Proc. of NOLISP*, Vic, Spain, 2009.

[273] M. Wöllmer, F. Eyben, A. Graves, B. Schuller, and G. Rigoll, "Bidirectional LSTM networks for context-sensitive keyword detection in a cognitive virtual agent framework," *Cognitive Computation*, vol. 2, no. 3, pp. 180–190, 2010.

[274] M. Wöllmer, F. Eyben, A. Graves, B. Schuller, and G. Rigoll, "Improving keyword spotting with a tandem BLSTM-DBN architecture," in *Non-Linear Speech Processing*, J. Sole-Casals and V. Zaiats, Eds. Springer Heidelberg, 2010, pp. 68–75.

[275] M. Wöllmer, F. Eyben, J. Keshet, A. Graves, B. Schuller, and G. Rigoll, "Robust discriminative keyword spotting for emotionally colored spontaneous speech using bidirectional LSTM networks," in *Proc. of ICASSP*, Taipei, Taiwan, 2009, pp. 3949–3952.

[276] M. Wöllmer, F. Eyben, S. Reiter, B. Schuller, C. Cox, E. Douglas-Cowie, and R. Cowie, "Abandoning emotion classes – towards continuous emotion recognition with modelling of long-range dependencies," in *Proc. of Interspeech*, Brisbane, Australia, 2008, pp. 597–600.

[277] M. Wöllmer, F. Eyben, B. Schuller, E. Douglas-Cowie, and R. Cowie, "Data-driven clustering in emotional space for affect recognition using discriminatively trained LSTM networks," in *Proc. of Interspeech*, Brighton, UK, 2009, pp. 1595–1598.

[278] M. Wöllmer, F. Eyben, B. Schuller, and G. Rigoll, "Robust vocabulary independent keyword spotting with graphical models," in *Proc. of ASRU*, Merano, Italy, 2009, pp. 349–353.

[279] M. Wöllmer, F. Eyben, B. Schuller, and G. Rigoll, "Recognition of spontaneous conversational speech using long short-term memory phoneme predictions," in *Proc. of Interspeech*, Makuhari, Japan, 2010, pp. 1946–1949.

[280] M. Wöllmer, F. Eyben, B. Schuller, and G. Rigoll, "Spoken term detection with connectionist temporal classification - a novel hybrid CTC-DBN decoder," in *Proc. of ICASSP*, Dallas, Texas, 2010, pp. 5274–5277.

[281] M. Wöllmer, F. Eyben, B. Schuller, and G. Rigoll, "A multi-stream ASR framework for BLSTM modeling of conversational speech," in *Proc. of ICASSP*, Prague, Czech Republic, 2011, pp. 4860–4863.

[282] M. Wöllmer, F. Eyben, B. Schuller, and G. Rigoll, "Temporal and situational context modeling for improved dominance recognition in meetings," in *Proc. of Interspeech*, Portland, Oregon, USA, 2012.

[283] M. Wöllmer, F. Eyben, B. Schuller, Y. Sun, T. Moosmayr, and N. Nguyen-Thien, "Robust in-car spelling recognition - a tandem BLSTM-HMM approach," in *Proc. of Interspeech*, Brighton, UK, 2009, pp. 2507–2510.

[284] M. Wöllmer, M. Kaiser, F. Eyben, B. Schuller, and G. Rigoll, "LSTM-modeling of continuous emotions in an audiovisual affect recognition framework," *Image and Vision Computing*, vol. 31, no. 2, pp. 153–163, 2013.

[285] M. Wöllmer, M. Kaiser, F. Eyben, F. Weninger, B. Schuller, and G. Rigoll, "Fully automatic audiovisual emotion recognition: Voice, words, and the face," in *Proc. of ITG*, Braunschweig, Germany, 2012.

[286] M. Wöllmer, N. Klebert, and B. Schuller, "Switching linear dynamic models for recognition of emotionally colored and noisy speech," in *Proc. of ITG*, Bochum, Germany, 2010.

[287] M. Wöllmer, E. Marchi, S. Squartini, and B. Schuller, "Multi-stream LSTM-HMM decoding and histogram equalization for noise robust keyword spotting," *Cognitive Neurodynamics*, vol. 5, no. 3, pp. 253–264, 2011.

[288] M. Wöllmer, E. Marchi, S. Squartini, and B. Schuller, "Robust multi-stream keyword and non-linguistic vocalization detection for computationally intelligent virtual agents," in *Proc. of ISNN*, Guilin, China, 2011, pp. 496–505.

[289] M. Wöllmer, A. Metallinou, F. Eyben, B. Schuller, and S. Narayanan, "Context-sensitive multimodal emotion recognition from speech and facial expression using bidirectional LSTM modeling," in *Proc. of Interspeech*, Makuhari, Japan, 2010, pp. 2362–2365.

[290] M. Wöllmer, A. Metallinou, A. Katsamanis, B. Schuller, and S. Narayanan, "Analyzing the memory of BLSTM neural networks for enhanced emotion classification in dyadic spoken interactions," in *Proc. of ICASSP*, Kyoto, Japan, 2012, pp. 4157–4160.

[291] M. Wöllmer and B. Schuller, "Enhancing spontaneous speech recognition with BLSTM features," in *Proc. of NOLISP*, Las Palmas de Gran Canaria, Spain, 2011, pp. 17–24.

[292] M. Wöllmer and B. Schuller, "Probabilistic speech feature extraction with context-sensitive bottleneck neural networks," *Neurocomputing*, 2013.

[293] M. Wöllmer, B. Schuller, A. Batliner, S. Steidl, and D. Seppi, "Tandem decoding of children's speech for keyword detection in a child-robot interaction scenario," *ACM Transactions on Speech and Language Processing*, vol. 7, no. 4, pp. 1–26, 2011.

[294] 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*, vol. 4, no. 5, pp. 867–881, 2010.

[295] M. Wöllmer, B. Schuller, and G. Rigoll, "Feature frame stacking in RNN-based Tandem ASR systems - learned vs. predefined context," in *Proc. of Interspeech*, Florence, Italy, 2011, pp. 1233–1236.

[296] M. Wöllmer, B. Schuller, and G. Rigoll, "A novel Bottleneck-BLSTM front-end for feature-level context modeling in conversational speech recognition," in *Proc. of ASRU*, Waikoloa, Big Island, Hawaii, 2011, pp. 36–41.

[297] M. Wöllmer, B. Schuller, and G. Rigoll, "Keyword spotting exploiting Long Short-Term Memory," *Speech Communication*, vol. 55, no. 2, pp. 252–265, 2013.

[298] M. Wöllmer, Y. Sun, F. Eyben, and B. Schuller, "Long short-term memory networks for noise robust speech recognition," in *Proc. of Interspeech*, Makuhari, Japan, 2010, pp. 2966–2969.

[299] M. Wöllmer, F. Weninger, F. Eyben, and B. Schuller, "Acoustic-linguistic recognition of interest in speech with Bottleneck-BLSTM nets," in *Proc. of Interspeech*, Florence, Italy, 2011, pp. 77–80.

[300] M. Wöllmer, F. Weninger, F. Eyben, and B. Schuller, "Computational assessment of interest in speech - facing the real-life challenge," *Künstliche Intelligenz, Special Issue on Emotion and Computing*, vol. 25, no. 3, pp. 225–234, 2011.

[301] M. Wöllmer, F. Weninger, J. Geiger, B. Schuller, and G. Rigoll, "Noise robust ASR in reverberated multisource environments applying convolutive NMF and Long Short-Term Memory," *Computer Speech and Language*, vol. 27, no. 3, pp. 780–797, 2013.

[302] M. Wöllmer, F. Weninger, S. Steidl, A. Batliner, and B. Schuller, "Speech-based non-prototypical affect recognition for child-robot interaction in reverberated environments," in *Proc. of Interspeech*, Florence, Italy, 2011, pp. 3113–3116.

[303] T. Wu, F. Khan, T. Fisher, L. Shuler, and W. Pottenger, "Posting act tagging using transformation-based learning," in *Foundations of Data Mining and knowledge Discovery*, T. Y. Lin, S. Ohsuga, C. J. Liau, X. Hu, and S. Tsumoto, Eds., 2005, pp. 319–331.

[304] V. H. Yngve, "On getting a word in edgewise," in *Chicago Linguistic Society. Papers from the 6th regional meeting*, vol. 6, 1970, pp. 567–577.

[305] K. Young, M. Regan, and M. Hammer, "Driver distraction: A review of literature," Monash University Accident Research Center, Tech. Rep., 2003.

[306] Z. Zeng, M. Pantic, G. I. Rosiman, and T. S. Huang, "A survey of affect recognition methods: Audio, visual, and spontaneous expressions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 1, pp. 39–58, 2009.

[307] D. Zhang, D. Gatica-Perez, S. Bengio, I. McCowan, and G. Lathoud, "Modeling individual and group actions in meetings: a two-layer HMM framework," in *Proc. of CVPR*, Washington DC, USA, 2004, pp. 117–125.

[308] H. Zhang, M. R. H. Smith, and G. J. Witt, "Identification of real-time diagnostic measures of visual distraction with an automatic eye-tracking system," *Human Factors*, vol. 48, no. 4, pp. 805–821, 2006.

[309] X. Zhe and A. Boucouvalas, "Text-to-emotion engine for real time internet communication," in *Proc. of the International Symposium on Communication Systems, Networks, and DSPs*, Staffordshire University, 2002, pp. 164–168.

[310] Q. Zhu, B. Chen, N. Morgan, and A. Stolcke, "Tandem connectionist feature extraction for conversational speech recognition," in *Machine Learning for Multimodal Interaction*. Springer, 2005, pp. 223–231.