TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Mensch-Maschine-Kommunikation

Feature Transfer Learning for Speech Emotion Recognition

Jun Deng

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.

Vorsitzende:		Univ.-Prof. Dr.rer.nat Doris Schmitt-Landsiedel
Prüfer der Dissertation:	1.	Univ.-Prof. Dr.-Ing. habil. Björn W. Schuller
	2.	Univ.-Prof. Dr.-Ing. Werner Hemmert

Die Dissertation wurde am 24.09.2015 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 10.05.2016 angenommen.

Abstract

Speech Emotion Recognition (SER) has achieved some substantial progress in the past few decades since the dawn of emotion and speech research. In many aspects, various research efforts have been made in an attempt to achieve human-like emotion recognition performance in real-life settings. However, with the availability of speech data obtained from different devices and varied acquisition conditions, SER systems are often faced with scenarios, where the intrinsic distribution mismatch between the training and the test data has an adverse impact on these systems.

To address this issue, this thesis makes use of autoencoders as an expressive learner to introduce a set of novel feature transfer learning algorithms. They are based on the goal to achieve a matched feature space representation for the target and source sets while ensuring source domain knowledge transfer. Partly inspired by the recent successes of feature learning, this thesis first incorporates sparse autoencoders into semi-supervised feature transfer learning. Furthermore, in the unsupervised setting, i.e., without the availability of any labeled target data in the training phase, this thesis takes advantage of denoising autoencoders, shared-hidden-layer autoencoders, adaptive denoising autoencoders, extreme learning machine autoencoders, and subspace learning with denoising autoencoders, for feature transfer learning.

Experimental results are presented on a wide range of emotional speech databases , demonstrating the advantages of the proposed algorithms over other modern transfer learning methods. Besides normal phonated speech, these transfer learning methods are also evaluated on whispered speech emotion recognition, which shows that these methods can be applied to create a recognition model owing a completely trainable architecture that can adapt it to a range of speech modalities.

Acknowledgments

First of all, I would like to deeply thank my supervisor Prof. Björn Schuller for giving me complete freedom in pursuing my interests while also providing enough research guidance, for sharing with me his intellectual, for reading our paper drafts. Thanks Björn for everything.

Furthermore, I am very grateful to Prof. Gerhard Rigoll for creating the great working atmosphere at the Institute for Human-Machine Communication at Technische Universität München, Germany.

I would further like to thank especially the following people for their support and collaboration: Zixing Zhang, Florian Eyben, Erik Marchi, Felix Weninger, XinZhou Xu, Yue Zhang, Rui Xia, Eduardo Coutinho, Peter Brand, Martina Römpp, Heiner Hundhammer, Shaowei Fan, Jürgen Geiger, and Fabien Ringeval.

I thank my family for their support. In particular, I thank my parents Tianzhi Deng and Qiongchao Zhang and brother Bo Deng who have given a lifetime of love and care. Finally, I would like to thank my wife Haiyan Zhou for unconditional support and encouragement.

The work in this thesis was funded by the Chinese Scholarship Council (CSC).

Jun Deng
Munich
September 2015

Dedicated to my parents, Tianzhi Deng and Qiongchao Zhang.

Contents

1 Introduction

1.1 Motivation

In our daily life, speech plays a prominent role in human communication. Accordingly, *Automatic Speech Recognition* (ASR) is dedicated to enable a machine to possess the ability to recognize and understand spoken words as well as humans do [1]. Although linguistic expression can be highly ambiguous, it can still often be interpreted correctly by today's advanced ASR techniques.

In speech, however, a listener can not only hear what a speaker is saying, but also perceive how a speaker is saying it. The listener's perceptions include the speaker's emotions. The emotions can be perceived by the listener due to the fact that changes in the autonomic and somatic nervous system have an indirect yet strong influence on the speech production process [2]. It means that apart from linguistic information such as words and sentences, speech also carries rich emotional information such as anger and happiness. Besides interpreting spoken words by ASR, therefore, an intelligent machine should also have the ability to recognize emotions from speech, so that the communication between humans and machines becomes natural and friendly just like human-to-human communication. This kind of capability is known as *Speech Emotion Recognition* (SER) or *acoustic emotion recognition*. The introduction of SER allows machines to extract and interpret human emotions by analyzing speech patterns with *machine learning* methods.

SER leads to many practical applications. For example, SER is being applied to develop communicative platforms for use by children with autism spectrum conditions [3], such as in the EU-funded ASC-Inclusion project[1]. In e-learning applications, SER can be used to improve students' learning experience by adjusting learning material delivery based on the their emotional states [4]. Further, robots use SER abilities to guide their behavior and further socially communicate affective reinforcement [5]. Therefore, it is not surprising that SER has grown in a major

[1]`http://asc-inclusion.eu/`

research topic into speech processing, human-computer interaction, and computer-mediated human communication over the last decades (see [6, 7, 8, 9, 10]).

Many SER engines achieve promising performance only under one common assumption, namely that the training and test data are drawn from the same corpus and the same feature space for parametrization is used. However, with speech data increasingly obtained from different devices and varied recording conditions, we are often faced with scenarios where such data are typically highly dissimilar in terms of acoustic signal conditions, speakers, spoken languages, linguistic content, type of emotion (e.g., acted, elicited, or naturalistic), or the type of labeling scheme used, such as categorical or dimensional label [1, 11, 12, 13]. When labeling an emotional corpus, even worse, there is no certain ground truth but a subjective ambiguous 'gold standard' because different human raters may exhibit different emotional states in responses to the same speech. The profound differences across emotional speech datasets are known as the *distribution mismatch* or *dataset bias*. The distribution mismatch is prone to give rise to significant performance downgrades for SER systems, since in training these systems we will not have prepared for data subsequently encountered in use. For example, if a system builds upon a classifier using features extracted from adults' speech corpora to identify children's emotional state, its performance can be expected as very low. In this example, this comes, as – among other factors – there is a relevant difference of certain features such as pitch between adults and children on which emotion phenomena rely heavily.

The best solution to alleviate the mismatch, of course, is to gain access to all variations by acquiring large amounts of emotional speech data. However, labeling emotional data not only requires skilled human raters but also is slow and expensive. Furthermore, it is impossible to anticipate all variations. So it is inevitable that there is a 'mismatch' between training and test data.

Such an inherent mismatch suggests that an SER system should stop hoping for annotated data that are not available, and instead embrace complexity and then make use of existing data so as to retrieve useful information within data for a related target task [14]. *Transfer learning* (also referred to as *domain adaptation*) has been proposed to deal with the problem of how to reuse the knowledge learned previously from 'other' data or features [15, 16]. To help advance the field of SER, this thesis puts a strong emphasis on addressing a mismatch between training and test data by integrating transfer learning into SER systems. In this mismatch, specifically, a model is trained on one database while tested on another disjoint one. As an example, the labeled corpus may be acted speech obtained through previous human labeling efforts. For a classification task on a newly spontaneous corpus where the data's features or data's distribution may be different. As a result, one may not be able to directly apply the emotion models learned on the acted speech to the new spontaneous data. In this case, new solutions presented in this thesis could transfer the classification knowledge from the acted data to the new spontaneous data.

For the major purpose of reducing a mismatch between training and test data, the following challenges are discussed in this thesis:

1. **Labeled target domain test data are partially available.** At present, a number of emotional speech corpora exist, but they are highly dissimilar and very small. In this case, a small amount of labeled data is usually insufficient to train a reliable acoustic model, which is likely to lead to low recognition accuracy. This means that there is the *data scarcity* problem in the field of SER [17, 18]. Furthermore, directly combining different corpora into the training set yields performance degradation simply because of the aforementioned differences across these corpora. Here, the combined training data typically consist of disjoint data dramatically different from the test data, and a small amount of target domain data which come from the same corpus as the test data. Hence, the first challenge this thesis deals with is how to make use of the combined data to produce satisfactory performance for emotion recognition.

2. **Labeled target domain test data are completely unavailable.** In emotion recognition, like many other machine learning tasks, data are of paramount importance. One always needs to label speech data tailored to a target task and then uses them extensively to build on a recognition model, in the hope that they provide objective guidance for discovering the relation between the inputs and the target task. It is evident that such a model only achieves success for the reason that the data distribution is stable between the training and test data. The task of emotion recognition becomes more interesting but more challenging when the whole training data only come from other domains remarkably different from the target domain, i.e., the training data without any labeled target domain data.

1.2 Contributions

To deal with the two above challenges of both normal phonated speech and whispered speech emotion recognition, this thesis attempts to make the following contributions:

1. To address the first challenge, this thesis contributes to the use of different sets of training data by proposing a novel *feature transfer learning* method. This method discovers knowledge in acoustic features from a small amount of labeled target data (similar to the test data) to achieve considerable improvement in accuracy when applying the knowledge to other domain training data (significantly different from the test data).

2. Further, this thesis puts more effort into the second challenge. Accordingly, a general framework, encompassing five feature transfer learning methods, is proposed to enhance the generalization of an emotion recognition model and

make it adaptable to a new domain. This framework enables SER to continue to enjoy the benefits of existing speech corpora.

3. Apart from *normal phonated speech* at which current studies mainly have made considerable efforts to date, in fact, *whispered speech* is another common form of speaking to communicate, which is produced by speaking with high breathiness and no periodic excitation. This thesis finally sheds light on *whispered speech emotion recognition*. It extends these transfer learning methods by showing how feature transfer learning can be applied to create a recognition engine owing a completely trainable architecture that can adapt it to a range of speech modalities, such as normal phonated speech and whispered speech.

1.3 Overview of this Thesis

The chapters are roughly grouped into three parts: a brief overview of SER is discussed in Chapter 2; Chapters 3 and 4 primarily present the theory of feature transfer learning and the experimental results; and conclusions and directions for the future work are given in Chapter 5.

Chapter 2 briefly reviews a typical SER system in general, and two fundamental components of this system, acoustic features and statistical modeling, are discussed in particular. It also provides background details for feedforward neural networks. Chapter 3 describes the distribution mismatch, and transfer learning with emphasis on transfer learning in speech processing. It also describes theoretically a set of feature transfer learning methods, which are proposed in this thesis, for dealing with the challenges outlined in Section 1.1. Chapter 4 contains practical evaluations of the methods presented in Chapter 3 on the task of SER. Chapter 5 summarizes the presented thesis and points out possible directions for future work.

2 Speech Emotion Recognition

This chapter gives an overview of *acoustic features* and *statistical modeling* methods for SER. Figure 2.1 presents a fundamental SER system made of *feature extraction* (i.e., computation of acoustic features) and statistical modeling (i.e., training classifiers and making predictions of emotions).

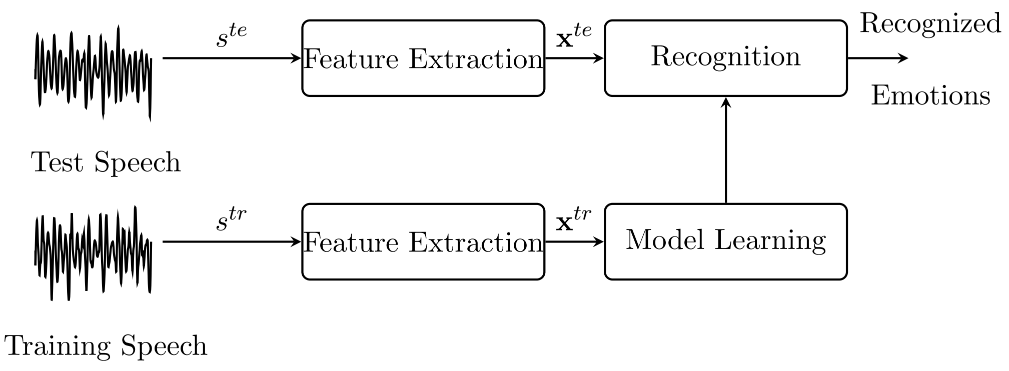


Figure 2.1: Overview of a typical speech emotion recognition system.

For SER, the raw speech signal is typically transformed into some new space of explanatory variables in which, hopefully, the emotion recognition problem will become much easier to solve. This transformation stage is known as feature extraction, which ends up producing acoustic features. Feature extraction might also be taken into account in order to reduce computation cost. For example, if the goal is real-time speech emotion detection in a distributed system, the client side with restricted computing ability must deal with large numbers of raw data per second, and sending these data directly to the server side may cause a computationally infeasible problem. In this case, the aim of the feature extraction stage is to create concise and useful features that are easy to compute, and yet that also preserve useful discriminatory information.

The core purpose of statistical modeling is to assign each input signal to one of discrete emotional states by leveraging the acoustic features obtained by feature extraction. A statistical modeling algorithm often produces a model which is determined in the training phase based on the training speech data. Once the model is trained well, it can then identify the emotional state of test speech signals.

This fundamental system (shown in Figure 2.1) plays the central role in this thesis, because this thesis will incorporate a variant of novel feature transfer learning algorithms into this fundamental SER system. Thereby, the following sections turn to an exploration of this fundamental system. The most important acoustic features, which are commonly used, are firstly discussed in Section 2.1. Statistic modeling methods used for the state-of-the-art SER systems are then presented in Section 2.2. Finally, metrics for assessing the quality of SER systems are discussed in Section 2.3.

2.1 Acoustic Features

Acoustic features, normally consisting of prosodic and spectral features, often serve to offer an extremely concise and discriminatory summary of raw speech data in SER systems [10]. Table 2.1 depicts acoustic features commonly used for SER. It is recognized that prosodic features such as pitch and the voicing probability are very important parameters which convey much of the emotional information in speech. For example, high intensity is associated with the emotion of surprise and anger while low intensity is associated with the emotion of sadness and disgust [19]. Among the most important prosodic features are the intensity, the fundamental frequency F_0, the voicing probability, and the formants [9, 20].

Besides prosodic features, various spectral features classically used to represent the phonetic content of speech for ASR, such as *Linear Prediction Cepstral Coefficients* (LPCCs) [21] and *Mel-Frequency Cepstral Coefficients* (MFCCs) [22], also efficiently work for recognizing emotions from speech signals. It is found that, the emotional state of a given speech signal leads to a remarkable impact on the distribution of the spectrum across the frequencies [19].

To facilitate acoustic feature extraction for SER, the openSMILE feature extraction toolkit [23, 24], which has become new standard of feature extraction in the field, is the first choice. The feature sets obtained by the openSMILE toolkit have been widely used, and usually adopted to build the baseline recognition systems for the recent computational paralinguistics challenges [25, 26, 27, 28]. For this reason, the feature sets chosen for this thesis are available in the toolkit and the feature extraction is fully dependent on it so that one can easily reproduce the findings of this thesis.

It is widely agreed that acoustic features can be categorized into segmental (short-time) and supra-segmental (long-time) types in accordance with their temporal structure [10, 20]. In the following two sections segmental features and supra-

Time domain descriptors
zero-crossing rate, amplitude
Energy
Root Mean Square (RMS) energy, logarithmic energy, loudness
Spectrum
linear magnitude spectrum, nonlinear magnitude/frequency scales band spectra, filterbank spectra
Spectral descriptors
band energies, spectral slope/flatness spectral centroid/moments/entropy
Cepstral features
MFCCs, PLP cepstral coefficients
Linear prediction
Linear Prediction (LP) coefficients, LP residual, LP spectrum, LPCCs
Voice quality
jitter, shimmer, Harmonics-to-noise ratio
Tonal features
semitone spectrum, pitch class profiles
Nonlinear vocal tract model features
critical band filterbanks, Teager energy operator envelopes

Table 2.1: Acoustic features commonly used for SER.

segmental features are characterized shortly. As the focus of this thesis is placed on feature transfer learning and not on acoustic features, the reader is referred to [20] and [24] for further discussion.

2.1.1 Segmental Features

Segmental features, also known as acoustic *Low Level Descriptors* (LLDs), are extracted from each short-time frame (usually 25 ms in length) based on short-time analysis. Segmental features mainly include short-term spectra and derived features: MFCCs, *Linear Prediction Coding* (LPC), LPCCs, Wavelets [29], and Teager energy operator based features [29, 30, 31]. Segmental features have proven to be successful at a variety of audio processing tasks including ASR [32], speaker recognition [33], music information retrieval applications such as genre classification [34, 35], emotion recognition [9, 10, 20, 24, 36], and computational paralinguistics [20].

Despite segmental features such as MFCCs are normally used for modeling segments such as phones for ASR, they have also served for emotion recognition. In emotion recognition, these features are either classified by using, for example, dynamic Bayesian networks and *Hidden Markov Models* (HMMs), or combined by applying some functionals , for example, taking the mean values of the features across the speech signal, and then modeled with statistical classifiers such as *Support Vector Machines* (SVMs) and *Multilayer Perceptrons* (MLPs). One evident advantage of segmental features for SER is that they retain the temporal information of the speech signals. The temporal information strongly reflects the change in the speech signals carrying an emotional state. In [19], an HMM-based classification model was proposed based on the short time log frequency power coefficients, MFCC, and LPCC. In [37], the authors used MFCCs as a representation in order to explore the temporal properties of emotion by a suite of hybrid classifiers based on HMMs and deep belief networks. In [38], the multi-taper MFCCs and *Perceptual Linear Predictions* (PLPs) were modeled using *Gaussian Mixture Models* (GMMs). Recently, an EmoNet integrating with deep learning approaches was the wining submission in the 2013 EmotiW challenge, where three types of MFCC features are used, comprising the 22 cepstral coefficients, their first-order derivatives and second-order derivatives [39, 40].

Although extracting segmental features at a fixed temporal length was usually considered in the previous work, few efforts have shown that different temporal lengths would be beneficial for modeling the underlying characteristics that result from different emotional states [41, 42]. In [42], the segmental features were extracted with 400 ms and 800 ms analysis frames, and therefore a novel fusion algorithm was introduced to fuse recognition results from classifiers trained on those multitemporal features.

Apart from the short-term characteristics, there are attempts at modeling long-term information in a speech signal based on the assumption that speech emotion is a phenomenon varying slowly over time. Wöllmer et al. [43] made use of LLDs such as signal energy, pitch, voice quality, and cepstral features which are modeled to explore contextual information of speech signals by using *Long Short-Term Memory* (LSTM) recurrent networks. One alternative method to explicitly extract long-term information hidden in a speech signal is the modulation spectrogram features [44, 45, 46]. The features, inspired by the auditory cortical, are extracted from a long-term spectro-temporal representation, using an auditory filterbank and a modulation filterbank. As a result, the modulation spectrogram features are intended to express both slow and fast change in spectrum in a way to capture information associated with speech intelligibility by quantifying the power of temporal events relating to articulatory movements in the speech signal [47]. In other words, those features are integrated with many important properties existing in human speech perception but missing from conventional short-term spectral features. Besides, the modulation spectrogram features, *Amplitude Modulation-Frequency Modulation* (AM-FM) has drawn attention in emotion recognition [31]. In [31], a smoothed nonlinear energy operator, which can track the energy needed to result in an AM-FM signal and separate it into amplitude and frequency components, was used to generate amplitude modulation cepstral coefficient features.

Although little recent research has shown that it is possible to directly use raw speech signal to model phone classes due to deep learning that is capable of finding the right features for a given task of interest, the computational cost is high and the performance is likely to be worse than for conventional features [48, 49, 50]. Therefore, segmental features are still of fundamental importance for SER. As shown above, MFCCs are the most frequently used segmental features for emotion recognition [19, 24, 37, 38, 39, 40], so the following section gives a short overview of the computation of MFCCs.

2.1.1.1 Mel-Frequency Cepstral Coefficients

Motivated by perceptual or computational considerations, *Mel-Frequency Cepstral Coefficients* (MFCCs) are designed to provide a compact representation of the short-term spectral envelope, based on the orthogonal *Discrete Cosine Transformation* (DCT) of a log power spectrum on a nonlinear mel scale of frequency. Generally, the procedure of MFCCs calculation is shown as follows:

1. Power spectrum representation of a windowed signal.

2. Map the power spectrum onto the mel scale.

3. Take the logarithms of the powers at each of the mel frequencies.

4. Take the decorrelation of the mel log powers by DCT.

It is worth noting that, human hearing perception is taken into consideration during the calculation of MFCCs. Because the human hearing understands lower frequencies more easily than higher ones [20], the power spectrum is mapped onto a mel scale by using triangular overlapping windows:

$$mel(f) = 2595 \log\left(1 + \frac{f}{700}\right), \tag{2.1}$$

where f indicates a linear frequency scale in Hz, and $mel(f)$ represents a mel scale.

In addition to coefficients 0 up to 16 or higher used for SER, their first order delta coefficients and second order delta coefficients are often appended to them.

Statistical functionals
Means
arithmetic, quadratic, root-quadratic
geometric mean, flatness, mean of absolute values
Moments
variance, standard deviation, skewness, kurtosis
Extreme values
maximum, minimum, range
Percentiles
quartiles and inter-quartile ranges
percentiles and various inter-percentile ranges
Regression
linear/quadratic regression, derivations
irreversibility, regression errors
Peak statistics
number of peaks
arithmetic mean of the peak amplitudes
absolute peak amplitude range
arithmetic mean of rising slopes
Segment statistics
number of segments
mininum, mean, maximum segment length
arithmetic mean of segment length, standard deviation of segment length
Modulation functionals
DCT coefficients, LPC coefficients
modulation spectrum, rhythmic features

Table 2.2: Common statistical and modulation functionals used for generating supra-segmental features in SER.

2.1.2 Supra-segmental Features

Unlike ASR which focuses on short-term phenomena such as phones, SER focuses on the long-term phenomena which do not change every second but evolve slowly over time. There are so-called *supra-segmental features* which tend to express the change of low-level features over a given period of time. The aim of supra-segmental features is to create a single, fixed length feature vector, summarizing serial LLDs (cf. Section 2.1.1) of possibly variable length [51]. This is the prominent approach to gathering paralinguistic feature information, because it offers a larger reduction of data that otherwise might rely too strongly on the phonetic content [20, 25]. Moreover, supra-segmental features became widespread in SER and other paralinguistic recognition tasks [20, 24, 52], and they were repeatedly reported to be superior to segmental ones in terms of classification accuracy and test time.

Supra-segmental features are derived by a projection of the multivariate time series comprised of LLDs such as MFCCs and pitch onto a single fixed dimension vector independent of the length of the entire utterance [12]. Such a projection is implemented by applying *functionals*. Simple examples of functionals are the *mean*, *variance*, *minimum*, and *maximum*. More advanced functionals are, for example, the local extrema in the input series or their distribution. The common functionals including statistic and modulation functionals which are used in SER are summarized in Table 2.2.

In contrast to segmental features (i.e., LLDs), the advantage of supra-segmental features is that they provide a representation of the variable length speech with a fixed length. Hence, they can make use of static modeling, such as *k-Nearest Neighbors* (k-NN) (see [53, 54]) and SVMs (see [25, 55, 56]), to analyze patterns in speech. In return, these approaches ease the way to build SER systems and considerably save the computation cost and test time, especially when the input speech is long.

It also seems that, the use of supra-segmental is a satisfactory solution to protect the users' privacy as well as to reduce the data transmission bandwidth from the client side to the server side for distributed recognition systems [57]. The procedure of extracting supra-segmental features is irreversible even if they originate from segmental features, for example MFCCs and pitch (see [58, 59]), which can be employed to reconstruct the audio. Therefore, they avoid the reconstruction of the speech signal and then allow to reduce the risk of leaking the speakers' speech content.This irreversibility minimizes the risk of the users' privacy violation. Further, supra-segmental features appear to save transmission bandwidth when compared to raw speech and LLDs as the vector size is always the same per utterance [57].

However, one of the major drawbacks of supra-segmental features is the loss of the temporal information, since they employ statistics of the LLDs features and neglect the sampling order. A solution to overcome the problem is to calculate statistics over rising/falling slopes or during the plateaux at minima/maxima [51, 60, 61]. An alternative solution is *feature frame stacking*. The general principle is simple: A defined number of frames are concatenated to a super-vector [20].

2.2 Statistical Modeling

When acoustic features provide the discriminatory information of the speech signal, the major role of statistical modeling is to use the these features to predict emotions and to get information about the underlying data mechanism [62]. This chapter gives a brief overview of the statistical modeling methods for SER and further a theoretical introduction to the modeling methods of interest in this thesis.

In the field of emotion recognition as well as other paralinguistic tasks, most studies have focused on classification of an utterance, where approach is normally to look for a function $f(\mathbf{x})$ – an algorithm that operates on an utterance $\mathbf{x}$ to output one category of emotional states. A wide diversity of classifiers have been used for the task of SER, which include HMMs [9, 19, 20, 37, 51, 63, 64, 65], GMMs [66, 67, 68, 69], SVMs [11, 13, 70, 71], *Artificial Neural Networks* (ANNs) or *Deep Neural Networks* (DNNs) [37, 40, 72, 73, 74, 74, 75, 76], k-NN [54, 77, 78], Bayesian networks [53], logistic regression [67], decision trees [79], ensemble learning [80, 81], and *Extreme Learning Machine* (ELM) [82, 83].

Apparently, the HMM classifier is among the most popular classifiers as it has been used in almost all speech tasks. The HMM is made of a first-order Markov chain, in which the states are unknown from the observer [84]. The HMM model is theoretically formulated under Bayesian probability theory, and hence can form the theoretical basis for use in capturing the temporal information of the data. The parameters of an HMM can be determined efficiently using the *Expectation Maximization* (EM) algorithm for maximizing the likelihood function [85]. In the HMM framework for SER, each utterance is represented as a sequence of low-level features. Thereby, given a sequence of low-level features, the HMM classifier estimates a maximum likelihood score and then infers the hidden state path with the highest probability as the predicted emotion of the utterance. One of the most useful properties of HMMs is their ability to be invariant with respect to local warping of the time axis [85]. Such a property is tailored to meet the need of the analysis for the short time behavior of human speech as considered in [64].

There are viable alternatives to making use of the HMM model for SER. For example, in the conventional HMM the hidden states are not defined explicitly, instead, in [65] the hidden states are defined by the temporal sequences of affective labels. Hence, the hidden states during the training process are known. This approach transforms the emotion classification problem into a best path-finding optimization problem in the HMM framework.

As well as being of great interest in its own right, the SVM model may have gradually dominated emotion recognition using the supra-segmental features. There are three reasons why it happened. First, the parameters of the SVM model are derived using convex optimization, leading to the global optimality. Second, it has good data-dependent generalization guarantees [86]. Third, there is growing popularity of the use of supra-segmental features as by openSMILE toolkit. Importantly, the kernel trick enables the SVM classifier to map the original feature space to a high-dimensional space where data points, it is hoped, can be more easily classified using a linear margin-based classifier.

Further, ANNs exhibit a great degree of flexibility to model these two different types of acoustic features for SER. This means that the ANN can estimate an affective label from a fixed length feature vector summarizing the utterance, as well as from a temporal sequence of low-level features. The nonlinear mapping property of ANNs is highly advantageous to find complex relationships in data. For example, an ANN/HMM system performs automatic emotion independent phone group partitioning based on using feedforward ANNs, before performing temporal modeling by HMMs [87]. In SER context, furthermore, *Convolutional Neural Networks* (CNNs), a variant of feedforward ANNs [88], serve as a feature extractor to learn affect-salient features, where simple features are learned in the lower layers, and affect-salient, discriminative features are obtained in the higher layers [89]. For the purpose of integrating emotional context, it is often effective to apply *Recurrent Neural Networks* (RNNs) to deal with emotion recognition problems [43, 75].

DNNs, the state-of-the-art ANNs, have shown great success in a wide range of applications such as computer vision, ASR, natural language processing, and audio recognition. Their success emerged in automatic emotion recognition from speech as well. Le and Provost [37] proposed a suite of hybrid classifiers where HMMs are used to capture the temporal information of emotion and deep belief networks are used to compute the emission probabilities. Further, an alternative method to take full advantage of DNNs is emoNet where an MLP is initially constructed by a deep belief network in an unsupervised way, and then inspired by a multi-time-scale learning model it pools the activations of the last hidden layer in order to aggregate information across frames before the final softmax layer [39, 40]. These two approaches were found to be competitive with the state-of-the-art methods.

In this section, only the key aspects of SVMs and ANNs as needed for application in SER are given. More general treatments of the two models can be found in the books by Bishop [85], and Schuller and Batliner [20].

2.2.1 Support Vector Machines

Support Vector Machines (SVMs) are one of the most widely used classifiers for various applications in machine learning, since a recognition system using an SVM model tends to achieve the satisfying recognition result. Recent research running a large number of classification experiments on 121 data sets concluded that the SVM is likely to achieve the best performance [90] when compared to other 17 classes of supervised classifiers. For paralinguistic tasks, the SVM is also extensively employed to build the baseline systems on the basis of supra-segmental features [20, 27, 91, 92, 93].

Motivated by statistical learning theory, the determination of the SVM parameters is equivalent to solving a convex quadratic programming problem [94, 95]. Therefore, this convexity property guarantees that any local solution is also a global optimum [85, 96]. In addition, the kernel trick for the SVM offers a computationally efficient mechanism for transforming input data to a high-dimensional space, in which the data are linearly separable.

Formally, given training examples and corresponding binary labels $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, N$, where $\mathbf{x}_i \in \mathbb{R}^n$ and $y_i \in \{-1, 1\}$, the SVMs solve the following minimization problem

$$\begin{aligned} &\underset{(\mathbf{w},b,\xi)}{\arg\min}\, \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i, \\ \text{subject to} \quad & y_i\big(\mathbf{w}^T\phi(\mathbf{x}_i + b)\big) \geq 1 - \xi_i,\ \xi_i \geq 0. \end{aligned} \tag{2.2}$$

Here, the penalty parameter C controls slack variables ξ_i to penalize data points which violate the margin requirements, and $\phi(\cdot)$ denotes a feature space mapping function. In the feature space defined by $\phi(\cdot)$, the SVMs look for a linear separating hyperplane by maximizing the margin. The margin constructed pivots around a subset of data points of the training data, which are called the *support vectors* since they support the hyperplanes on both sides of the margin. Figure 2.2 illustrates the separating hyperplane, margins, and support vectors for an SVM.

It is worth emphasizing the importance of applying the feature space mapping function $\phi(\cdot)$ in the formation of SVMs. An obvious but crucial remark is that a nonlinear classification function plays a vital role in optimally classifying nonlinearly separable data. In applying the SVM for nonlinear separable data, therefore, the input data are linearly mapped to much higher (or even infinite) dimensional space in which the data are linearly separable. This nonlinear feature space mapping is defined by $\phi(\cdot)$. Such a strategy, referred to as the *kernel trick*, enables SVMs to efficiently classify the data in very high dimensional spaces. Given the nonlinear feature space mapping function $\phi(\cdot)$, specifically, the kernel function on two examples $\mathbf{x}_i$ and $\mathbf{x}_j$ is defined as

$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j). \tag{2.3}$$

The simplest example of kernels is the linear kernel with the identity function, in which case $\phi(\mathbf{x}) = \mathbf{x}$ and $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T\mathbf{x}_j$. Another commonly used kernel is a Gaussian kernel (or radial basis function kernel) defined by

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{-||\mathbf{x}_i - \mathbf{x}_j||}{2\sigma^2}\right), \tag{2.4}$$

where σ is the width.

There is intrinsic sparsity in SVMs, providing a clue to the relationship between the solution of the SVM and the different types of the training data. The support vectors determine the location of the maximum margin hyperplane. As a result, the

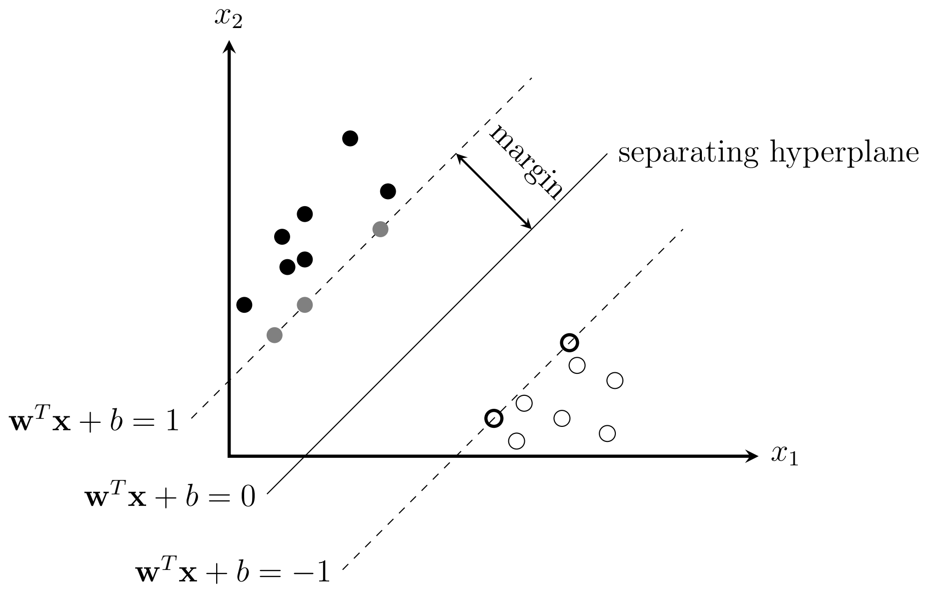

x_2
x_1
margin
separating hyperplane
$\mathbf{w}^T\mathbf{x} + b = 1$
$\mathbf{w}^T\mathbf{x} + b = 0$
$\mathbf{w}^T\mathbf{x} + b = -1$

Figure 2.2: Separating hyperplane and margins for an SVM. Samples on the margin are known as the support vectors, which are indicated by the gray dots and bold circles. The separating hyperplane is achieved by maximizing the margin.

test phase of the SVM does rely only on those data which are the support vectors. In contrast, the rest of the data can be moved around freely without affecting the separating hyperplane. Hence, the solution of the SVM is ideally independent of the rest of the data points.

However, the training phase of finding the solution of the SVM has to make use of the whole training data since the support vectors are unknown in advance. So it is important to have efficient algorithms for solving the linearly constrained quadratic problem arising from the SVM. This class of algorithms include *stochastic gradient descent*, *protected conjugate gradients*, *sub-gradient descent*, and *coordinate descent.* In particular, one well-known coordinate descent algorithm called *Sequential Minimal Optimization* (SMO) [97] is widely used to solve the problem.

SVMs are fundamentally applicable for two-class tasks. In practice, however, multiclass problems are often encountered. This method can be extended to tackle a multiclass problem by combining several binary SVMs: *One-versus-all* classification uses one binary SVM for each class, and then classifies new instances relying on which class has the highest output function, while *one-versus-one* classification uses one binary SVM for each pair of classes, and then classifies new instances relying on which class has the most votes [98].

Available implementations of SVMs include LIBLINEAR [99], LIBSVM [100], WEKA [101]. Note, throughout the thesis, SVMs are the favorite classifier, which is consistently applied to build the final recognition model.

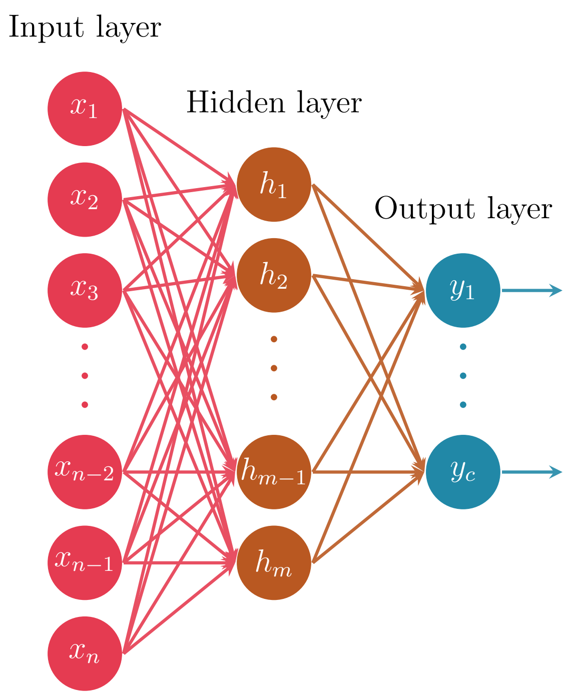


Figure 2.3: A feedforward neural network comprising an input layer, a hidden layer, and an output layer.

2.2.2 Neural Networks

Artificial Neural Networks (ANNs) are another frequently used model for SER. They are also known as *Feedforward Neural Network* (FFNN), *Deep Neural Networks* (DNNs), *Multilayer Perceptrons* (MLPs), or simply *neural networks*. Figure 2.3 visualizes the structure of an FFNN with one hidden layer. Consisting of simple but nonlinear modules, ANNs are amenable to nonlinearly transforming the input of the previous layer into a new space of the next layer. It turns out to be good at learning very complex functions with the combination of a sufficient number of such transformations.

Formally, layer l, where $l = 0, \dots, n_l$, first computes a weighted linear combination of its input vector $\mathbf{h}^{(l-1)}$ from the previous layer, starting with the raw input vector $\mathbf{x} = \mathbf{h}^{(0)}$,

$$\mathbf{z}^{(l)} = \mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}, \tag{2.5}$$

with a vector $\mathbf{b}^{(l)}$ and a matrix $\mathbf{W}^{(l)}$ of adaptive parameters. The vector $\mathbf{b}^{(l)}$ is referred as *biases*, the matrix $\mathbf{W}^{(l)}$ is referred to as *weights*, which are adapted during training. The data entrance to the network is called the *input layer*, supplying the input pattern $\mathbf{x}$ for the network. The results $\mathbf{z}^{(l)}$ are called *activations*. They are then passed through a differentiable and nonlinear *activation function* $f(\cdot)$, such as the sigmoid function (see Section 2.2.2.1), to result in a new representation of the input $\mathbf{h}^{(l-1)}$ in the form,

$$\mathbf{h}^{(l)} = f(\mathbf{z}^{(l)}). \tag{2.6}$$

The layers between the input layers and the last layers are known as the *hidden layers*. With nonlinear activation function, the hidden layers can been viewed as expressing the input in a nonlinear way so that targets become linearly separable by the last layer [102].

Similar to Equation (2.6), the last layer n_l, known as the *output layer*, gives a set of network output

$$\mathbf{y} = \mathbf{h}^{(n_l)} = f(\mathbf{z}^{(n_l)}), \tag{2.7}$$

which is used to make predictions, as well as to link the input patterns $\mathbf{x}$ and the corresponding targets $\mathbf{t}$. The output layer may use an activation function different from the one used in the hidden layers, e.g., the softmax function [85].

An objective function, typically convex in $\mathbf{y}$, describes a measure of the difference between the target vectors and the actual output vectors of the network. Given a training set consisting of a set of input vectors $\{\mathbf{x}_i\}$, where $i = 1, \ldots, N$, along with a corresponding set of target vectors $\{\mathbf{t}_i\}$, the objective (or cost) function can be defined as the *Sum of Squared Error* (SSE)

$$\mathcal{J}(\mathbf{W}, \mathbf{b}) = \sum_{i=1}^{N} \|\mathbf{t}_i - \mathbf{y}_i\|^2, \tag{2.8}$$

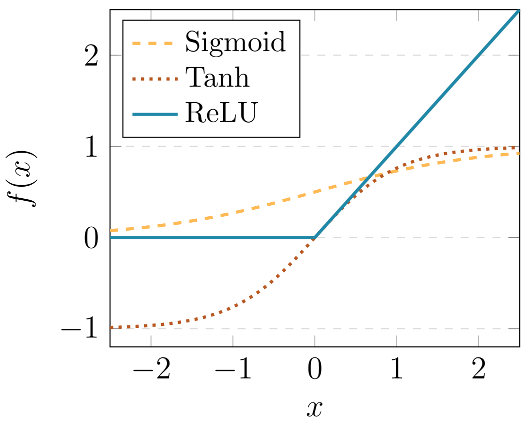
whose value is expected to be minimized during training. Thus, the adaptive parameters $\mathbf{W}$ and $\mathbf{b}$ are determined by minimizing $\mathcal{J}(\mathbf{W}, \mathbf{b})$.

In addition to the SSE, it is very common to use the *Negative Log-Likelihood* (NLL) as the objective function. This is equivalent to a maximum likelihood approach.

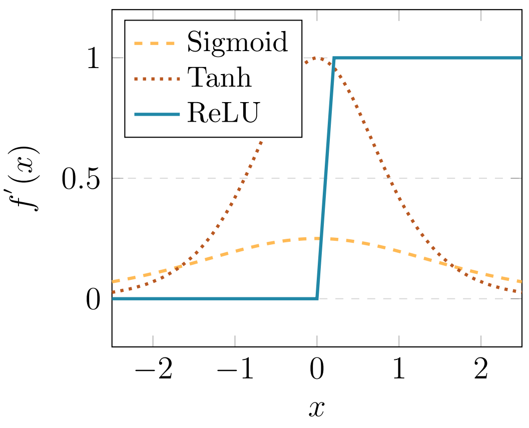
Although both the SSE and NLL objective functions are suited for neural networks, there is a clear contrast between them in use. When the target $\mathbf{t}$ is a discrete label, i.e., for classification problems, the training of the SSE objective function is more robust to outliers than the NLL objective function since the SSE is bounded. It is found that, however, the NLL objective function usually results in faster convergence and a better local optimum [103, 104]. In sum, there is a common choice of an objective function based on the type of the problem to be solved. For a regression problem, the SSE objective function is generally used, for a classification problem, the NLL objective function is often considered.

2.2.2.1 Activation Functions

The neural network model is very powerful due to its nonlinearity property. As described in Equation (2.6), activation functions serve as the element-wise nonlinearity



(a) Activation functions



(b) Derivatives

Figure 2.4: Common activation functions in neural networks, along with their derivatives: sigmoid, hyperbolic tangent (tanh), and rectified linear unit (ReLU).

applied in hidden units of a network. In this section, three types of popular activation functions including the *sigmoid*, *hyperbolic tangent*, and *Rectified Linear Unit* (ReLU) functions are summarized below. Figure 2.4 shows them and the corresponding derivatives.

First, the sigmoid function that is a monotonically increasing function is widely used in neural networks. We often refer to it as the logistic function. Mathematically, the sigmoid(x) function and its derivative are defined as follows

$$f(x) = \frac{1}{(1+\exp^{-x})}, \tag{2.9}$$

$$f'(x) = f(x)(1-f(x)). \tag{2.10}$$

Second, another popular activation function is the hyperbolic tangent function tanh(x). The tanh(x) and its derivative take the following forms

$$f(x) = \frac{\exp^{x} - \exp^{-x}}{\exp^{x} + \exp^{-x}}, \tag{2.11}$$

$$f'(x) = 1 - f(x)^2. \tag{2.12}$$

In practice, the hyperbolic tangent function that is symmetric with respect to the origin (see Figure 2.4) is often recommended because it often converges faster than the sigmoid function [105]. In fact, however, the hyperbolic tangent function has linear relation with the sigmoid: $\tanh(x) = 2\,\mathrm{sigmoid}(2x) - 1$.

Third, the most popular activation function is the ReLU at present, whose definition and derivative are given by

$$f(x) = \max(0, x), \tag{2.13}$$

$$f'(x) = \begin{cases} 1 & x > 0, \\ 0 & x \leq 0. \end{cases} \tag{2.14}$$

The ReLU is onesided and allows a network with many layers to obtain sparse representations in hidden units, further leading to learning much faster than smoother activation functions such as the $\tanh(x)$ [106].

Apart from those three activation functions commonly found in the literature, there are many other neural network activation functions such as the softplus introduced by Dugas et al. [107] and the maxout [108]. Many variants are available. For example, recently, a parametric rectified linear unit is proposed, which adaptively learns the parameters of the rectifiers [109].

2.2.2.2 Backpropagation

For the task of determining a set of weights and biases of a neural network, it is critical to compute the gradient of an objective function $\mathcal{J}(\mathbf{W}, \mathbf{b})$. The use of gradient information can lead to a significant reduction of computational cost. *Backpropagation* (BP) is often used to compute the gradient information because it is relatively simple and powerful [110, 111, 112, 113].

In fact, the BP approach is a practical application of the chain rule, which makes the task of computing the gradient of an objective function with respect to each weight in the network computationally efficient. This whole approach computes the gradients in two passes, namely the *forward pass* and the *backward pass*. For a multilayer network, the forward pass computes the activations of all of the layers by successive use of Equations (2.5) and (2.6). According to the chain rule, the backward pass computes the gradient of the objective with respect to the parameters of the network. Once the forward pass has been complete, the backward pass propagates gradients through all layers, starting at the top layer (i.e., the output layer) and working backwards until the bottom (i.e., the input layer) is reached.

First of all, the intuition of the backward pass is discussed. Given a FFNN with n_l layers and an objective function $\mathcal{J}(\mathbf{W}, \mathbf{b})$, applying the chain rule for the gradient of the objective function with respect to a weight w_{ij} and a bias b_i results in

$$\frac{\partial \mathcal{J}}{\partial w_{ij}} = \frac{\partial \mathcal{J}}{\partial z_i}\frac{\partial z_i}{\partial w_{ij}}, \tag{2.15}$$

$$\frac{\partial \mathcal{J}}{\partial b_i} = \frac{\partial \mathcal{J}}{\partial z_i}\frac{\partial z_i}{\partial b_i}. \tag{2.16}$$

It is very convenient to introduce an *error term*

$$\delta_i \overset{\text{def}}{=} \frac{\partial \mathcal{J}}{\partial z_i}. \tag{2.17}$$

Then, substituting Equation (2.17) into Equation (2.15) and (2.16) has

$$\frac{\partial \mathcal{J}}{\partial w_{ij}} = \delta_i \frac{\partial z_i}{\partial w_{ij}}, \tag{2.18}$$

$$\frac{\partial \mathcal{J}}{\partial b_i} = \delta_i \frac{\partial z_i}{\partial b_i}, \tag{2.19}$$

which imply that the desired gradients are achieved by multiplying the value of δ by $\frac{\partial z_i}{\partial w_{ij}}$ or $\frac{\partial z_i}{\partial b_i}$. Thus, for the evaluation of the gradients, it is needed to compute the value of δ for each hidden and output nodes in the network, and then apply Equations (2.18) and (2.19).

Following this intuition, the backward pass starts with the output nodes and the error term is obtained

$$\delta_i^{(n_l)} = \frac{\partial \mathcal{J}}{\partial h_i^{(n_l)}} f'(z_i^{(n_l)}), \tag{2.20}$$

where the fact $h_i^{(n_l)} = y_i$ is applied.

For each hidden layer $l = n_l - 1, n_l - 2, \ldots, 1$, the error term is given by

$$\begin{aligned}\delta_i^{(l)} = \frac{\partial \mathcal{J}}{\partial z_i^{(l)}} &= \frac{\partial \mathcal{J}}{\partial h_i^{(l)}} \frac{\partial h_i^{(l)}}{\partial z_i^{(l)}} \\ &= \sum_j \left(\frac{\partial \mathcal{J}}{\partial z_j^{(l+1)}} \frac{\partial z_j^{(l+1)}}{\partial h_i^{(l)}} \right) f'(z_i^{(l)}) \\ &= \sum_j \left(w_{ji}^{(l+1)} \delta_j^{(l+1)} \right) f'(z_i^{(l)}). \end{aligned} \tag{2.21}$$

In the end, the required gradients are written as

$$\frac{\partial \mathcal{J}}{\partial w_{ij}^{(l+1)}} = h_j^{(l)} \delta_i^{(l+1)}, \tag{2.22}$$

$$\frac{\partial \mathcal{J}}{\partial b_i^{(l+1)}} = \delta_i^{(l+1)}, \tag{2.23}$$

where $l = 0, \ldots, n_l - 1$. Because the calculation of the value of the error term δ needs the derivative $f'(\cdot)$ with respect to its input, the activation function $f(\cdot)$, which is adopted for neural networks, is differentiable. The most commonly used activation functions and their derivatives are presented in Section 2.2.2.1.

Algorithm 2.1 *Backpropagation* (BP) for multilayer *Feedforward Neural Networks* (FFNNs) using matrix-vectorial notation

```
1: Perform a forward pass, computing the activations of all of the hidden and output
   units, using Equation (2.5) and (2.6).
2: For the output layer $n_l$, evaluate
       $$\delta^{(n_l)} = \frac{\partial \mathcal{J}}{\partial \mathbf{h}^{(n_l)}} \circ f'(\mathbf{z}^{(n_l)}). \qquad (2.25)$$
3: For each hidden layer $l = n_l - 1, n_l - 2, \ldots, 1$, recursively compute
       $$\delta^{(l)} = \left( \left( \mathbf{W}^{(l+1)} \right)^T \delta^{(l+1)} \right) \circ f'(\mathbf{z}^{(l)}). \qquad (2.26)$$
4: Evaluate the gradients of $\mathcal{J}$ w.r.t. $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$, $l = 0, 1, \ldots, n_l - 1$,
       $$\frac{\partial \mathcal{J}}{\partial \mathbf{W}^{(l+1)}} = \delta^{(l+1)} (\mathbf{h}^{(l)})^T, \qquad (2.27)$$
       $$\frac{\partial \mathcal{J}}{\partial \mathbf{b}^{(l+1)}} = \delta^{(l+1)}. \qquad (2.28)$$
```

In practice, matrix multiplication is likely to speed up learning of a network. For that consideration, the BP is shown in Algorithm 2.1 using matrix-vectorial notation, in which the error term in matrix form is given by

$$\delta \stackrel{\text{def}}{=} \frac{\partial \mathcal{J}}{\partial \mathbf{z}}, \qquad (2.24)$$

and the symbol "$\circ$" denotes the element-wise product operator, also known as the Hadamard product.

Gradient Checking

When analytic gradients are computed using the BP algorithm, in practice, the results should be compared with the numerical gradients so as to ensure the correctness of the software implementation. This procedure is often called *gradient checking.*

The numerical gradient is usually computed by the *symmetrical central differences* of the form

$$\frac{\partial \mathcal{J}}{\partial w_{ij}} = \frac{\mathcal{J}(w_{ij} + \epsilon) - \mathcal{J}(w_{ij} - \epsilon)}{2\epsilon}. \qquad (2.29)$$

The value of variable ϵ has a strong effect on the numerical accuracy of the results obtained using Equation (2.29). For the models used in this thesis, $\epsilon = 10^{-4}$ is chosen since it always gave satisfying performance.

2.2.2.3 Gradient Descent Optimization

Neural networks can be simply viewed as a class of nonlinear functions from a set of input variables to a set of output variables specified by a set of weights and biases. On the one hand, neural networks are very good at disentangling the nonlinear information. On the other hand, there is little hope of finding a global minimum for the objective function because the objective function has a highly nonlinear dependence on the weights and biases. For these reasons, neural network learning becomes difficult.

Gradient descent is often used to repeatedly adjust the weights in a small step towards the direction of the negative gradient

$$\mathbf{W}^{(\tau+1)} = \mathbf{W}^{(\tau)} - \eta \nabla \mathcal{J}(\mathbf{W}^{(\tau)}), \tag{2.30}$$

where τ indicates the iteration step and the variable $\eta > 0$ is the *learning rate.*

In each step, *batch* methods use the whole training data at once to compute the gradients $\nabla \mathcal{J}(\mathbf{W}^{(\tau)})$ and then update the weights, which have been found very useful for training neural networks. Many sophisticated batch optimization methods such as *conjugate gradient* and *Limited-memory Broyden-Fletcher-Goldfarb-Shanno* (L-BFGS) [114], have proved more efficient and much more stable than standard gradient descent [115]. The intuition behind these algorithms is to compute an approximation to the Hessian matrix, so that it can take more rapid steps towards a local optimum.

The *Stochastic Gradient Descent* (SGD), however, an online version of gradient descent, is the predominant optimization method for training neural networks on large datasets [88]. In this method, an update to the weights is based on the gradient value of the objective for one example only. This update is repeated for a number of small sets of examples selected from the training data. The simple procedure is often surprisingly fast, resulting in a good set of weights and scales easily with the number of training examples when compared with more elaborate optimization methods [116].

The most obvious drawback of neural networks is that they are very prone to get stuck in poor *local minima* [117]. It turns out that there are many serious problems, such as the *overfitting* and the *vanishing gradient* problems [118] , during training with the gradient descent procedure. Hence, particular care must be taken to ensure that the procedure converges fast as well as finds a good set of parameters which are more negligible than the global minimum. A number of simple but useful tricks that can be used to facilitate neural network learning are introduced as follows.

A trick that is often used to address the overfitting problem is that of regularization, which explicitly adds a penalty term to an objective function in order to regularize the behavior of the weights towards the desired direction. One of the simplest forms of regularizer is given by the sum-of-squares of the weight matrix elements, for example, the objective function of the SSE (see Equation (2.8)) turns to have the form

$$\mathcal{J}(\mathbf{W},\mathbf{b}) = \sum_{i=1}^{N} \|\mathbf{t}_i - \mathbf{y}_i\|^2 + \frac{\lambda}{2}\sum_{l=1}^{n_l} \|\mathbf{W}^{(l)}\|^2, \tag{2.31}$$

where the penalty term $\lambda > 0$. This method is well known as *weight decay* or L^2 weight decay, leading the weight values towards zero. In this case, the weights' update for the BP procedure is accordingly rewritten as

$$\frac{\partial \mathcal{J}}{\partial \mathbf{W}^{(l)}} = \delta^{(l+1)}(\mathbf{h}^{(l)})^T + \lambda \mathbf{W}^{(l)}. \tag{2.32}$$

Besides, an L^1 penalty that induces the sparsity property can be applied to help control overfitting [119].

Another trick frequently used to speed up the convergence of the neural network training is called the *momentum* update [120]. The intuition behind the momentum update is to accumulate a velocity vector in directions of persistent reduction in the objective across iterations. The momentum update is given by

$$\mathbf{W}^{(\tau+1)} = \mathbf{W}^{(\tau)} - \eta\nabla\mathcal{J}(\mathbf{W}^{(\tau)}) + \mu\left(\mathbf{W}^{(\tau)} - \mathbf{W}^{(\tau-1)}\right), \tag{2.33}$$

where $\eta > 0$ is the learning rate and μ is the momentum term. The term μ is usually set to values such as $\{0.5, 0.9, 0.95, 0.99\}$

With the increasing growth of deep learning, a variant of the momentum update has recently been widely used, namely the *Nesterov momentum* update [121, 122]. The Nesterov momentum update is written as

$$\mathbf{W}^{(\tau+1)} = \mathbf{W}^{(\tau)} - \eta\nabla\mathcal{J}\left(\mathbf{W}^{(\tau)} + \mu\left(\mathbf{W}^{(\tau)} - \mathbf{W}^{(\tau-1)}\right)\right) + \mu\left(\mathbf{W}^{(\tau)} - \mathbf{W}^{(\tau-1)}\right). \tag{2.34}$$

Inspired by convex optimization, it has better convergence rate and seems to work more effectively for optimizing some types of neural networks in practice than the momentum update described above [121, 122].

A powerful trick that is frequently used to control overfitting at present is *dropout* [123, 124]. This randomly drops units with probability q during training so as to prevent units from co-adaptation too much. Dropout can be viewed alternatively as a process of constructing new inputs by multiplying noise [119]. The probability q is tunable but is usually set to either 0.2 or 0.5. It is observed that, dropout is more effective than other common regularizers, such as the aforementioned weight decay and L^1 penalty regularization. Also combining dropout with unsupervised pretraining may result in an improvement.

2.2.2.4 Deep Learning

Neural networks have been gaining popularity and have been rebranded as *deep learning* in the machine learning community since greedy layer-wise unsupervised pre-training was first proposed to train very deep neural networks in 2006 [125, 126, 127]. These methods have dramatically advanced the state-of-the-art in speech processing [128, 129, 130, 131], image recognition [132], object detection [133], drug discovery [134], natural language understanding [135], language translation [136], and paralinguistic tasks [40, 137]. Naturally, deep learning is a very broad family of machine learning methods.

The wide-spreading success of deep learning has substantially encouraged both industry and academics across the board. In speech recognition, many of the major commercial speech recognition systems (e.g., Microsoft Cortana, Xbox, Skype Translator, Apple Siri, and iFlyTek voice search) heavily depend on deep learning methods. Such large successes of deep learning also happened in image recognition. The real impact of deep learning in image recognition emerged when deep CNNs won the ImageNet Large-Scale Visual Recognition Challenge 2012 by a large margin over the then-state-of-the-art shallow learning methods [132]. Since then, the error on the ImageNet task was further reduced at a rapid rate by different deep nets with large amount parameters such as GoogLeNet[138]. Performance obtained by using deep learning, on the ImageNet task, is close to that of humans [109]. Besides, deep learning has been the most prevalent in image-related classification.

Whereas supervised learning has been the workhorse of recent successes of deep learning, unsupervised learning also plays a key role in the renewed interest of deep learning. For small datasets, unsupervised learning is used to initialize neural networks, allowing to train a deep supervised network. By that, it can prevent overfitting and often yields notable performance gains when compared to models without using unsupervised learning. This recipe is referred to as *greedy layerwise unsupervised pre-training* [126, 127, 139]. Moreover, it is highly believed that unsupervised learning will become far more important in the future of deep learning [102].

Advances in hardware have also been an important contributing factor for deep learning. In particular, fast *Graphics Processing Units* (GPUs) are tailored to suit the needs of matrix/vector computation involved in the forward pass and the backward pass of deep learning. GPUs have been shown to train networks 10 or 20 times faster, leading training times of weeks back to days. Most popular deep learning software frameworks, such as Caffe [140], Theano [141], or CURRENNT [142], support parallel GPU computing.

Up to now, in this section, the key topics needed to understand neural networks, such as activation functions (see Section 2.2.2.1), BP (see Section 2.2.2.2), and gradient descent (see Section 2.2.2.3), have been briefly discussed. On the basis of them, this thesis will present several novel feature transfer learning methods which are exemplified by SER. Other important topics of neural networks, especially such as *Convolutional Neural Networks* (CNNs) [88], *Recurrent Neural Networks* (RNNs) [143], *Long Short-Term Memory* (LSTM) networks [144], *Restricted Boltzmann Machines* (RBMs) [125], are not given in this thesis. The interested reader is encouraged to refer to good surveys on the subject [102, 145, 146, 147, 148].

2.3 Classification Evaluation

In this section, the focus is placed on classification evaluation methods, which provide a way of judging the quality of different classification systems. Evaluation criteria are normally obtained by comparing discrete predicted class labels with the ground truth targets [20]. Without loss of the general case, the classification task mathematically can be seen as a mapping f from a vector $\mathbf{x}$ to a scalar $y \in \{1, \ldots, c\}$

$$f : \mathbf{X} \to \{1, \ldots, c\} \quad \mathbf{x} \mapsto y. \tag{2.35}$$

After training, the classification system is evaluated on a different set of examples, which is known as a test set. Given a test set $\mathbf{X}^{te}$, each test example is labeled to one target class $t \in \{1, \ldots, c\}$, so the test set satisfies the following condition

$$\mathbf{X}^{te} = \bigcup_{t=1}^{c} \mathbf{X}_t^{te} = \bigcup_{t=1}^{c} \{\mathbf{x}_{t,n} \mid n = 1, \ldots, N_t\}, \tag{2.36}$$

where N_t is the number of examples in the test set that belong to class t, leading to the size of the test set $|\mathbf{X}^{te}| = \sum_{t=1}^{c} N_t$.

In the general case of two or more class classification problems (i.e., $c \geq 2$), the most frequently used measure is the overall probability that a test example is classified correctly [149], which is given by

$$\begin{aligned} \text{Acc} &= \frac{\#\,\text{correctly classified test examples}}{\#\,\text{test examples}} \\ &= \frac{\sum_{t=1}^{c} |\{\mathbf{x} \in \mathbf{X}_t^{te} \mid y = t\}|}{|\mathbf{X}^{te}|}. \end{aligned} \tag{2.37}$$

This is known as *accuracy* (Acc) or simply *recognition rate.*

Another common measure is called *recall* that evaluates the class-specific performance. Analogous to Equation (2.37), the recall has the dependence on the examples only with class t

$$\text{Recall}_t = \frac{|\{\mathbf{x} \in \mathbf{X}_t^{te} \mid y = t\}|}{N_t}. \tag{2.38}$$

It is usually desired to take the distribution of all classes into consideration when evaluating the general performance of a classification system. Let $p_t = N_t/|\mathbf{X}^{te}|$

denote the prior probability of class t in the test set, the *Weighted Average Recall* (WAR) is given by

$$\text{WAR} = \sum_{t=1}^{c} p_t \, \text{Recall}_t. \tag{2.39}$$

Further, if the distribution of examples among classes is highly imbalanced, one may prefer to replace the priors p_t for all classes by the constant weight $\frac{1}{c}$. This is known as *Unweighted Average Recall* (UAR) or *unweighted accuracy*

$$\text{UAR} = \frac{\sum_{t=1}^{c} \text{Recall}_t}{c}. \tag{2.40}$$

It is worth noting that, the UAR is often used as the officially-recommended measure for paralinguistic tasks [25, 26, 27]. For this reason, the UAR is adopted as the primary metric to evaluate the recognition performance in this thesis.

2.4 Significance Tests

Apart from these above measures to compare different systems, it is often of interest to further estimate the p-value obtained by significance tests, to show whether system B performing *significantly* better than system A is due to 'luck'. The following section briefs on one frequently used type of significance tests for classification tasks.

The *z-test* as a simple variant of the binomial test described by Dietterich [150] is widely used to assess whether the accuracies of two recognition systems A and B are significantly different. Let p_A and p_B denote the probabilities of correct classification, and without loss of generality, assume that $p_B > p_A$. Then, one gives a hypothesis that the observed performance differences are the results of identical random processes from the average probability of correct classification, $p_{AB} = (p_A + p_B)/2$, and rejects this hypothesis at a given level of significance.

If a random variable N_c denotes the number of correct classifications on the test set, then under the null hypothesis N_c follows a binomial distribution with probability p_{AB}

$$N_c \sim \text{Bin}(N, p_{AB}), \tag{2.41}$$

where $\text{Bin}(\cdot)$ denotes the binomial distribution function and N is the number of examples in the test set.

Since the binomial distribution can be approximated by a normal distribution with the estimated average Np_{AB} and variance $Np_{AB}(1 - p_{AB})$, the standard score z is given by

$$z = \frac{N_c - Np_{AB}}{\sqrt{Np_{AB}(1 - p_{AB})}} = \frac{p_B - p_{AB}}{\sqrt{p_{AB}(1 - p_{AB})}} \sqrt{N}, \tag{2.42}$$

where p_B is substituted for N_c/N.

Given the probability of observing the improved accuracy of B

$$P(N_c > p_B N) = 1 - P(N_c \leq p_B N), \tag{2.43}$$

one-tailed *p-values* are calculated as

$$p = 1 - \Phi(z) < \alpha, \tag{2.44}$$

for the significance level α and the standard normal cumulative distribution function $\Phi(\cdot)$.

In general, the p-value represents the probability of rejecting the null hypothesis. For example, if the p-value is lower than α (typically set as $\alpha < 0.05$ in practice), then one disproves the null hypothesis and therefore concludes there are significant performance differences in the above case, i.e., system B is better than system A.

One remarkable advantage of the z-test is that its calculation is very simple and easy, only depending on the accuracies of both systems and the size of the test set. However, the z-test is likely to overestimate significance [150]. In spite of that, the z-test is among the most broadly used. Throughout the thesis, the z-test is adopted and the significance level α is 0.05 unless stated otherwise.

3

Feature Transfer Learning

This chapter presents the topic of feature transfer learning. This can be extremely useful in practice when the training set and the test set present a distribution mismatch. This chapter starts by describing the distribution mismatch (Section 3.1), which causes an adverse effect on classification. Then, it moves on to introducing transfer learning and gives an overview of the related work with particular focus on importance weighting methods and domain adaptation in speech processing (Section 3.2). Next, a coarse framework of feature transfer leaning is laid out in Section 3.3. Based on the major purpose of this thesis, i.e., reducing the problem of a distribution mismatch, a variant of novel autoencoder-based feature transfer learning methods is discussed in Section 3.4.

3.1 Distribution Mismatch

Many traditional machine learning methods may live up to expectations due to one common assumption that training examples are drawn according to the same feature space and the same probability distribution as the unseen test examples. This assumption is important because it permits the estimation of the generalization error and the uniform convergence theory gives essential guarantees on the accurate classification. In real life, however, this common assumption rarely holds. By contrast, one is often faced with the situations in which the training data are different from that of unknown test data. The difference between the training and test data is known as the *distribution mismatch* or *dataset bias*.

With the availability of rich data obtained from different devices and varied acquisition conditions, unfortunately, a distribution mismatch happens in both speech processing and image processing. For example, a common technique in speech recognition is *speaker adaptation* which consists in adapting one previously trained model to a new speaker (or even a group of speakers). In object recognition, it is found that popular image datasets contain the existence of different types of built-in bias such as selection bias, capture bias, category or label bias, and negative set bias [151, 152]. Besides, mismatches arise even when image datasets are neutrally composed of images from the same visual categories. The mismatches can be due to many external factors in image data collection, such as cameras, labels, preferences over certain types of backgrounds, or annotator tendencies. In automatic dialog act tagging, classifiers are trained on labeled sets, but then applied on new utterances from different genre and/or language [153]. In cross-corpus emotion recognition from speech, a classification engine trained on one corpus is evaluated by another which may differ from labeling concepts and interaction scenarios. In the above two cases, it is natural to observe the distribution mismatch.

In more general terms, we are all used to knowing the situation in which one has a large number of labeled examples on a task drawn from one certain domain (the *source domain* or the *auxiliary domain* in some studies), while one needs to solve the same task on a domain of interest (the *target domain*) with few or even no labeled data. In this situation, rather than tediously collecting and labeling data and building a system from scratch, it is desirable to effectively take advantage of examples from both domains, no matter how different the two domains might be.

3.2 Transfer Learning

This thesis takes advantage of *transfer learning* (also referred to as *domain adaptation*) to address the general problem (i.e., the distribution mismatch) by leveraging over prior knowledge found in one source when facing a new target task. The insight behind transfer learning is that prior experience gained in learning to perform one task can help with a related, but different task. The research topic of transfer learning has long been studied in the psychological literature [154, 155]. It was found that people appear to have the ability to transfer aspects of their prior knowledge to guide their behavior in new settings. Similarly, researchers in the community of machine leaning have put considerable efforts into replicating such transfer ability in an artificial intelligence machine [15, 16].

The goal of transfer learning is to provide performance improvements in the target task due to knowledge from the source task. There are three common types of performance improvements along with the increase in the number of target training examples [15], illustrated in Figure 3.1:

1. Higher start: the initial performance achievable in the target task is higher than learning from the target task alone.

2. Higher slope: performance grows more rapidly than learning from scratch.

3. Higher asymptote: the final performance level is better compared to the final level without transfer.

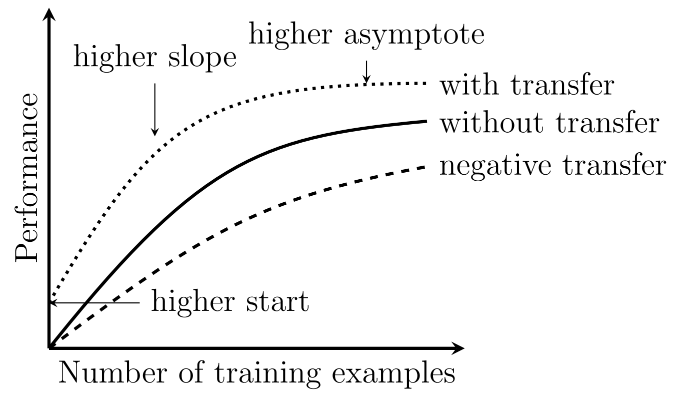


Figure 3.1: Three ways in which transfer learning might provide the performance improvements along with the increase in the number of target training examples. The negative transfer occurs when a transfer method is forced to learn unrelated sources. (The figure is taken from [15])

.

In addition to those benefits, a transfer learning method might even hurt performance, which is called *negative transfer*. For a target task, the effectiveness of a transfer learning method relies on the source task and the similarity between the source and the target. If the similarity is close and the transfer learning method can use it, the performance in the target task can dramatically improve through transfer. However, if the source task is not sufficiently related or if the similarity is not well exploited by the transfer learning method, the performance may not simply fail to improve, but it may decline. Hence, one of the major challenges in developing transfer learning methods is to achieve positive transfer between appropriately related tasks while preventing negative transfer between tasks that are less related [15, 16].

Transfer learning is strongly related to *multitask learning*, which is to optimize the performance over multiple tasks simultaneously by improving generalization of a model from related tasks [156]. In order to improve performance of a categorical emotion recognition task, for example, the multitask learning framework using the deep belief network was applied to leverage the information of two related tasks (arousal and valence) [157]. Transfer learning can be even regraded as a particular form of multitask learning. For example, the shared-hidden-layer autoencoder model (Section 3.4.5) is an instance of multitask learning that makes transfer learning successful with neural networks in general. Although multitask learning is typically used as a supervised learning method, the more general term of transfer learning is considered in the context of unsupervised learning and reinforcement learning [158] as well.

An extreme case of transfer learning is *one-shot learning* [159], where a new task may be learned from a single training example (or just a few). One-shot learning is possible because a model that has previously learned provides the model information that helps it to learn new tasks with fewer training examples. A related case of one-shot learning is *zero-shot learning* [160], which tackles a more extreme situation where no training examples are available.

Self-labeling approaches are popular among the various ways of transfer learning. Self-labeling includes *self-training* and *co-training*. Generally speaking, these are iterative methods that train an initial model based on the labeled source data, use that to recognize the target data, then use the recognized labels for retraining the model. The learning process involves the prediction work from the machine oracle (e.g., the prediction uncertainty), the human oracle (e.g., the label uncertainty or human agreement level), and the combinations thereof. Recently, the effectiveness of self-labeling was empirically investigated in SER [161]. Active learning by label uncertainty was found very efficient to reduce human labeling effort when building a classifier for acoustic emotion recognition [55]. Zhang et al. [162] successfully introduced co-training for the purpose of exploiting unlabeled data in acoustic emotion recognition, where co-training is used to build two learners by maximizing the mutual agreement on two distinct 'views' of the unlabeled data set. More recently, a cooperative learning was proposed in order to take advantage of active learning and self-training [163]. The cooperative learning method allows sharing the labeling effort between human and machine oracles while easing the drawbacks of active learning and self-training.

Several research projects have made use of transfer learning. One of the famous such projects is the FP7 ERC starting grant project iHEARu[1]. It absorbs transfer leaning and then attempts to develop a universal sound analysis system for computational paralinguistics, which can be easily learned and can be adapted to new, previously unexplored characteristics [164].

In general, transfer learning techniques are categorized into two classes depending on whether the target domain data are either partially labeled or completely unlabeled. If there are a small amount of labeled target data, which are drawn from the same distribution of the test data, the problem closely resembles semi-supervised learning and is called *semi-supervised transfer learning*. In semi-supervised transfer learning, correspondences of labeled target data are often used to learn domain transformations [165]. On the other hand, if no labeled target data are available, the problem is known as *unsupervised transfer learning* like unsupervised learning. Unsupervised transfer learning uses strategies which assume a known class of transformations between the domains, the availability of discriminative features which are common to or invariant across both domains, a latent space where the difference in distribution of source and target data is minimal [11, 166], and a mapping 'path' by which the domain transformation maps the source data onto the target domain [167].

[1]`http://www.ihearu.eu/`

3.2.1 Importance Weighting for Domain Adaptation

A naive approach to alleviating the dataset bias problem is to assign more weight to those training examples that are most similar to the test data, and less weight to those that poorly reflect the distribution of the target (test) data. This idea of weighting the input data based on the test data is known as *Importance Weighting* (IW) for covariate shift. The goal is to estimate importance weights β, from training examples $\{\mathbf{x}_i^{tr}\}_{i=1}^{n_{tr}}$ and test examples $\{\mathbf{x}_i^{te}\}_{i=1}^{n_{te}}$ by taking the ratio of their densities

$$\beta(\mathbf{x}) = \frac{p_{te}(\mathbf{x})}{p_{tr}(\mathbf{x})}, \tag{3.1}$$

where $p_{te}(\mathbf{x})$ and $p_{tr}(\mathbf{x})$ are test and training input densities [168, 169, 170]. Normally, the IW optimization is formulated as a convex optimization model, which can be efficiently solved by gradient descent and feasibility satisfaction iteratively.

For example, Kanamori et al. proposed *Unconstrained Least-Squares Importance Fitting* (uLSIF) to estimate the importance weights by a linear model [168]. A similar idea called *Kullback-Leibler Importance Estimation Procedure* (KLIEP) is proposed in [169], where the importance function is formulated as a linear or kernel model, such as Gaussian kernels, resulting in a convex optimization problem with a sparse solution. Its goal is to estimate weights to maximize similarity between the test and weight-corrected training distributions, but distribution similarity is formulated with respect to *Kullback-Leibler* (KL) divergence. In addition, *Kernel Mean Matching* (KMM) gave a straightforward way to estimate the importance weights, in order to lead the weighted training distribution towards the test distribution [171]. In doing so, distribution similarity is measured as the disparity in the weighted example means of the data mapped in a reproducing kernel Hilbert space.

The three methods have recently been shown to lead to significant improvement in acoustic emotion recognition when Hassan et al. first considered to explicitly compensate for acoustic and speaker differences between training and test databases [170]. For this reason, the three methods serve as the models for fair comparison in the validation phase. Here, the characteristics of these methods are shortly introduced. For more details on these algorithms, the interested reader is referred to [168, 169, 171, 172, 173].

3.2.1.1 Kernel Mean Matching

Kernel Mean Matching (KMM) was proposed to deal with dataset bias by inferring the importance weight directly by distribution matching of training and test sets in feature space in a non-parametric way [171, 172]. As a result, the means of the training and test examples in a reproducing kernel Hilbert space are close. The objective function is given by the discrepancy term between the two empirical means

$$\mathcal{J} = \left\| \frac{1}{n_{tr}} \sum_{i=1}^{n_{tr}} \beta_i \Phi(\mathbf{x}_i^{tr}) - \frac{1}{n_{te}} \sum_{i=1}^{n_{te}} \beta_i \Phi(\mathbf{x}_i^{te}) \right\|^2, \tag{3.2}$$

where $\Phi(\cdot)$ are the *canonical* mapping functions.

Using $K_{ij} \overset{\text{def}}{=} k(x_i^{tr}, x_j^{tr})$ and $\kappa_i \overset{\text{def}}{=} \frac{n_{tr}}{n_{te}} \sum_{j=1}^{n_{te}} k(x_i^{tr}, x_j^{te})$, the objective function above can be regarded as the quadratic problem for finding suitable β

$$\begin{aligned} &\arg\min_\beta \tfrac{1}{2}\beta^T K \beta - \kappa^T \beta \\ &s.t.\, \beta_i \in [0, B] \text{ and } \left| \sum_{i=1}^{n_{tr}} \beta_i - n_{tr} \right| \leq n_{tr}\epsilon, \end{aligned} \tag{3.3}$$

where the upper limit of importance weight $B > 0$ and $\epsilon > 0$ are tuning parameters, and k is the kernel function. Since KMM optimization is formulated as a convex problem, it can be solved efficiently using interior point methods or any other successive optimization procedure and leads to a unique global solution. It can be seen from the above that the solution β depends only upon the input training and the test data without any requirement for estimating the true distributions. An advantage of KMM, hence, is that it may avoid the curse of dimensionality.

In practice, a Gaussian kernel is typically chosen as the kernel in the learning algorithm. Thus, there are three parameters that need to be tuned for the algorithm: the Gaussian kernel width σ , the upper limit of importance weight B, and ϵ. To reduce the great effort at tuning these parameters, it is suggested to use values of $\sigma = 0.1, B = 1000$, and $\epsilon = (\sqrt{n_{tr}} - 1/\sqrt{n_{tr}})$ are used.

3.2.1.2 Unconstrained Least-Squares Importance Fitting

Unconstrained Least-Squares Importance Fitting (uLSIF) formulates the direct importance estimation problem as a least-squares function fitting problem [168, 173]. The resulting formulation can be seen as a convex quadratic problem, which can be efficiently solved. Specifically, the importance $\beta(\mathbf{x})$ is formulated by the following linear mode in the form

$$\hat{\beta}(\mathbf{x}) = \alpha^T \Phi(\mathbf{x}), \tag{3.4}$$

where $\alpha = (\alpha_1, \ldots, \alpha_b)^T$, is a parameter to be learned, b is the number of parameters, $\Phi(\mathbf{x}) = (\Phi_1(\mathbf{x}), \ldots, \Phi_b(\mathbf{x}))^T$ are basis functions so that $\Phi(\mathbf{x}) \geq 0$.

To derive the parameters α, the following squared error is minimized

$$\begin{aligned} \mathcal{J} &= \frac{1}{2} \int \left(\hat{\beta}(\mathbf{x}) - \frac{p_{te}(\mathbf{x})}{p_{tr}(\mathbf{x})} \right)^2 p_{tr}(\mathbf{x})\, \mathrm{d}\mathbf{x} \\ &= \frac{1}{2} \int \frac{1}{2} \hat{\beta}(\mathbf{x})^2 p_{tr}(\mathbf{x})\, \mathrm{d}\mathbf{x} - \int \hat{\beta}(\mathbf{x}) p_{te}(\mathbf{x})\, \mathrm{d}\mathbf{x} + C, \end{aligned} \tag{3.5}$$

where $C = \frac{1}{2}\int \beta(\mathbf{x})p_{te}(\mathbf{x})\,\mathrm{d}\mathbf{x}$ is a constant and is ignored in the objective function. The squared error above is further rewritten as

$$\mathcal{J} = \frac{1}{2}\alpha^T\mathbf{H}\alpha - \mathbf{h}^T\alpha, \tag{3.6}$$

where

$$\mathbf{H} = \int \Phi(\mathbf{x})\Phi(\mathbf{x})^T\,\mathrm{d}\mathbf{x}, \tag{3.7}$$

$$\mathbf{h} = \int \Phi(\mathbf{x})p_{te}(\mathbf{x})\,\mathrm{d}\mathbf{x}. \tag{3.8}$$

Using the empirical approximation, one obtains the unconstrained formulation

$$\arg\min_{\alpha} \frac{1}{2}\alpha^T\hat{\mathbf{H}}\alpha - \hat{\mathbf{h}}^T\alpha + \frac{\mu}{2}\alpha^T\alpha, \tag{3.9}$$

where $\hat{\mathbf{H}}$ is the $b \times b$ matrix and $\hat{\mathbf{h}}$ is the b-dimensional vector

$$\hat{\mathbf{H}} = \frac{1}{n_{tr}}\sum_{i}^{n_{tr}} \Phi(\mathbf{x}_i^{tr})\Phi(\mathbf{x}_i^{tr})^T, \tag{3.10}$$

$$\hat{\mathbf{h}} = \frac{1}{n_{te}}\sum_{i}^{n_{te}} \Phi(\mathbf{x}_i^{te}), \tag{3.11}$$

and μ is a quadratic regularization term.

In contrast to KMM, an advantage of the above unconstrained formulation is that the solution can be efficiently computed by solving a system of linear equations. Therefore, the computation is fast and stable.

3.2.1.3 Kullback-Leibler Importance Estimation Procedure

The *Kullback-Leibler Importance Estimation Procedure* (KLIEP) also directly gives an estimate of the importance function by using the divergence between the importance-weighted test distribution and the true test distribution in terms of KL divergence [169].

Based on the basic importance model (see Equation (3.1)), an estimate of the test density $p_{te}(\mathbf{x})$ is given by

$$\hat{p}_{te}(\mathbf{x}) = \hat{\beta}(\mathbf{x})p_{tr}(\mathbf{x}). \tag{3.12}$$

In KLIEP, the parameters α are determined so that the KL divergence from $p_{te}(\mathbf{x})$ to $\hat{p}_{te}(\mathbf{x})$ is minimized

$$\begin{aligned}\mathrm{KL}(p_{te}(\mathbf{x})\|\hat{p}_{te}(\mathbf{x})) &= \int p_{te}(\mathbf{x}) \log \frac{p_{te}(\mathbf{x})}{\hat{\beta}(\mathbf{x})p_{tr}(\mathbf{x})}\, \mathrm{d}\mathbf{x} \\ &= C - \int p_{te}(\mathbf{x}) \log \hat{\beta}(\mathbf{x})\, \mathrm{d}\mathbf{x}, \qquad (3.13)\end{aligned}$$

where $C = p_{te}(\mathbf{x}) \log \frac{p_{te}(\mathbf{x})}{p_{tr}(\mathbf{x})}\, \mathrm{d}\mathbf{x}$ is independent of α.

Ignoring the constant term and using the linear model (see Equation (3.4)), one defines the objective function

$$\begin{aligned}\mathcal{J} &= \int p_{te}(\mathbf{x}) \log \hat{\beta}(\mathbf{x})\, \mathrm{d}\mathbf{x} \\ &\approx \frac{1}{n_{te}} \sum_{i=1}^{n_{te}} \log \hat{\beta}(\mathbf{x}_i^{te}) \\ &= \frac{1}{n_{te}} \sum_{i=1}^{n_{te}} \log \left(\sum_{l=1}^{b} \alpha_l \Phi_l(\mathbf{x}_i^{te}) \right). \qquad (3.14)\end{aligned}$$

Using the empirical approximation based on the training examples, the KLIEP optimization problem is given by

$$\begin{aligned}&\arg\max_{\alpha} \sum_{i=1}^{n_{te}} \log \left(\sum_{l=1}^{b} \alpha_l \Phi_l(\mathbf{x}_i^{te}) \right) \\ &\qquad s.t. \sum_{j=1}^{n_{tr}} \sum_{l=1}^{b} \alpha_l \Phi_l(\mathbf{x}_j^{tr}) \text{ and } \alpha \geq 0, \qquad (3.15)\end{aligned}$$

which is a convex optimization problem and leads to the global solution. The solution α tends to be sparse, which contributes to speeding up the test phase.

3.2.2 Domain Adaptation in Speech Processing

ASR is faced with many similar mismatch problems, and the speech community has done a considerable amount of related work to reduce the mismatch problem. In return, ASR systems are capable of adapting to new environments and unknown target speakers.

Vocal Tract Length Normalization (VTLN) is a frequently-used way of doing fast speaker adaptation [174]. The underlying aim is to cushion the effect of vocal length variation on speakers. Based on the observation that the effect of vocal tract length can be modeled well by a linear warping of the frequency axis, it warps the frequency axis of the acoustic features by a speaker-specific warping factor, usually towards a global average vocal tract length. The resulting spectral estimates are more homogeneous across speakers and thus more suitable for speech processing. However, its performance is somehow confined to a manually designed frequency warping function. Interestingly, VTLN may have different usage. A method, called vocal tract length perturbation, uses VTLN to augment mel log filter bank training data [175]. In this method, VTLN is used to generate random distortions, creating an augmented dataset. In turn, a learner trained on the augmented data can learn to be invariant to vocal tract length differences.

A technique of speaker adaptation that estimates a set of transformations to mitigate the mismatch between an initial model and the adaptation data is called *Maximum Likelihood Linear Regression* (MLLR). Specifically, MLLR computes a set of linear regression-based transformations for the mean and variance parameters of an HMM system. The transformation matrices are computed to maximize the likelihood of the adaptation data and can be implemented using the forward-backward algorithm. By applying the transformations to shift the component means and modify the variance in the initial model, each state in the HMM system more easily generates the adaptation data. A notable advantage of this method is that arbitrary adaptation data can be used [176].

Cepstral Mean Normalization (CMN) is a simple method to make speech recognition systems more robust to acoustic environment changes. This is performed by subtracting the mean value of the cepstrum calculated across the whole utterance. The method does not require prior knowledge of the new environment, so it adapts quickly to changing environment.

One major research direction focuses on leveraging neural networks to compensate for the effects of the mismatch problems in an automatic way. To enable adaptation using small amounts of unlabeled speech data from a new speaker, a feed-forward network based method is proposed to factor the speech knowledge into speaker-independent models, continuous speaker-specific parameters, and a transformation which alters the models in accordance with the speaker parameters [177]. In [178, 179], auto-associative neural networks are used in a way of capturing the speaker-specific features so that the effects of the mismatch problem are optimally minimized. Due to the recent popularity of DNNs for acoustic modeling, a large number of speaker adaptation techniques placed focus on DNN acoustic models [180, 181]. An intuitive way is to modify part or all of DNN weights using the available adaptation data based on the standard BP training procedure [182]. To enable very fast adaptation using only a very limited amount of adaptation data, a small size of speaker-dependent code is designed as a compact description of each speaker [181], which is estimated by optimizing the overall composite network performance. As a result, speech features are transformed into a speaker-independent space based on the speaker code in order to normalize speaker variations.

Domain adaptation has also been increasingly studied for cross-corpus experiments [12, 13, 183]. For acoustic emotion recognition, it is empirically reported

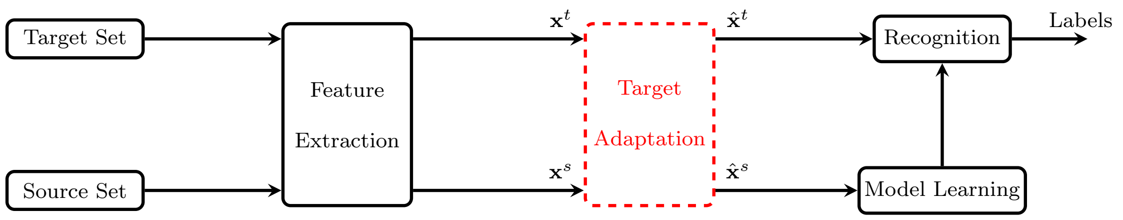


Figure 3.2: Overview of a basic recognition system integrating target adaptation. Target adaptation is achieved by feature transfer learning in this thesis.

that performance decreases greatly when directly operating cross-corpora-wise [12]. In [183], adding unlabeled emotional speech to agglomerated multi-corpus training sets can improve recognition performance across emotion models, emotion elicitation methods, and acoustic conditions. Recently, domain adaptation has even been introduced to explore the similarities between the acoustic structure of music and speech signals in the communication of emotion. Inspired by denoising autoencoder based transfer learning, Coutinho et al. [184] demonstrated that domain adaptation using LSTM networks is suitable for cross-modal time-continuous predictions of emotion in the acoustic domain.

3.3 Feature Transfer Learning

Feature transfer learning aims to distill a common representation across the source domain and the target domain, which can make the two domains appear to have similar distributions, leading to positive transfer. For feature transfer learning to occur, a target adaptation module needs to be added to the top of the feature extraction in a basic recognition system. Such an adaptation module aims to adapt a classifier trained on the source data for use in the target data (see Figure 3.2). In order to alleviate the distribution mismatch (see Section 3.1), normal feature transfer leaning methods either alter the original feature space or map the original feature space into a new space which is predictive across both domains.

Probably the simplest approach of feature transfer learning might be the feature augmentation method introduced by Daumé III [185]. In this approach, feature transfer learning is achieved by taking each feature in the original problem and duplicating each feature to three versions: a general version, a source-specific version, and a target-specific version. Therefore, the augmented source data contain only general and source-specific versions, while the augmented target data consist of general and target-specific versions. Afterwards, the augmented data are fed into common supervised learning algorithms. The most appealing advantage of this approach is that it is extremely easy to implement.

Besides, more sophisticated feature transfer approaches have been proposed for the last decade. Blitzer et al. [186] introduced the concept of *pivot features* from structural correspondence learning to identify correspondences among features from different domains. Specifically, pivot features are features which behave in the same way for discriminative learning in both domains. Non-pivot features from different domains, which are correlated with many of the same pivot features, are assumed to correspond. In the context of multitask learning, Argyriou and Evgeniou [187] presented a sparse feature method to learn a low-dimensional representation shared across multiple related tasks.

Feature transfer learning has really emerged as the major research direction in transfer learning in recent years due to the increasing growth of feature learning. *Feature learning* (or *representation learning*) tries to automatically learn transformations of the data that make it easier to extract useful information when building classifiers or other predictors [146]. Feature learning approaches most commonly used for transfer include *sparse coding*, *Principal Component Analysis* (PCA), and *autoencoders*. The goal of these approaches is to learn either a low-dimensional latent feature space or a shared feature space. The resulting feature space can serve as a bridge of transferring meaningful knowledge from the source domain to the target domain. PCA is a simple, non-parametric method, which is typically used to project the data along the direction of maximal variance. But combining PCA and a Bregman divergence-based regularization became an effective transfer learning method which can result in a subspace wherein the distribution difference between two domains is minimized [188]. Sparse coding was originally proposed as an unsupervised feature learning model [189]. Based on sparse coding, Raina et al. [190] presented a self-taught learning framework to exploit the unlabeled data. Besides, the use of sparse coding and dictionary learning also works well in the context of transfer leaning and multitask leaning [191]. The core idea is that the task parameters can be well approximated by sparse linear combinations of the atoms of a dictionary on a high or infinite dimensionality.

A notable advantage of feature learning is that it can produce *distributed* (or *sparse*) and *abstract* features [146]. Distributed features are expressive, which means that a reasonably-size learned representation can capture a huge number of possible input configurations. In other words, they are insensitive to small variations of a given input. More abstract concepts are generally invariant to most local changes of the input. That makes the representations that capture these concepts generally highly nonlinear functions of the raw input. Thus, distributed and abstract features potentially have greater predictive power.

The focus of this thesis is mainly placed on autoencoder based feature transfer learning. Unlike sparse coding which consists in solving a convex optimization problem, the solution of the autoencoder model is determined by optimizing a neural network. Autoencoders embrace the great advantage of feature learning, i.e., producing distributed and abstract features. That is true of deep architectures, where they tend to result in progressively more abstract features at higher layers of representations [146]. As an example, a deep architecture can automatically find patterns in a hierarchy in the large amount of raw image data, including edges and circles, semantic textons, motifs, discriminative parts of the image (e.g., eyes and noses), and objects. For speech, deep architectures can discover multiple spectral bands and phone classes from the raw time speech signal [48, 50]. Most importantly, autoencoder based feature transfer learning algorithms have shown an advantage for transfer learning tasks because, in practice, they won the two transfer learning challenges held in 2011. Other examples of the successful application of such algorithms can be found in sentiment classification [166, 192].

3.4 Feature Transfer Learning based on Autoencoders

3.4.1 Notations

To facilitate discussion, this section first introduces some notations. In this thesis, the superscript notations t and s are used to distinguish the target domain and the source domain. Therefore $\mathcal{D}^t$ represents data in the target domain, $\mathcal{D}^s$ represents data in the source domain. Let us assume a given target training set of n_t examples $\mathbf{X}^t = \{\mathbf{x}_1^t, \ldots, \mathbf{x}_{nt}^t\}$, along with a corresponding label set $\mathbf{t}^t = \{t_1^t, \ldots, t_{n_t}^t\}$ drawn from some distribution $\mathcal{D}^t$, and a source training set of n_s examples $\mathbf{X}^s = \{\mathbf{x}_1^s, \ldots, \mathbf{x}_{n_s}^s\}$, along with a corresponding label set $\mathbf{t}^s = \{t_1^s, \ldots, t_{n_s}^s\}$ drawn from some distribution $\mathcal{D}^s$. Given the dimension of features n and the overall number of classes n_c, the target training set and the source share the same feature space and label space, i.e., each input feature vector $\mathbf{x}_i \in \mathbb{R}^n$ and the corresponding class label $t_i \in \{c_1, \ldots, c_{n_c}\}$. However, we do not assume that the target data $\mathbf{X}^t$ was drawn from the same distribution as the source data $\mathbf{X}^s$, which means the classifiers learned from the source set cannot classify the (target) test data well due to different data distribution. In addition, the size of $\mathbf{X}^t$ is often inadequate to train a good classifier for the test data. Transfer learning aims to help improve the learning of the target predictive function in $\mathbf{X}^t$ using the knowledge in $\mathbf{X}^s$ [16]. Note that, the following sections apply the above definitions to introduce different autoencoder based feature transfer learning algorithms. Throughout the thesis, we assume θ refers to the tunable parameters in the proposed autoencoder models.

3.4.2 Autoencoders

The most common example of feature learning approaches is the autoencoder network. An autoencoder, also known as a single-hidden layer feedforward neural network, is an unsupervised learning method which sets the target values to be equal to the input and then learns new representations from the data in a nonlinear parametric closed form [193, 194, 195]. It is typically composed of an *encoder* that takes the input data and computes a different representation, and a *decoder* that takes the new representation given by the encoder and generates a reconstruction of the original input. When learning proceeds to minimize recognition error based on the *Backpropagation* (BP) procedure, autoencoders are not only expected to preserve as much information as possible, but are also expected to make the new representation have desired properties.

There are variants of autoencoders which are often used as a key element in deep learning to find common data representation from the input [126, 196]. For example, a successful autoencoder variant is the *denoising autoencoder* put forward by Vincent et al. [197], based on the motivation of robustness to small input perturbations. The denoising autoencoder simply injects noise in the input and then sends the corrupted form through the autoencoder. Further, it is trained to reconstruct the clean and complete input (i.e., to denoise). In doing so, it has to capture the essential structure of the data distribution. In emotion recognition applications, the denoising autoencoder is found very suitable to model gender information in speech emotional features [70]. In addition to the denoising autoencoder, if a sparsity constraint is imposed on the representation, such autoencoders are called *sparse autoencoders*. The aim is to have a majority of the elements of the representation close to zero. Besides, *Contractive autoencoders* encourage the intermediate representation to be robust to small changes of the input around the training examples by using a well chosen penalty term. This penalty term corresponds to the Frobenius norm of the Jacobian matrix of the encoder activations with respect to the input [198]. The contractive autoencoder is strongly related to the denoising autoencoder. It is shown that, in the limit of small Gaussian corrupted input noise, the denoising reconstruction error amounts to a contractive penalty on the reconstruction [199].

The aforementioned autoencoders directly ignore the 2D image structure. This is not only a problem when dealing with realistically sized inputs, but also brings redundancy to the parameters, forcing each feature to be global (i.e., to span the entire visual field). In fact, the trend in vision and object recognition often used by the most successful models such as CNNs is to discover localized features that repeat themselves all over the input. *Convolutional autoencoders* are different from common autoencoders because they include a pooling layer, such as max-pooling, and make use of shared weights among all locations in the input. The reconstruction is hence due to a linear combination of basic image patches based on the internal representation [200]. One major advantage of a convolutional autoencoder is that the resulting representations given by the convolutional autoencoder are likely to tolerate translation of the input image.

An autoencoder, like a basic neural network (Section 2.2.2), consists of an *input layer*, a *hidden layer*, and an *output layer*, illustrated in Figure 3.3. Formally, in response to an input example $\mathbf{x} \in \mathbb{R}^n$, the *hidden representation* $\mathbf{h} \in \mathbb{R}^m$, or *code* is

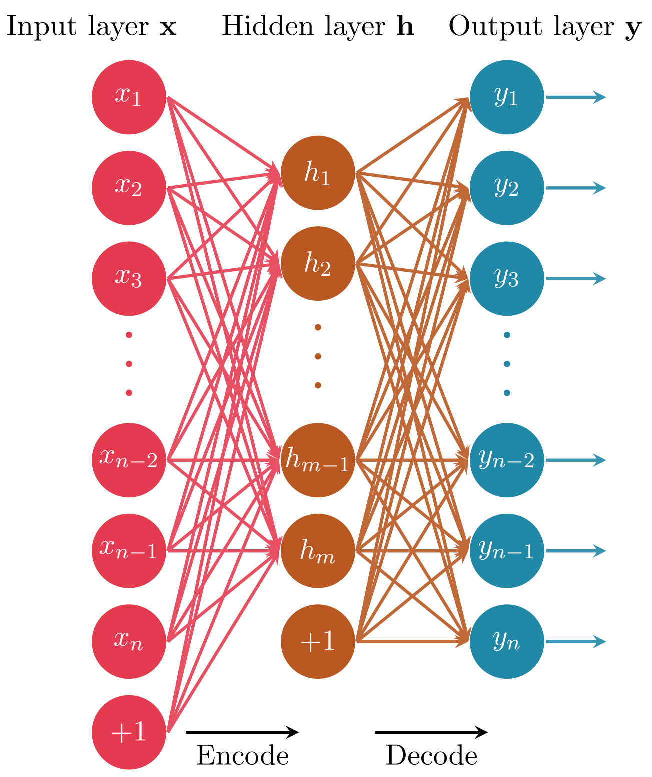


Figure 3.3: An autoencoder architecture. An autoencoder, a special feed-forward neural network, consists of an *input layer*, a *hidden layer*, and an *output layer*.

$$\mathbf{z}^{(1)} = \mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}, \tag{3.16}$$

$$\mathbf{h} = f(\mathbf{z}^{(1)}), \tag{3.17}$$

where $f(\cdot)$ is specified as an activation function (typically a logistic sigmoid function or hyperbolic tangent non-linearity function applied component-wise), $\mathbf{W}^{(1)} \in \mathbb{R}^{m \times n}$ is a weight matrix, and $\mathbf{b}^{(1)} \in \mathbb{R}^m$ is a bias vector. This process that nonlinearly transforms an input into a new representation is known as the *encoder*. After the autoencoder training, the encoder usually produces the representation more robust than the original input, which can be applied to the subsequent process.

The *decoder* maps the hidden representation $\mathbf{h}$ back to a reconstruction $\mathbf{y} \in \mathbb{R}^n$

$$\mathbf{z}^{(2)} = \mathbf{W}^{(2)}\mathbf{h} + \mathbf{b}^{(2)}, \tag{3.18}$$

$$\mathbf{y} = f(\mathbf{z}^{(2)}), \tag{3.19}$$

where $\mathbf{W}^{(2)} \in \mathbb{R}^{n \times m}$ is a weight matrix, and $\mathbf{b}^{(2)} \in \mathbb{R}^n$ is a bias vector. If the two weight matrices are constrained to be of the form $\mathbf{W}^{(2)} = (\mathbf{W}^{(1)})^T$, this is known as *tied weights* which appears to reduce the number of adaptive parameters.

Algorithm 3.2 *Backpropagation* (BP) for autoencoders using matrix-vectorial notation

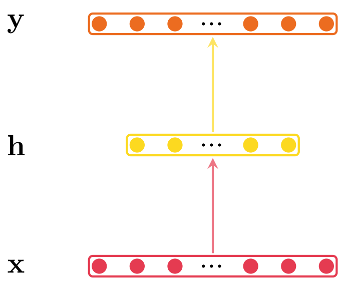
```
1: Perform a forward pass, computing the activations of the hidden and output
   units, using Equations (3.16–3.19).
2: For the output layer, evaluate
       $$\delta^{(2)} = -2(\mathbf{x} - \mathbf{y}) \circ f'(\mathbf{z}^{(2)}). \quad (3.21)$$
3: For the hidden layer, compute
       $$\delta^{(1)} = \left( \left(\mathbf{W}^{(2)}\right)^T \delta^{(2)} \right) \circ f'(\mathbf{z}^{(1)}). \quad (3.22)$$
4: Evaluate the gradients of $\mathcal{J}$ w.r.t. $\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \mathbf{W}^{(2)}$, and $\mathbf{b}^{(2)}$.
       $$\begin{aligned} \frac{\partial \mathcal{J}}{\partial \mathbf{W}^{(1)}} &= \delta^{(1)} \mathbf{x}^T, & \frac{\partial \mathcal{J}}{\partial \mathbf{b}^{(1)}} &= \delta^{(1)}, \\ \frac{\partial \mathcal{J}}{\partial \mathbf{W}^{(2)}} &= \delta^{(2)} (\mathbf{h}^{(1)})^T, & \frac{\partial \mathcal{J}}{\partial \mathbf{b}^{(2)}} &= \delta^{(2)}. \end{aligned} \quad (3.23)$$
```

Given a set of input examples $\mathbf{X}$, the autoencoder (AE) training consists in finding a set of parameters $\theta = \left\{ \mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)} \right\}$ by minimizing the reconstruction error. Many different objective forms can be used as a measure of the reconstruction, in dependence on the distributional assumptions on the input given the representation. If the input is interpreted as either binary vectors or discrete vectors, the *cross-entropy* objective function can be used. For the real-valued input, the SSE (see Section 2.2.2) can be used

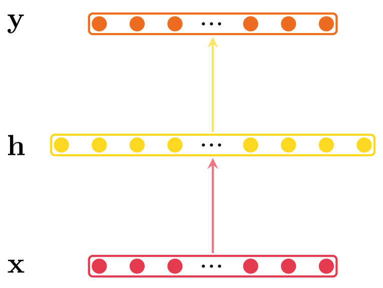
$$\mathcal{J}^{AE}(\theta) = \sum_{\mathbf{x} \in \mathbf{X}} \left\| \mathbf{x} - \mathbf{y} \right\|^2. \tag{3.20}$$

The minimization is usually realized either by BP with SGD or more advanced optimization techniques such as the L-BFGS or conjugate gradient algorithms.

As discussed in Section 2.2.2.2, standard neural networks often apply the BP algorithm to compute the gradient of the objective function with respect to the parameters. Here, like the standard BP algorithm (see Algorithm 2.1), the BP algorithm for basic autoencoders with the objective function in Equation (3.20) is presented in Algorithm 3.2. In the following sections, a number of novel autoencoders will be given based on the basic autoencoder, so that feature transfer learning would profit from the internal representations learned by these autoencoders. For the purpose of helping the reader implement these transfer learning algorithms, the process of computing the gradient information can be achieved by accordingly modifying Algorithm 3.2.



(a) Under-complete Autoencoders



(b) Over-complete Autoencoders

Figure 3.4: Two normal cases of autoencoders, i.e., under-complete autoencoders (number of input units is greater than number of hidden units) and over-complete autoencoders (number of input units is less than number of hidden units).

The topology structure of the autoencoder completely relies on the size of the input layer n and the number of hidden units m. Based on the relation between them, there are two normal cases of autoencoders, namely *under-complete autoencoders* and *over-complete* autoencoders, illustrated in Figure 3.4. The under-complete autoencoder corresponds to an autoencoder in which the number of the input units is more than that of the hidden units, i.e., $n > m$. The under-complete autoencoder attracted major attention of the early work because it can easily avoid learning useless representations. It is agreed that an under-complete autoencoder is sometimes equivalent to PCA if linear activation functions or only a sigmoid activation function are used. This implies that such an autoencoder can only capture a set of directions of variation that are the same everywhere in space [146, 194]. By contrast, the over-complete autoencoder, with an explicit constraint that the input dimension is less than the hidden dimension $n < m$, is also applicable and becoming more interesting. Recent work showed that the over-complete framework allows such autoencoders to capture the structure of the input distribution. Hence, the hidden layer size of an autoencoder is very crucial to controlling both the dimensionality reduction and the capacity.

3.4.3 Sparse Autoencoder

Speech is produced by modulating a relatively small number of parameters of a dynamical system [201, 202], and this implies that its true underlying structure is much lower-dimensional than is immediately apparent in a window that contains hundreds of coefficients [128]. It is believed, therefore, that speech emotional features also have such underlying structure if there is a method that can effectively exploit information embedded in a large data set. To allow for feature transfer learning, one can use the underlying feature structure learned from target data to reconstruct other source data accordingly and preserve the data's own information as much as possible. The *Sparse Autoencoder* (SAE) is used to exploit the underlying feature structure on target data, represented by a set of weight matrices and a bias vector. Given source data are fed to the learned autoencoder structure to reconstruct its own. This section describes briefly the SAE, and then presents in detail the *sparse autoencoder feature transfer learning.*

Sparsity is of strong interest in computational neuroscience and machine learning. Olshausen and Field [189] first presented in computational neuroscience that sparsity can be induced by sparse coding which can result in the sparse representation of natural images sufficient to account for the principal spatial properties of simple-cell receptive fields. In pursuit of sparsity, a learning algorithm usually attempts to convert its input into a new sparse representation, whose elements are mostly either close to zero or equal to zero. Sparsity has been widely used in various autoencoders to produce a sparse distributed representation [127, 165]. It has also been a key element of modern neural networks. For example, the L^1 penalty term, leading to a solution with sparse parameters, has been found useful to prevent the overfitting issue in neural networks. Plus, the widespread acceptance of the ReLU activation function (see Section 2.2.2.1) is due to its ability to easily produce sparse representation [106].

Sparsity has a lot of notable advantages. An advantage of sparsity is that a sparse representation may facilitate information disentangling in deep learning algorithms. A dense representation is highly sensitive to any change in the data. In contrast, a sparse representation is more robust because small input changes produce almost negligible effects on the set of non-zero features. Another advantage is that a spare representation is more prone to be decoded by a linear model at a very low computational cost. Also sparsity plays a key role in learning Gabor-like filters [203]. A sparse variant of deep belief networks is proposed to faithfully mimic certain properties of the visual area V2 in the cortical hierarchy. The first layer of the network results in localized, oriented, edges filters. Further, the network can effectively discover high-level features in the image data. Nevertheless, it is worth noting that, bringing too much sparsity to a model may adversely affect the generalization performance since it limits the capacity of the model.

An SAE is a simple autoencoder on whose objective function is often imposed a sparsity penalty in addition to the reconstruction squared error. A sparsity penalty term acts as a regularizer or as a log-prior on the representations $\mathbf{h}$. For example, it is common to make use of the Laplace prior or the Student-t prior [189] to construct the sparsity penalty.

Another common form of a sparsity penalty for SAEs is to exploit some distribution similarity measure so as to lead the representation towards some low target value. Here, a sparsity penalty, which was introduced in [203], is given in detail. The idea of such a penalty is to constrain the expected activation of the hidden units to be sparse. To this end, a regularizer is added so that it penalizes a deviation of the expected activation of the hidden units from a (low) fixed level ρ such as $\rho = 0.05$. Similar to normal autoencoders introduced in Section 3.4.2, thus, an SAE tries to

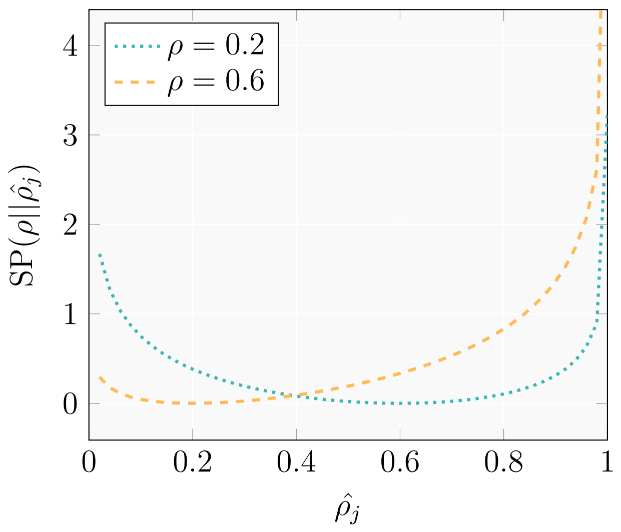


Figure 3.5: Sparsity penalty function $\mathrm{SP}(\rho||\hat{\rho}_j)$ with respect to $\hat{\rho}_j$ given $\rho = 0.2$ or $\rho = 0.6$, which is successfully applied to a sparse autoencoder.

solve the following optimization problem

$$\mathcal{J}^{SAE}(\theta) = \sum_{\mathbf{x}\in\mathbf{X}} \left\|\mathbf{x} - \mathbf{y}\right\|^2 + \beta \sum_{j=1}^{m} \mathrm{SP}(\rho||\hat{\rho}_j), \tag{3.24}$$

where

$$\mathrm{SP}(\rho||\hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_j} \tag{3.25}$$

is a sparse penalty term, $\hat{\rho}_j = \frac{1}{N}\sum_{i=1}^{N} h_j(\mathbf{x}_i)$ is the average activation of hidden unit j (averaged over the training set), m is the number of hidden units, $\rho \in [0,\, 1)$ is a sparsity level, and β controls the weight of the sparsity penalty term.

In the above objective function, in fact, the penalty $\mathrm{SP}(\rho||\hat{\rho}_j)$ is just the KL divergence between the Bernoulli random variable with mean ρ and the Bernoulli random variable with mean $\hat{\rho}_j$. The KL divergence is a measure of the difference between two different distributions [204]. The KL divergence satisfies $\mathrm{KL}(p||q) \geq 0$ and has the crucial property that $\mathrm{KL}(p||q) = 0$ if, and only if $q = p$, and otherwise it rises monotonically as q diverges from p. Analogous to the KL divergence, the penalty function $\mathrm{SP}(\rho||\hat{\rho}_j)$ reaches its minimum of 0 at $\hat{\rho}_j = \rho$, and dramatically grows as $\hat{\rho}_j$ comes close to 0 or 1 (cf. Figure 3.5). By that, $\hat{\rho}_j$ is expected to approximate ρ in the learning phase. To induce sparsity in the representation, hence, ρ is often set to a small value, for example, $\rho = 0.01$ in this thesis.

Despite SAEs involve the sparse penalty term, it is still easy to find a solution to the objective function through the BP algorithm. The learning of SAEs proceeds in a similar way as common autoencoders, which was presented in Algorithm 3.2.

Algorithm 3.3 *Sparse Autoencoder* (SAE) Feature Transfer Learning

```
Input: Two labeled data sets $\mathbf{X}^t$ and $\mathbf{X}^s$, and the corresponding class set
    $\{c_1, \ldots, c_{n_c}\}$, $n_c$ is the overall number of classes.
Output: Learned classifier $\mathcal{C}$ for the target task.
 1: Initialize reconstructed data $\tilde{\mathbf{X}}^s = \emptyset$.
 2: for $k = 1$ to $n_c$ do
 3:     Initialize an autoencoder $\mathrm{SAE}^k(\theta)$.
 4:     Choose class-specific examples $\mathbf{X}^t_{c_k}$ from $\mathbf{X}^t$.
 5:     Train $\mathrm{SAE}^k(\theta)$ using $\mathbf{X}^t_{c_k}$.
 6:     Choose class-specific examples $\mathbf{X}^s_{c_k}$ from $\mathbf{X}^s$.
 7:     Reconstruct the source $\mathbf{X}^s_{c_k}$: $\tilde{\mathbf{X}}^s_{c_k} = \mathrm{SAE}^k_{\text{Recon}}(\mathbf{X}^s_{c_k})$ (cf. Equation (3.27)).
 8:     Update the reconstructed data $\tilde{\mathbf{X}}^s = \tilde{\mathbf{X}}^s \cup \tilde{\mathbf{X}}^s_{c_k}$.
 9: end for
10: Learn a classifier $\mathcal{C}$ by applying supervised learning algorithm $s$ (e.g., SVMs) to
    the reconstructed data $\tilde{\mathbf{X}}^s$.
11: return The learned classifier $\mathcal{C}$.
```

However, it is still necessary to replace Equation (3.22) with the following term

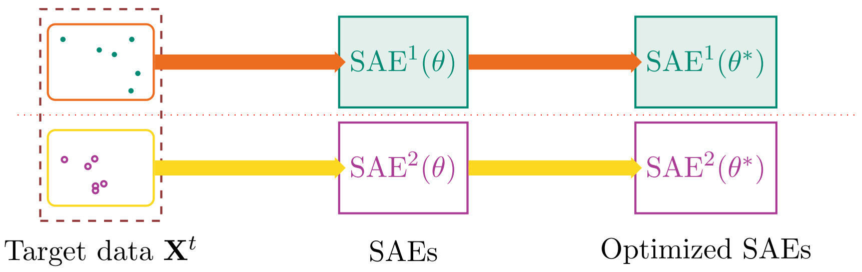
$$\delta^{(1)} = \left(\left(\mathbf{W}^{(1)}\right)^T \delta^{(2)} + \beta \mathrm{SP}(\rho || \hat{\rho}) \mathbf{1} \right) \circ f'(\mathbf{z}^{(1)}), \tag{3.26}$$

where $\mathrm{SP}(\rho||\hat{\rho}) \in \mathbb{R}^m$, $\mathbf{1}$ is a row vector, whose size depends on the number of training examples N.

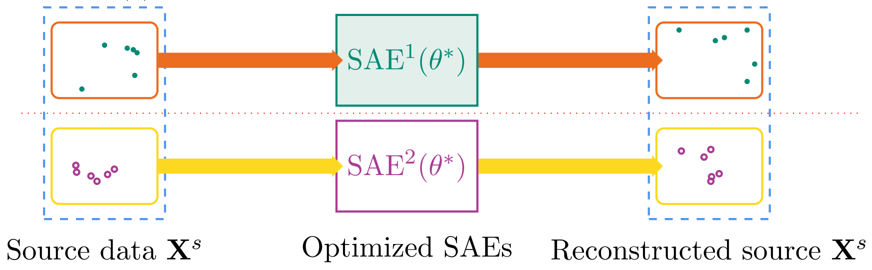
3.4.3.1 Sparse Autoencoder Feature Transfer Learning

Since speech can be segmented into units of analysis, such as phonemes, previous work tends to learn a sparse representation in speech related tasks via stacked autoencoders. For example, Dahl et al. [205] proposed a context-dependent model for large vocabulary speech recognition that uses deep belief networks for phone recognition. This is not applicable in emotion recognition from speech since common units of analysis can be hardly found. However, emotional features are highly correlated in terms of a specific emotion, thus examples with the same emotional state can be assumed to share implicitly a common structure. The autoencoder has shown the capability of discovering a common structure in the data. Motivated by these, a sparse autoencoder-based feature transfer learning method is presented for semi-supervised transfer learning.

For class label c_k in the target training data $\mathbf{X}^t$, first apply an SAE to class-specific examples $\mathbf{X}^t_{c_k} \subset \mathbf{X}^t$ to learn a set of parameters $\mathbf{W}_1$, $\mathbf{W}_2$, $\mathbf{b}_1$, and $\mathbf{b}_2$. To transfer each of the class-specific examples $\mathbf{X}^s_{c_k}$ from the source data $\mathbf{X}^s$, i.e., $\mathbf{X}^s_{c_k} \subset \mathbf{X}^s$, to the target domain, then compute features $\tilde{\mathbf{X}}^s_{c_k}$ based on the learned set of the



(a) Train SAEs on specific class examples in target data


SAE$^1(\theta^*)$
SAE$^2(\theta^*)$
Source data $\mathbf{X}^s$
Optimized SAEs
Reconstructed source $\mathbf{X}^s$

(b) Reconstruct source data by corresponding SAE

Figure 3.6: Flowchart of sparse autoencoder (SAE) feature transfer learning for a two-class problem. Examples with different labels are indicated by the dots and circles.

parameters by the forward pass

$$\tilde{\mathbf{X}}^s_{c_k} = \text{SAE}_{\text{Recon}}(\mathbf{X}^s_{c_k}), \tag{3.27}$$

where $\text{SAE}_{\text{Recon}}(\mathbf{x}) = f(\mathbf{W}^{(2)} f(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}) + \mathbf{b}^{(2)})$ is the output of the autoencoder. The aim of Equation (3.27) is to force the input source $\mathbf{X}^s_{c_k}$ to reconstruct itself through computing a sparse nonlinear combination of the parameters learned on the target data. The reconstructing procedure, in turn, decreases the difference between the source data and the target data, as well as completes the feature transfer from the source domain to the target domain.

A formal description of the framework is given in Algorithm 3.3 and its flowchart is demonstrated in Figure 3.6. As can be seen from the algorithm, at each iteration step, examples belonging to class c_k in the target set are used to train an SAE denoted $\text{SAE}^k(\theta)$ which captures a general mapping structure for the input examples. Then, the algorithm moves to transfering information from the source to the target domain. For the source set, examples with the corresponding class are reconstructed by using $\text{SAE}^k_{\text{Recon}}(\mathbf{X}^s_{c_k})$, as described in Equation (3.27), according to the mapping structure learned by the trained autoencoder $\text{SAE}^k(\theta^*)$. Next, like most emotion recognition systems, this algorithm uses these reconstructed features as input to standard supervised classification algorithms $\mathcal{C}$ – here, SVMs. Finally, a test partition is used to evaluate the classifier. Apparently the small amount of the labeled target

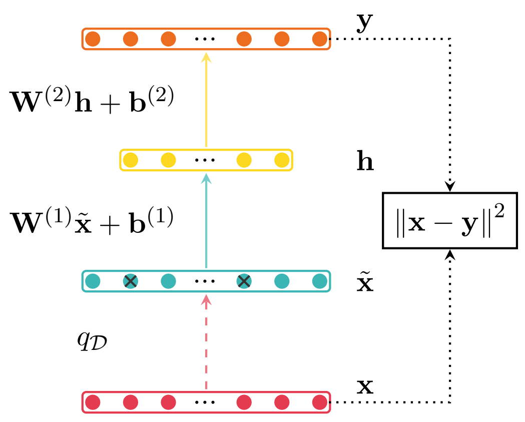

$\mathbf{y}$
$\mathbf{W}^{(2)}\mathbf{h} + \mathbf{b}^{(2)}$
$\mathbf{h}$
$\|\mathbf{x} - \mathbf{y}\|^2$
$\mathbf{W}^{(1)}\tilde{\mathbf{x}} + \mathbf{b}^{(1)}$
$\tilde{\mathbf{x}}$
$q_{\mathcal{D}}$
$\mathbf{x}$

Figure 3.7: A denoising autoencoder (DAE) architecture. An input $\mathbf{x}$ is corrupted (via $q_{\mathcal{D}}$) to $\tilde{\mathbf{x}}$. The black crosses ("$\times$") illustrate a corrupted version of the input $\mathbf{x}$ made by $q_{\mathcal{D}}$.

data play a key role in the transfer method. Hence, the presented transfer method suits for a semi-supervised transfer learning problem.

3.4.4 Denoising Autoencoders

A *Denoising Autoencoder* (DAE) – a more recent variant of the basic autoencoder – is trained to reconstruct a clean 'repaired' input from a corrupted version [197]. In doing so, the learner must capture the underlying structure of the input distribution in order to reduce the effect of the corruption process [146]. It turns out that in this way more robust features are learned compared to a basic autoencoder. Due to this useful characteristic, the DAE has been broadly adopted to efficiently help provide better representation in SER [70, 206, 207]. Furthermore, a DAE with bidirectional LSTM recurrent neural networks has been successfully applied to acoustic novelty detection, aiming at identifying abnormal/novel acoustic signals [208]. The architecture of a DAE is given in Figure 3.7.

A DAE closely resembles an autoencoder introduced in Section 3.4.2. In a DAE, however, there is a particular process, adding artificially noise to the input. That is, an input example $\mathbf{x} \in \mathbb{R}^n$ is converted to a corrupted version $\tilde{\mathbf{x}}$ by means of a corrupting function $\tilde{\mathbf{x}} \sim q_{\mathcal{D}}(\tilde{\mathbf{x}}|\mathbf{x})$, which could be either a binary masking noise (deleting random elements of the input), or additive isotropic Gaussian noise, or salt-and-pepper noise in images.

In particular, a binary masking noise randomly chooses a part of the input elements and has their value set to 0. Mathematically, a corrupting function using a binary masking noise can be written as

$$\tilde{\mathbf{x}} = \mathrm{Bin}(n, 1 - p_n) \circ \mathbf{x} \tag{3.28}$$

Algorithm 3.4 *Denoising Autoencoder* (DAE) Feature Transfer Learning

```
Input: Unlabeled target data set $\mathbf{X}^t$, labeled source set $\mathbf{X}^s$.
Output: Learned classifier $\mathcal{C}$ for the target task.
1: Initialize an autoencoder DAE($\theta$), $\theta = \{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)}\}$ .
2: Train DAE($\theta$) using $\mathbf{X}^t$.
3: Obtain the encoder: Encoder($\mathbf{x}$) $= f(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)})$ (cf. Equation (3.17)).
4: Generate the target representation $\mathbf{H}^t$: $\mathbf{H}^t$ = Encoder($\mathbf{X}^t$).
5: Generate the source representation $\mathbf{H}^s$: $\mathbf{H}^s$ = Encoder($\mathbf{X}^s$).
6: Learn a classifier $\mathcal{C}$ by applying supervised learning algorithm $s$ (e.g., SVMs) to
   the source representations $\mathbf{H}^s$.
7: return The learned classifier $\mathcal{C}$.
```

where Bin($\cdot$) denotes the binomial distribution function, n denotes the dimension of $\mathbf{x}$, and the symbol "$\circ$" denotes the Hadamard product, also known as the element-wise product, $p_n \in [0, 1)$ is the given corruption level.

After the corrupting process, the corrupted version $\tilde{\mathbf{x}}$ of the input runs through the encoder and decoder. Then, the objective function is used to measure the difference between the reconstruction and the clean input. It turns out that such an autoencoder has the ability to meaningfully capture the structure of the input data.

Typically one can simply perform BP to compute gradients, similar to regular autoencoders (Algorithm 3.2). The only difference is the corruption of the input. More details on DAEs can be found in [197].

3.4.4.1 Denoising Autoencoder Feature Transfer Learning

DAEs have been successfully applied to feature transfer learning. For example, Glorot et al. [166] applied a stacked DAE with sparse rectifiers to domain adaption in large-scale sentiment analysis. Another successful application of DAE feature transfer learning was applied in the field of image processing [209].

Here, *denoising autoencoder feature transfer learning* is given as follows. The pseudocode is described in Algorithm 3.4. For the purpose of discovering the knowledge from the target domain, the unlabeled target data $\mathbf{X}^t$ are fed into the training procedure of a DAE. Then, both the target data $\mathbf{X}^t$ and the source data $\mathbf{X}^s$ are transformed to their new representations ($\mathbf{H}^t$ and $\mathbf{H}^s$) according to the feature encoding function (Equation (3.17)). In this way, the difference between the target data and the source data in the new space only learned on the target data is hopefully decreased. Afterwards, these representations are taken to build a standard supervised classifier.

During yielding the feature transformation, however, this method apparently ignores an attempt to explore the information behind the source data, and forces the source data to generate their new representation under the characteristics as given by the target data. In this case, one may unexpectedly lose those examples of the source data that are not following these characteristics, such that one may lose information useful for the subsequent supervised classifier to a certain degree. Even worse, negative transfer learning may arise, which may lead the learner to performing worse than no transferring at all [16]. Nevertheless, the denoising autoencoder transfer method is appealing since it is a simple but efficient transfer technique.

3.4.5 Shared-hidden-layer Autoencoders

The idea behind transfer learning is to exploit commonalities between different learning tasks in order to share statistical strength, and transfer knowledge across tasks [15, 146, 210]. As an example, for a low-level visual feature space together with attribute and object-labeled image data, a convex multitask feature learning approach was introduced to learn a shared lower-dimensional representation by optimizing a joint loss function that favors common sparsity patterns across both types of prediction tasks [211]. This idea also seems to be very helpful for multimodal learning, which involves discovering relationships across multiple sources. For example, a shared representation learning structure based on deep autoencoders was found to be a very efficient way of modeling correlations across speech and visual signals [212]. Similar to this work, a recent approach proposes a multimodal deep Boltzmann machine model for learning joint representations in multimodal data [213]. The key idea is to learn a joint density model over image and text inputs.

Based on the motivation of the 'sharing idea' in transfer learning, this section proposes an alternative structure of autoencoder that attempts to minimize the reconstruction error on both source set and target set [206]. The *Shared-hidden-layer Autoencoder* (SHLA) shares the same parameters for the mapping from the input layer to the hidden layer, but uses independent parameters for the reconstruction process. This makes it much easier to discover the nonlinear commonalities across different sets. The structure of the SHLA is shown in Figure 3.8.

Following the notations used for the introduction of autoencoders in Section 3.4.2, this section presents the formulations of the SHLA method as follows. Given a source set of examples $\mathbf{X}^s$, and a target set of examples $\mathbf{X}^t$, the two objective functions, similar to Equation (3.20), are formed as follows

$$\mathcal{J}^s(\theta^s) = \sum_{\mathbf{x}\in\mathbf{X}^s} \|\mathbf{x}-\mathbf{y}\|^2, \tag{3.29}$$

$$\mathcal{J}^t(\theta^t) = \sum_{\mathbf{x}\in\mathbf{X}^t} \|\mathbf{x}-\mathbf{y}\|^2, \tag{3.30}$$

where the parameters $\theta^s = \left\{\mathbf{W}^{(1)}, \mathbf{W}^{s(2)}, \mathbf{b}^{(1)}, \mathbf{b}^{s(2)}\right\}$, and $\theta^t = \left\{\mathbf{W}^{(1)}, \mathbf{W}^{t(2)}, \mathbf{b}^{(1)}, \mathbf{b}^{t(2)}\right\}$ share the same parameters $\{\mathbf{W}^{(1)}, \mathbf{b}^{(1)}\}$.

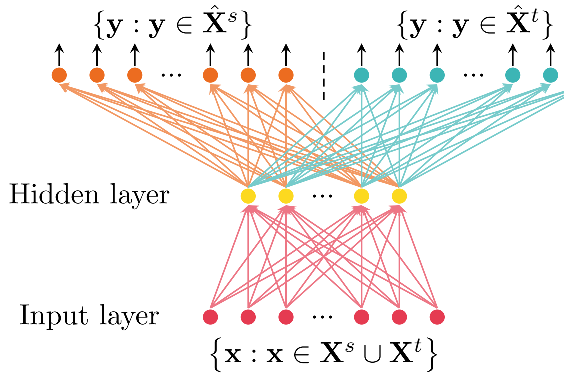

$\{\mathbf{y} : \mathbf{y} \in \hat{\mathbf{X}}^s\}$
$\{\mathbf{y} : \mathbf{y} \in \hat{\mathbf{X}}^t\}$
Hidden layer
Input layer
$\{\mathbf{x} : \mathbf{x} \in \mathbf{X}^s \cup \mathbf{X}^t\}$

Figure 3.8: Structure of the *Shared-hidden-layer Autoencoder* (SHLA) on the source set $\mathbf{X}^s$ and target set $\mathbf{X}^t$. The SHLA shares the same parameters for the mapping from the input layer to the hidden layer, but uses independent parameters for the corresponding reconstructions $\hat{\mathbf{X}}^s$ and $\hat{\mathbf{X}}^t$.

Besides, optimizing the joined distance for the two sets leads to the following overall objective function

$$\mathcal{J}^{SHLA}(\theta) = \mathcal{J}^s(\theta^s) + \gamma\mathcal{J}^t(\theta^t), \tag{3.31}$$

where $\theta = \left\{\mathbf{W}^{(1)}, \mathbf{W}^{s(2)}, \mathbf{W}^{t(2)}, \mathbf{b}^{(1)}, \mathbf{b}^{s(2)}, \mathbf{b}^{t(2)}\right\}$ are the parameters to be optimized during training, and the hyper-parameter γ controls the strength of the regularization.

Training the SHLA is equivalent to training a basic autoencoder, and the standard BP algorithm can be applied. Nevertheless, it is necessary to make the following modifications to the original BP for the SHLA.

$$\frac{\partial\mathcal{J}^t}{\partial\mathbf{W}^{(1)}} = \delta^{t(1)}(\mathbf{x}^t)^T, \tag{3.32}$$

$$\frac{\partial\mathcal{J}^s}{\partial\mathbf{W}^{(1)}} = \delta^{s(1)}(\mathbf{x}^s)^T, \tag{3.33}$$

$$\frac{\partial\mathcal{J}^{SHLA}}{\partial\mathbf{W}^{(1)}} = \frac{\partial\mathcal{J}^s}{\partial\mathbf{W}^{(1)}} + \gamma\frac{\partial\mathcal{J}^t}{\partial\mathbf{W}^{(1)}}, \tag{3.34}$$

$$\frac{\partial\mathcal{J}^{SHLA}}{\partial\mathbf{b}^{(1)}} = \delta^{t(1)} + \gamma\delta^{s(1)}. \tag{3.35}$$

By explicitly adding the regularization term from the target set, the SHLA is equipped with extensive flexibilities to directly incorporate the knowledge of the target set. Hence, during minimizing the objective function, the shared hidden layer is biased to make the distribution induced by the source set as similar as possible to the distribution induced by the target set. This helps to regularize the functional behavior of the autoencoder. It further turns out to lessen the effects of the difference in the source and target sets.

Algorithm 3.5 *Shared-hidden-layer Autoencoder* (SHLA) Feature Transfer Learning

```
Input: Unlabeled target data set $\mathbf{X}^t$, the labeled source set $\mathbf{X}^s$.
Output: Learned classifier $\mathcal{C}$ for the target task.
1: Initialize an autoencoder SHLA($\theta$), $\theta = \{\mathbf{W}^{(1)}, \mathbf{W}^{s(2)}, \mathbf{W}^{t(2)}, \mathbf{b}^{(1)}, \mathbf{b}^{s(2)}, \mathbf{b}^{t(2)}\}$
2: Train SHLA($\theta$) using $\mathbf{X}^t$ and $\mathbf{X}^s$.
3: Obtain the encoder: Encoder($\mathbf{x}$) = $f(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)})$ (cf. Equation (3.17)).
4: Generate the target representation $\mathbf{H}^t$: $\mathbf{H}^t$ = Encoder($\mathbf{X}^t$).
5: Generate the source representation $\mathbf{H}^s$: $\mathbf{H}^s$ = Encoder($\mathbf{X}^s$).
6: Learn a classifier $\mathcal{C}$ by applying supervised learning algorithm $s$ (e.g., SVMs) to
   the source representations $\mathbf{H}^s$.
7: return The learned classifier $\mathcal{C}$.
```

Besides, the SHLA can be regarded as an instance of multitask learning [156]. From the point of view of multitask learning, the low layer of the SHLA can be shared across all domains, while domain-specific parameters of the last layer can be learned on top of a shared representation (associated respectively with $\{\mathbf{W}^{s(2)}, \mathbf{b}^{s(2)}\}$ and $\{\mathbf{W}^{t(2)}, \mathbf{b}^{t(2)}\}$). In this way, improved generalization can be achieved by the SHLA due to the shared parameters.

3.4.5.1 Shared-hidden-layer Autoencoder Feature Transfer Learning

Algorithm 3.5 depicts the pseudocode of the *shared-hidden-layer autoencoder feature transfer learning.* As can been seen from Algorithm 3.5, this method is analogous to the denoising autoencoder feature transfer learning (see Section 3.4.4.1), but it learns the common knowledge of the source data and the target data simultaneously. Generally speaking, the overall algorithm can be divided roughly into three phases – feature learning, classifier training, and testing. In detail, this method first applies the source data and target data in the training of an SHLA in an unsupervised manner. After the training, it results in the feature transformation which would in particular balance the 'conflicts' between the two mismatched data in an optimization way. Subsequently, this method yields the new representations by using the encoder described by the shared parameters ($\mathbf{W}^{(1)}$ and $\mathbf{b}^{(1)}$) and trains a supervised classifier on the new representations of the source data. Finally, the classifier is tested on the target data.

3.4.6 Adaptive Denoising Autoencoders

It is known that a conventional DAE is good at capturing the structure of the input data (Section 3.4.4). For the purpose of transfer learning, however, the DAE model appears to lack the ability to access the knowledge of the source domain, which may cause the difficulty of performing the transfer between the source domain and target domain. Recently, a novel method using a DAE in conjunction with a new variant of DAEs, namely *Adaptive Denoising Autoencoders* (A-DAEs), was used for unsupervised transfer learning. This section starts with introducing the A-DAE model following the notations used for the autoencoder (Section 3.4.2) and then presents the new transfer learning method on the basis of DAEs and A-DAEs.

The A-DAE model is partially inspired by [214, 215], in which a knowledge transfer model incorporates prior knowledge from a pre-trained model. More specifically, SVMs, such as adaptive SVMs [215] and adaptive least-square SVMs [214], are used to learn from the source model $\boldsymbol{w}^s$ by regularizing the distance between the learned model $\boldsymbol{w}$ and $\boldsymbol{w}^s$. To this end, an adaptive regularization term is used to measure the distance between them. Furthermore, such models can be easily extended to a multi-model knowledge transfer model, aiming to exploit the knowledge from multiple source datasets [214]. The similar idea of exploiting prior models has also been considered for one-shot learning [159], metric-learning, and hierarchical classification [215].

Besides, there has been a large body of related work on model based transfer learning in the field of neural networks [216, 217, 218, 219] by extending the model compression idea [220]. The aim is to transfer the knowledge in a complex and large-size DNN (or even a big ensemble of neural networks) to a small-size DNN, which thus mimics the model learned by the large net and achieves similar accuracy. In the industry, these techniques are highly appealing because they can allow devices with limited computational and storage resources, such as smart phones and wearable devices, to run a 'low-cost' neural network as powerful and accurate as a large-size neural network with a very large number of parameters. Unlike the aforementioned methods which perform knowledge transfer depending on the parameters of the models, the neural network methods depend on either the logits (activations before softmax) [216], or the posterior probabilities (softmax outputs) [217], or the posterior probabilities and the intermediate representations [219].

However, previous studies motivated by such an idea seem appealing under one key assumption that a few labeled target examples are available for the learners such as SVMs. In contrast, a successful approach is the A-DAE model, extending this idea to an unsupervised scenario [11]. In the case of the A-DAE model, the learned model corresponds to a well trained DAE. That is, a DAE is first learned in a fully unsupervised way from the target domain adaptation data, resulting in the weight matrices $\mathbf{W}^{t(1)}$ (input to hidden layer) and $\mathbf{W}^{t(2)}$ (hidden to output layer) from Equation (3.20) as well as the bias vectors $\mathbf{b}^{t(1)}$ and $\mathbf{b}^{t(2)}$.

An adaptive DAE next forces its weights to adapt to the provided weights as well as minimize the reconstruction error between the input and the output at the same time. The output bias vectors $\mathbf{b}^t$ of the DAE are not adapted. Hence, given a source example $\mathbf{x} \in \mathbf{X}^s$ and the weights $\mathbf{W}^{t(1)}$ and $\mathbf{W}^{t(2)}$ of a DAE, which were estimated without supervision from the target domain adaptation data (i.e., without knowledge

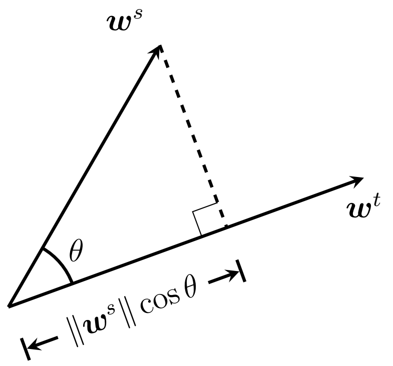


Figure 3.9: Visualization of the projection of the vector $\boldsymbol{w}^s$ onto $\boldsymbol{w}^t$.

of target labels), the objective function of an A-DAE is formulated as follows

$$\begin{aligned} \mathcal{J}^{A-DAE}(\theta) &= \frac{\lambda}{2}\left(\sum_{l=1}^{2}\sum_{j}\left\|\boldsymbol{w}_j^{s(l)} - \beta\boldsymbol{w}_j^{t(l)}\right\|^2\right) \\ &+ \sum_{\mathbf{x}\in\mathbf{X}^s}\|\mathbf{x}-\mathbf{y}\|^2, \end{aligned} \tag{3.36}$$

where $\mathbf{y}$ denotes the reconstruction, and the hyper-parameter β controls the amount of transfer regularization. The weights $\mathbf{W}^{s(1)}$ and $\mathbf{W}^{s(2)}$ are initialized randomly and learned during training, while the weights $\mathbf{W}^{t(1)}$ and $\mathbf{W}^{t(2)}$ are kept constant during training. The parameter β acts as a weighting factor, which scales the importance of the old model. If β is set equal to 0, the adaptive DAE corresponds to the original DAE model without any adaption to previous knowledge. It is worth noting that like a DAE, the A-DAE model has a corrupting process, which artificially injects noise into the input.

Without loss of generality, the intuition of the adaptive DAE for incorporating prior knowledge can be understood by expanding the weight decay regularization term (see Section 2.2.2.3)

$$\begin{aligned} \left\|\boldsymbol{w}^s - \beta\boldsymbol{w}^t\right\|^2 &= \|\boldsymbol{w}^s\|^2 + \beta^2\left\|\boldsymbol{w}^t\right\|^2 \\ &-2\beta\|\boldsymbol{w}^s\|\left\|\boldsymbol{w}^t\right\|\cos\theta, \end{aligned} \tag{3.37}$$

where θ is the angle between the two column vectors $\boldsymbol{w}^s$ and $\boldsymbol{w}^t$ as given in Figure 3.9.

On the one hand, apart from minimizing the original term $\|\boldsymbol{w}^s\|^2$, the optimization problem aims to use the term $-2\beta\|\boldsymbol{w}^s\|\|\boldsymbol{w}^t\|\cos\theta$ to make the transfer by maximizing $\cos\theta$, which is equivalent to minimizing the angle θ between the $\boldsymbol{w}^s$ and $\boldsymbol{w}^t$. Note that, the term $\cos\theta$ is maximized only if θ is 0. On the other hand, the term $\|\mathbf{x}-\mathbf{y}\|^2$ in the objective function also causes $\|\boldsymbol{w}^s\|$ to adjust to the training data and prevents $\boldsymbol{w}^s$ being close to $\boldsymbol{w}^t$. Thus, an adaptive DAE training consists

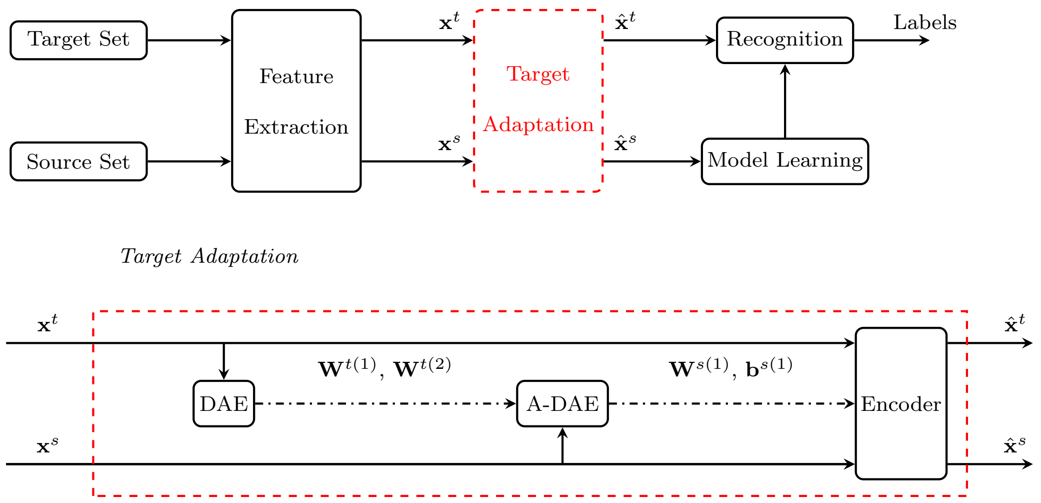


Figure 3.10: Overview of the recognition system integrating the proposed adaptive DAE feature transfer learning method. The function "Encoder" refers to the feed-forward procedure (i.e., Equation (3.17)) from input data to the activations of the hidden layer of a pre-trained DAE.

of optimizing a trade-off between the reconstruction error on the training data and target domain knowledge transfer.

As discussed above, the objective function of the A-DAE includes the adaptive regularization term with respect to the weight matrices, in addition to the reconstruction error. Hence, care must be taken in computing the gradients of these weights. For the BP algorithm of the adaptive DAE model, one can use the following equations in place of the ones in the standard BP algorithm (see Algorithm 3.2).

$$\frac{\partial \mathcal{J}}{\partial \mathbf{W}^{(1)}} = \delta^{(1)}\mathbf{x}^T + \lambda\left(\mathbf{W}^{s(1)} - \mathbf{W}^{t(1)}\right), \tag{3.38}$$

$$\frac{\partial \mathcal{J}}{\partial \mathbf{W}^{(2)}} = \delta^{(2)}(\mathbf{h}^{(2)})^T + \lambda\left(\mathbf{W}^{s(2)} - \mathbf{W}^{t(2)}\right). \tag{3.39}$$

3.4.6.1 Adaptive Denoising Autoencoders Feature Transfer Learning

Adaptive denoising autoencoders feature transfer learning is a novel three-stage data-driven approach for unsupervised domain adaptation. It is based on adaptive DAEs which can learn from a source training set with the guidance of a template learned previously from target domain adaptation data, which yields a common representation across source and target domains. This proposed method for domain adaptation can be divided into three main stages: firstly, unsupervised learning of target prior knowledge with DAEs on target domain adaptation data; secondly, using such prior knowledge to regularize the training of source data with adaptive DAEs;

Algorithm 3.6 *Adaptive Denoising Autoencoder* (A-DAE) Feature Transfer Learning

```
Input: Unlabeled target data set $\mathbf{X}^t$, the labeled source set $\mathbf{X}^s$.
Output: Learned classifier $\mathcal{C}$ for the target task.
1: Initialize a denoising autoencoder DAE($\theta^t$), $\theta^t = \{\mathbf{W}^{t(1)}, \mathbf{W}^{t(2)}, \mathbf{b}^{t(1)}, \mathbf{b}^{t(2)}\}$.
2: Train DAE($\theta$) using $\mathbf{X}^t$.
3: Initialize an A-DAE($\theta^s$), $\theta^s = \{\mathbf{W}^{s(1)}, \mathbf{W}^{s(2)}, \mathbf{b}^{s(1)}, \mathbf{b}^{s(2)}\}$.
4: Train A-DAE($\theta^s$) using $\mathbf{X}^s$ and the learned parameters $\mathbf{W}^{t(1)}$ and $\mathbf{W}^{t(2)}$ .
5: Obtain the encoder: Encoder($\mathbf{x}$) = $f(\mathbf{W}^{s(1)}\mathbf{x} + \mathbf{b}^{s(1)})$ (cf. Equation (3.17)).
6: Generate the target representation $\mathbf{H}^t$: $\mathbf{H}^t$ = Encoder($\mathbf{X}^t$).
7: Generate the source representation $\mathbf{H}^s$: $\mathbf{H}^s$ = Encoder($\mathbf{X}^s$).
8: Learn a classifier $\mathcal{C}$ by applying supervised learning algorithm $s$ (e.g., SVMs) to
   the source representations $\mathbf{H}^s$.
9: return The learned classifier $\mathcal{C}$.
```

and thirdly encoding target data and source data with a feed-forward procedure.

In general, the aim is to capture source domain knowledge in training an adaptive DAE with the guidance of the prior knowledge previously learned from target domain data by a DAE. Algorithm 3.6 presents this proposed method and Figure 3.10 depicts the basic recognition method integrated with the proposed domain adaptation method. The proposed method is composed of the following three stages: First, a DAE is learned in a fully unsupervised way from the target domain adaptation data, resulting in the weight matrices $\mathbf{W}^{t(1)}$ (input to hidden layer) and $\mathbf{W}^{t(2)}$ (hidden to output layer) from Equation (3.20) as well as the bias vectors $\mathbf{b}^{t(1)}$ and $\mathbf{b}^{t(2)}$.

Finally, we encode target data and source data via Equation (3.17) using the weights ($\mathbf{W}^{s(1)}$ and $\mathbf{b}^{s(1)}$) learned by the adaptive DAE. Then, this transformed representation of the source data is used to train a standard supervised classifier (e.g., SVMs) for a recognition system as shown in Figure 3.10, while the transformed target data is used for evaluation.

3.4.7 Extreme Learning Machine Autoencoders

Recently, *Extreme Learning Machine* (ELM) has been proposed for training single hidden layer feedforward neural networks since traditional BP algorithms for neural networks tend to converge to local optima and suffer from slow convergence. In ELM, the hidden nodes are randomly initiated and then fixed without iteratively tuning. The only trainable parameters are the weights between the hidden layer and the output layer. In this way, ELM is treated as a *linear-in-the-parameter* model which amounts to solving a linear system. Therefore, these trainable parameters can be analytically derived by solving a generalized inverse problem.

The advantages of ELM in efficiency and generalization performance over traditional BP algorithms have been demonstrated on a wide range of problems from different fields [221]. For example, even with randomly generated hidden nodes, ELM has fast learning speed and is prone to reach a global optimum. The ELM has also drawn considerable attention in the field of emotion recognition [40, 82, 222]. Han et al. [82] used a DNN as feature extractor to obtain the effective emotional features from short-term acoustic features (see Section 2.1.1) and fed the resulting utterance-level features to the ELM classifier. It is worth noting that, the generalization ability of ELM is comparable to SVMs and its variants [221].

Extreme Learning Machine Autoencoders (ELM-AEs) are a special case of ELM, where the output is equal to the input. Unlike the existing autoencoders used in neural networks, such as BP-based denoising autoencoders or sparse autoencoders, the input weights and biases are generated by searching the path back from a random space. Theoretical studies of ELM have shown that with commonly used activation functions, random feature mapping can maintain universal approximation capability, and more importantly, salient information can be exploited for hidden layer feature representation. In [223], Kasun et al. first showed empirically that ELM-AEs are comparable to DAEs and other DNN frameworks for a handwritten digit recognition task on the MNIST data. Nevertheless, ELM-AEs have not been used for transfer learning, to the best of my knowledge. This section gives a brief on the ELM-AE theory, and then demonstrates ELM-AE feature transfer learning, which is motivated by DAE feature transfer learning (see Section 3.4.4)

ELM-AEs are analogous to conventional autoencoders in terms of the topology structure and the forward pass, but have a particular training framework. An ELM-AE randomly generates the parameters of the hidden nodes $\mathbf{W}^{(1)}$ and $\mathbf{b}^{(1)}$, and only tunes the weights of the output layer $\mathbf{W}^{(2)}$ in the training phase. Given N training examples of $\mathbf{X}$, the input data is first mapped to a random feature space (called ELM feature space) by a nonlinear activation (associated with Equation (3.17)). In this case, the outputs of the final layer turn to be written as follows

$$\mathbf{Y} = \mathbf{W}^{(2)}\mathbf{H}, \tag{3.40}$$

where $\mathbf{H} = [\mathbf{h}_1, \dots, \mathbf{h}_N]$ are the hidden representations, $\mathbf{h} \in \mathbb{R}^m$, and m denotes the number of hidden units. The weights $\mathbf{W}^{(2)}$ are determined by minimizing the reconstruction error in the squared error sense

$$\arg\min \|\mathbf{W}^{(2)}\mathbf{H} - \mathbf{X}\|^2, \tag{3.41}$$

where $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]$ are the input data, $\mathbf{x} \in \mathbb{R}^n$, and n denotes the size of the input layer. It is evident that the cost function of the ELM-AE model is equivalent to that of the normal autoencoder model(see Section 3.4.2). Furthermore, different norms of output weights, such as L^2 norm, can be considered so as to achieve better generalization performance [221].

The output weights are then calculated by

$$\mathbf{W}^{(2)} = \mathbf{X}\mathbf{H}^{\dagger}, \tag{3.42}$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of matrix $\mathbf{H}$. In practice, instead of costly computing the Moore-Penrose inverse in the above expression, one can make use of the orthogonal projection method and add a regularization term C as suggested in [224] to improve the generalization capability and further obtain the solution faster

$$\mathbf{W}^{(2)} = \mathbf{X}\mathbf{H}^{T}\left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}^{T}\right)^{-1}, \tag{3.43}$$

where $\mathbf{I}$ is an identity matrix. Note that, if the size of the input layer is not equal to the number of hidden units (i.e., $n \neq m$), this thesis uses Equation (3.43) to compute the output weights, otherwise uses Equation (3.42).

The ELM-AE training algorithm can be summarized as follows:

1. Randomly generate the hidden node parameters $\mathbf{W}^{(1)}$ and $\mathbf{b}^{(1)}$.
2. Calculate the hidden layer outputs in the random feature space.
3. Compute the output weight matrix $\mathbf{W}^{(\mathbf{2})}$ through Equation (3.42) or Equation (3.43).

Different from the conventional autoencoder model which believes that hidden nodes serve as an encoder to produce the meaningful representations of the input data, ELM-AEs show that, output weights instead of hidden nodes serve as the encoder. From the point of view of the ELM theory, hidden nodes are important for learning, but do not need to be tuned and are independent of training data. Output weights correspond to building the transformation from the feature space to input data. It turns out that the representations $\tilde{\mathbf{H}}$, which are learned by ELM-AEs, are defined via the weights $\mathbf{W}^{(2)}$ in the form

$$\tilde{\mathbf{H}} = \left(\mathbf{W}^{(2)}\right)^{T}\mathbf{X}. \tag{3.44}$$

3.4.7.1 ELM Autoencoder Feature Transfer Learning

Previous studies have shown that both autoencoders and the variants are efficient models that have the ability to learn the structure of the data. Thanks to this important ability, a variety of autoencoder-based feature transfer learning methods have been proposed. The ELM-AE model not only inherits this natural ability from autoencoders, but also evolves towards more greater predictive power. This is because ELM can approximate any continuous target function provided the number of hidden nodes is large enough. In this light, this section presents an *ELM autoencoder feature transfer learning* algorithm, which is illustrated in Algorithm 3.7.

Algorithm 3.7 *Extreme Learning Machine Autoencoder* (ELM-AE) Feature Transfer Learning

```
Input: Unlabeled target data set $\mathbf{X}^t$, the labeled source set $\mathbf{X}^s$.
Output: Learned classifier $\mathcal{C}$ for the target task.
1: Randomly initialize ELM-AE($\theta$), $\theta = \left\{\mathbf{W}^{(1)}, \mathbf{b}^{(1)}\right\}$.
2: Train ELM-AE($\theta$) using $\mathbf{X}^t$, resulting in $\mathbf{W}^{(2)}$.
3: Obtain the encoder: Encoder($\mathbf{x}$) = $(\mathbf{W}^{(2)})^T\mathbf{x}$ (cf. Equation (3.44)).
4: Generate the target representation $\mathbf{H}^t$: $\mathbf{H}^t = \text{Encoder}_{\text{ELM-AE}}(\mathbf{X}^t)$.
5: Generate the source representation $\mathbf{H}^s$: $\mathbf{H}^s = \text{Encoder}_{\text{ELM-AE}}(\mathbf{X}^s)$.
6: Learn a classifier $\mathcal{C}$ by applying supervised learning algorithm $s$ (e.g., SVMs) to
   the source representations $\mathbf{H}^s$.
7: return The learned classifier $\mathcal{C}$.
```

Similar to DAE feature transfer learning (cf. Section 3.4.4), this algorithm leverages ELM-AEs to model the essential structure of the target data. To this end, an ELM-AE is trained with the target data in accordance with the ELM theory. Next, it creates the transformation with the learned output weights for compensation for the mismatch between the source domain and the target domain. This transformation is achieved by using Equation (3.44). Then, the transformed source data are used to train a supervised classifier. Ultimately, such a classifier is evaluated on the target data.

3.4.8 Feature Transfer Learning in Subspace

To cope with the typical inherent mismatch between the source data and target data, this section presents a feature transfer learning method using DAEs to build a high-order subspace of the source and target domain, where features in the source domain are transferred to the target domain by an additional regression neural network [225].

In the literature, there exists a large body of work in the spirit of feature transfer learning in subspace. The basic idea is to align the source and target data in the learned subspace by altering the distributions of either the source or the target data, or both. In [226], the authors proposed a method, called generalized transfer subspace learning through low-rank constraint, which can be applied to visual domain adaptation for object recognition. This method projects both, source and target data to a generalized subspace where each target example can be represented by a certain combination of source examples. By using a low-rank constraint during this transfer, the structures of the source and the target domains are retained. In doing so, good alignment between the domains is ensured through the use of only relevant data in some subspace of the source domain in reconstructing the data in the target domain. Furthermore, the discriminative power of the source domain is naturally passed on to the target domain. Gopalan et al. [167] discussed a two-stage unsupervised adaptation approach that generates intermediate domains between source and target to deal with the domain shift based on the Grassmann manifold. To this end, this approach learns the 'path' between the source and target domains by exploiting the geometry of their underlying space and by pursuing interpolations that are statistically optimal. It derives intermediate cross-domain data representations by sampling points along this path, and then trains a discriminative model using these representations.

Subspace learning can also be found in voice conversion [227, 228, 229], which aims to change specific information in the speech of a source speaker to that of a target speaker while preserving linguistic information. Basically, these voice conversion methods made use of some generative and graphical models, such as deep belief nets, conditional restricted Boltzmann machines, and recurrent temporal restricted Boltzmann machines, to build a high-order eigen space of the source/target speakers, where it is hopefully easier to convert the source speech to the target speech than in the original acoustic feature space such as cepstrum space. In addition to some graphical model, a neural network is adopted to connect and convert the speaker individuality abstractions in the high-order space. A noticeable advantage of these methods is that they have a deep nonlinear architecture, ensuring that the complex characteristics of speech can be captured more precisely than by a shallow model.

Motivated by the work [227], the author of this thesis proposed a feature transfer learning method by using a combination of DAEs and regression *Neural Networks* (NNs) [225]. The intuition behind this proposed approach is that the mismatch between the target and source domains in subspace is measured by a model, which thus alters the source data towards the target domain. Specifically, this approach first trains exclusive DAEs for source and target data in an unsupervised way so as to build two independent subspaces. By training a DAE for input data, the subspace gets implicitly grounded by the input data modality, allowing to give a high-order feature representation for each input example. Besides, it maps the target data into the source subspace as well. Then, a regression NN is used to discover the difference between the resulting features for target data in the source subspace and the ones in the target subspace. It is expected that the NN becomes a link which is able to compensate for the disparity between the source domain and the target domain as desired. Therefore, the proposed algorithm feeds the resulting high-order representations for the source data into the NN to predict new high-order representations in the target subspace. In turn, it leads to reducing the disparity between high-order features for source data. Ultimately, this framework uses the new high-order features for source data in the target subspace as training set and the original subspace features for target data as test set to carry out normal supervised algorithms for classification.

Figure 3.11 and Algorithm 3.8 depict an overview of the proposed method, which is composed of the following three steps. This approach first prepares two different

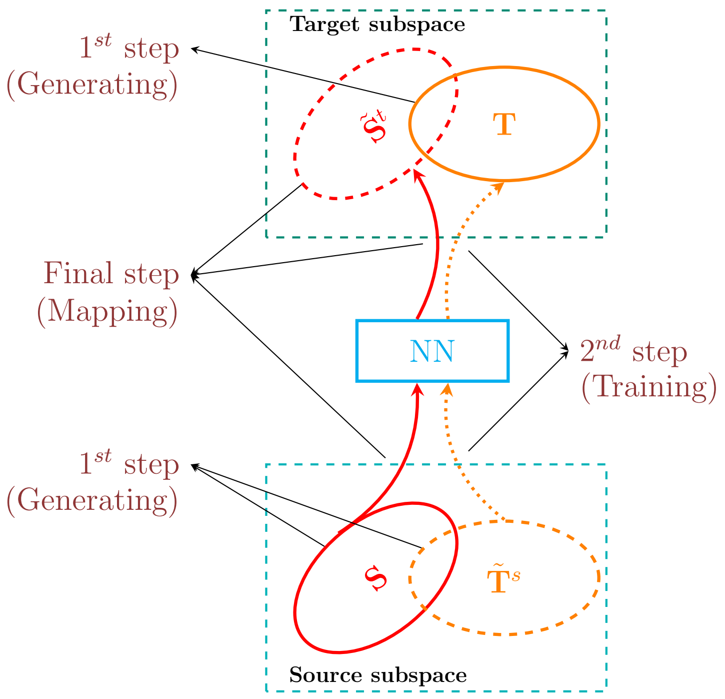


Figure 3.11: Overview of the proposed feature transfer learning in subspace method. The function "NN" refers to a regression neural network with one hidden layer. The sets $\mathbf{S}$ and $\tilde{\mathbf{S}}^t$ are high-order features in the source subspace and in the target subspace. The sets $\mathbf{T}$ and $\tilde{\mathbf{T}}^s$ are high-order features in the target subspace and in the source subspace.

Algorithm 3.8 Feature Transfer Learning in Subspace

```
Input: Unlabeled target data set $\mathbf{X}^t$, labeled source set $\mathbf{X}^s$.
Output: Learned classifier $\mathcal{C}$ for the target task.
1: Train DAE$^t$ using $\mathbf{X}^t$.
2: Train DAE$^s$ using $\mathbf{X}^s$.
3: Generate the target representations in the target subspace $\mathbf{T}$ via DAE$^t$.
4: Generate another target representations in the source subspace $\tilde{\mathbf{T}}^s$ via DAE$^s$.
5: Train NN with the target pairs $\mathbf{T}$ and $\tilde{\mathbf{T}}$.
6: Generate the source representations in the source subspace $\mathbf{S}$ via DAE$^s$.
7: Estimate the source representations in the target subspace $\tilde{\mathbf{S}}^t$ on $\mathbf{S}$ via NN.
8: Learn a classifier $\mathcal{C}$ by applying supervised learning algorithm $s$ (e.g., SVMs) to
   the source representations $\tilde{\mathbf{S}}^t$.
9: return The learned classifier $\mathcal{C}$.
```

DAEs for the source domain data $\mathbf{X}^s$ and the target domain data $\mathbf{X}^t$ so as to capture the domain-individuality information, which leads to generating the features in high-order subspace via encoding original features for the source or target data from the input layer to the hidden layer of the corresponding DAEs (see Equation (3.17)). It is worth noting that, the two DAEs are configured with the same number of hidden units. In addition, one can also generate the high-order features for the target data in source subspace, which is built by the DAE for the source domain. As a result, there are the high-order features of the source data in the source subspace $\mathbf{S}$, the features of the target data in the target subspace $\mathbf{T}$, and the features of the target data in the source subspace $\tilde{\mathbf{T}}^s$.

Next, a regression neural network, consisting of one hidden layer, is used to exploit the difference between the source subspace and the target subspace. At this point, the NN is trained to minimize the squared error for the target data between the high-order features $\mathbf{T}$ in the target subspace and its 'other' version $\tilde{\mathbf{T}}^s$ in the source subspace. Specifically, given a target example in the target subspace $\mathbf{h}^t \in \mathbf{T}$ and the respective version in the source subspace of it $\tilde{\mathbf{h}}^t \in \tilde{\mathbf{T}}^s$, the NN learns by solving the following optimization problem

$$\mathcal{J}^{\mathbf{NN}}(\theta) = \sum_{\substack{\mathbf{h}^t \in \mathbf{T} \\ \tilde{\mathbf{h}}^t \in \tilde{\mathbf{T}}^s}} \left\| \mathbf{h}^t - g(\tilde{\mathbf{h}}^t) \right\|^2, \tag{3.45}$$

where

$$\begin{aligned} g(\mathbf{x}) &= f(\mathbf{W}^{(2)}(f(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)})) + \mathbf{b}^{(2)}), \\ \theta &= \{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)}\}. \end{aligned} \tag{3.46}$$

Here $f(\cdot)$ is a nonlinear activation function, the parameters $\mathbf{W}^{(1)} \in \mathbb{R}^{k\times m}, \mathbf{W}^{(2)} \in \mathbb{R}^{m\times k}$ are the weights and $\mathbf{b}^{(1)} \in \mathbb{R}^k, \mathbf{b}^{(2)} \in \mathbb{R}^m$ are the bias terms, m denotes the number of hidden nodes for the DAE model, k denotes the number of hidden nodes for the NN. Note that, the size of the input layer is the same as the output layer in the special architecture of the NN.

Finally, the features of the source data $\mathbf{S}$ from the source subspace are transferred to the target subspace by means of the trained NN, which leads to a new form of the source data $\tilde{\mathbf{S}}^t$ in the target domain. In the end, the new form of the source data $\tilde{\mathbf{S}}^t$ and the features of the target data $\mathbf{T}$ will be taken to build a standard supervised classifier for speech emotion recognition in the following exemplary use-case.

4 Evaluation

Speech Emotion Recognition (SER) that focuses on recognizing emotional states from speech signals has drawn considerable attention in the past few decades since the dawn of emotion and speech research. Previous work in this field has delivered highly promising results for the community. Standard machine learning methods that have been proven successful for SER include HMMs using segmental features and SVMs using supra-segmental features. In addition to accurately making predictions, SER has recently turned to research producing reliable confidence measures (i.e., beyond simple posterior probabilities) for each prediction, which are crucial for any real-world application [230, 231].

However, there is very little work on the distribution mismatch problem. Schuller et al. [12] investigated different types of normalization (i.e., speaker normalization and corpus normalization) to deal with the high variances in cross-corpora evaluation experiments. Zhang [161] approached the challenge of data scarcity (i.e., small amount of labeled examples, limited number of speakers, and high-level labeling disagreement) by applying semi-autonomous data enrichment and optimization approaches to take advantage of richly unlabeled data.

Different from previous work, this chapter focuses on using the feature transfer leaning methods presented in Chapter 3 to overcome the distribution mismatch problem in SER. Accordingly, comprehensive experiments regarding SER are designed to justify the effectiveness of these methods. In this chapter, first, a set of eight emotional speech databases is introduced, which are used for experimental evaluations. Then, an experimental setup including the descriptions of a generic label mapping and the selected feature set is given in Section 4.2. Next, the effectiveness of sparse autoencoder (SAE) feature transfer learning is evaluated on six databases in Section 4.3. Afterwards, Sections 4.4 to 4.6 present a systemic evaluation of the shared-hidden-layer autoencoder (SHLA) feature transfer learning, adaptive denoising autoencoder (A-DAE) feature transfer learning, and feature transfer learning in subspace, respectively. Based on two public databases, finally, Section 4.7 shows that whispered speech emotion recognition can also benefit from autoencoder feature

Table 4.1: Overview of the 8 selected speech emotion databases: database 1–4 shown here (ABC, AVIC, EMO-DB, and eNTERFACE). Age (adults or children). Recording environment (env.). Sampling rate (Rate). Number of female (# female) and male (# male) subjects. Number of utterances per binary valence (# Valence negative (neg.), # Valence positive (pos.)), and overall number of utterances (# All).

Corpus	ABC	AVIC	EMO-DB	eNTERFACE
Age	adults	adults	adults	adults
Language type	German	English	German	English
Content	fixed	variable	fixed	fixed
Emotion type	induced	natural	acted	acted
Recording env.	studio	studio	studio	normal
Time (hh:mm)	1:15	1:47	0:22	1:00
Rate (kHz)	16	44	16	16
# male	4	11	5	34
# female	4	10	5	8
# Valence neg.	213	553	352	855
# Valence pos.	217	2 449	142	422
# All	400	3 002	494	1 277

transfer learning.

4.1 Emotional Speech Databases

To comprehensively investigate the performance of the proposed feature transfer learning methods, a large number of eight well known and public available emotional speech databases have been chosen to cover traditional acted emotional speech to fully natural and spontaneous affective speech, children speech, adult speech, German speech, English speech, French speech, and whispered speech. Specifically, the chosen databases include ABC, AVIC, EMO-DB, eNTERFACE, FAU AEC, GEWEC, and SUSAS. Statistic for the eight emotional speech databases are summarized in Tables 4.1 and 4.2. Naturally, these databases emerge the inherent database biases for cross-corpus experiments, which lead to problems worth to be addressed: the cross-speaker problem (associated with the speaker independent problem), the cross-language problem (English vs. German, or French vs. German), the cross-age problem (children vs. adults), and the cross-speech-mode problem (normal phonated mode vs. whispered mode).

Table 4.2: Overview of the 8 selected speech emotion databases: database 5–8 shown here (FAU AEC, GEWEC, SUSAS, VAM). Age (adults or children). Recording environment (env.). Sampling rate (Rate). Number of female (# female) and male (# male) subjects. Number of utterances per binary valence (# Valence negative (neg.), # Valence positive (pos.)), and overall number of utterances (# All).

Corpus	FAU AEC	GEWEC	SUSAS	VAM
Age	children	adults	adults	adults
Language type	German	French	English	German
Content	variable	fixed	fixed	variable
Emotion type	natural	acted	natural	natural
Recording env.	normal	studio	noisy	noisy
Time (hh:mm)	9:20	0:13	1:01	0:47
Rate (kHz)	16	44.1	8	16
# male	21	2	4	15
# female	30	2	3	32
# Valence neg.	5 823	640	1 616	876
# Valence pos.	12 393	640	1 977	71
# All	18 216	1280	3 593	947

4.1.1 Aircraft Behavior Corpus

The *Aircraft Behaviour Corpus* (ABC) [232] was introduced for the special application of automatic public transport surveillance about passenger emotions. During the recording, a certain emotion was induced by a script, which guided the subjects through a given storyline: Pre-recorded cabin announcements controlled by an unseen test-conductor were automatically played back by five speakers at different positions. Subjects had to imagine that they are on a vacation (and return) flight, made of six scenes: take-off, serving of wrong food, turbulence, sleeping, talking to the person in the next seat, and landing. The scene setup consisted of an airplane seat for the subject, which was put in front of a blue screen. Eight German-speaking subjects in gender balance from 25 to 48 years (average 32 years) old actively participated in the recording. 11.5 h of audiovisual recording material was obtained and – after pre-segmentation – was annotated independently by three experienced male labelers using a pre-defined, closed set of emotion categories, including *neutral*, *tired*, *aggressive*, *cheerful*, *intoxicated*, and *nervous*. In total, there are 431 clips with an average per-clip length of 8.4 s.

4.1.2 Audio-Visual Interest Corpus

The *Audio-Visual Interest Corpus* (AVIC) introduced by Schuller et al. [233] provides spontaneous emotion samples of non-restricted spoken content. It was used as dataset for the INTERSPEECH 2010 Paralinguistics Challenge [234]. In its scenario setup, a product presenter leads a subject through an English commercial presentation. In the recording, there are 21 subjects (10 female). The *Level of Interest* (LOI) is annotated for every sub-turn (pause based sub-division of speaker turns) with three discrete labels ranging from *boredom* (the subject is bored with the conversation and/or the topic, is very passive, and does not follow the discourse; also referred to as *LOI1*), over *neutral* (the subject follows and participates in the discourse and it can not be recognized, whether the subject is interested in or indifferent towards the topic; also referred to as *LOI2*) to *joyful* interaction (presenting a strong wish of the subject to talk and learn more about the topic; also referred to as *LOI3*). These three discrete levels were obtained from *Majority Voting* (MV) over four individual raters opinions and after combining the original 5 level annotation to only 3 levels to avoid too much sparsity in some of the 5 levels. For the evaluations in this thesis, all 3 002 phrases are used, in contrast to only 996 phrases with high inter-label agreement as e.g., utilized in [233].

4.1.3 Berlin Emotional Speech Database

A well known set of normal phonated emotional speech – the *Berlin Emotional Speech Database* (EMO-DB) [235] – is chosen to test the effectiveness of SER. It covers *anger*, *boredom*, *disgust*, *fear*, *happiness*, *neutral*, and *sadness* as speaker emotions. The spoken content is again pre-defined by ten German emotionally neutral sentences like *"Der Lappen liegt auf dem Eisschrank"* (*The cloth is lying on the fridge.*). Ten (five female) professional actors speak ten sentences. The actors were asked to express each sentence in all seven emotional states. The sentences were labeled according to the state they should be expressed in, i.e., one emotion label was assigned to each sentence. 494 phrases are marked as minimum 60 % natural and minimum 80 % agreement by 20 subjects in a listening experiment. This selection is usually used in the literature reporting results on the corpus (e.g., [236, 237, 238]), therefore, it is also used for this thesis.

4.1.4 eNTERFACE Database

The eNTERFACE database [239] is a publicly audiovisual emotion corpus. The emotional categories contain *anger*, *disgust*, *fear*, *joy*, *sadness*, and *surprise*. 42 subjects (eight female) from 14 nations participated in the recording. The recording scenario is an office environment, where pre-defined English spoken content are provided. To induce the emotions, each subject was instructed to listen to six successive short stories, with each story eliciting a certain emotion. Afterwards, they had to react to each of the situations by speaking previously read phrases that fit the short story. Five phrases are available for each emotion, such as *"I have nothing to give you! Please don't hurt me!"* in the case of fear. Two labelers independently judged whether the reaction expressed the induced in an unambiguous way. Only in this case, this sample was added to database. In total, there are 1 277 samples in the database.

4.1.5 FAU Aibo Emotion Corpus

The *FAU Aibo Emotion Corpus* (FAU AEC) [240, 241] is well known to the SER community as was adopted for the INTERSPEECH 2009 Emotion Challenge task [25]. It features recordings of 51 children interacting with Sony's pet robot Aibo, using a *Wizard of Oz* (WOZ) setup. The corpus comprises spontaneous, German speech that is emotionally colored. The children at age of 10–13 years from two different schools were made believe that the Aibo was responding to their commands, whereas the robot was actually remote-controlled and did not respond to their commands. Sometimes the robot disobeyed in order to elicit negative emotional actions. Speech was transmitted with a high quality wireless head set and recorded with a DAT-recorder (16 bit, 48 kHz, down-sampled to 16 kHz). The recordings were segmented automatically into "turns" by splitting the speech with a pause threshold of 1 s. Five advanced students of linguistics listened to the turns in sequential order and annotated each word independently from each other as neutral (default) or as belonging to one of ten other classes. Since many utterances are only short commands and rather long pauses can occur between words due to Aibo's reaction time, the emotional/emotion-related state of the child can change also within turns. Hence, the data is labeled on the word level. The MV is carried out: if three or more labelers agreed, the label was attributed to the word. In the following, the number of cases with MV is given in parentheses: *joyful* (101), *surprised* (0), *emphatic* (2 528), *helpless* (3), *touchy*, i.e., *irritated* (255), *angry* (84), *motherese* (1 260), *bored* (11), *reprimanding* (310), *rest*, i.e., non-neutral, but not belonging to the other categories (3), *neutral* (39 169); 4 707 words had no MV; all in all, there were 48 401 words. The unit of analysis are not single words, but semantically meaningful, manually defined chunks (18 216 chunks, 2.66 words per chunk on average). Heuristic algorithms were used to map the decisions of the five labelers on the word level onto a single emotion label for the whole chunk.

This thesis concentrates on the two-class problem consisting of the cover classes NEGative (subsuming *angry*, *touchy*, *reprimanding*, and *emphatic*) and IDLe (consisting of all non-negative states). Speaker independence is guaranteed by using the data of one school (OHM, 13 male, 13 female) for training and the data of the other school (MONT, 8 male, 17 female) for testing. Note that, a ground truth label as well as a human agreement score are assigned for each instance in the database. There are two reasons for chunks with low emotional human agreement: 1) not all words in a NEG chunk have to be NEG themselves; some words may also be produced in the state IDL. 2) Even if all words in a chunk are labeled as NEG, the agreement of the five labels for single words may be low, e.g., 3 out of 5. Of course, combinations of both phenomena can occur as well. A detailed description of the FAU AEC database can be found in [25, 240, 241].

4.1.6 Geneva Whispered Emotion Corpus

In addition to the above outlined databases which only contain normal phonated speech, one database, called *Geneva Whispered Emotion Corpus* (GEWEC), containing whispered speech is employed to evaluate the effectiveness of the feature transfer learning methods. The corpus provides normal phonated/whispered paired utterances. Two male and two female professional French-speaking actors in Geneva were recruited to speak eight predefined French pseudo-words (*"belam"*, *"molen"*, *"namil"*, *"nodag"*, *"lagod"*, *"minad"*, and *"nolan"*) with a given emotional state in both normal phonated and whispered speech modes following the lead given by the *Geneva Multimodal Emotional Portrayals* (GEMEP) corpus that was used in the INTERSPEECH 2013 Computational Paralinguistics Challenge [242]. Speech was expressed in four emotional states: *angry*, *fear*, *happiness*, and *neutral*. The actors were requested to express each word in all four emotional states five times. The utterances were labeled based on the state they should be expressed in, i.e., one emotion label was assigned to each utterance. As a result, GEWEC consists of 1 280 instances in total. To give an in-depth evaluation of the proposed method, labels for binary valence/arousal from the emotion categories were further generated. In the valence dimension, angry and fear have negative valence, happiness and neutral have positive valence. In the arousal dimension, neutral is assigned as low arousal; angry, happiness, and fear are assigned as high arousal. An overview of the corpus is found in Table 4.9.

Recording was done in a sound proof chamber using professional recording equipment. All recordings were recorded with a 16 bit PCM encoded single channel at a sampling rate of 44.1 kHz. The distance from the microphone was about 0.5 m during recording. Recordings were accompanied by visual cues on a screen, which indicated which word has to be vocalized and which emotional state needs to be expressed. Cues were visible on the screen for 1 s length, separated by a blank screen of 2 s. The cue duration of 1 s was chosen such that the actors were guided to vocalize each word with a duration of about 1 s, which ensures that the vocalizations were comparable in length.

Pre-processing steps were applied to each utterance before feature extraction, in which all utterances were normalized to mean energy, as well as scaled to a mean of 70 dB *Sound Pressure Level* (SPL) and added manually a fade-in/fade-out duration of 15 ms.

4.1.7 Speech Under Simulated and Actual Stress

The *Speech Under Simulated and Actual Stress* (SUSAS) [243] database was a first reference for spontaneous recordings. To increase the difficulty, speech is partially masked by field noise. The 3 593 actual stress speech samples are used for the upcoming evaluation in this thesis, which were recorded in subject motion fear and stress tasks. Seven subjects (three female) in roller coaster and free all actual stress situations are contained in this set. Next to *neutral* speech and *fear* two different stress conditions have been collected: *medium stress*, and *high stress*, and *screaming*. SUSAS is also restricted to a pre-defined spoken content of 35 English air-traffic commands, such as “brake”, “help” or “no”. Thus, only single words are contained.

4.1.8 Vera Am Mittag Database

The *Vera am Mittag* (VAM) database [244] includes audiovisual recordings extracted from the German TV talk show (i.e., Vera Am Mittag). The corpus used consists 946 spontaneous and emotionally colored utterances from 47 guests of the talk show, which were recorded from unscripted and authentic discussions. The topics were mainly personal issues, such as friendship crises, fatherhood questions, or love affairs. To obtain non-acted emotions, the guests were not told that the recordings were going to be analyzed for scientific purposes. For annotation of the speech data, the audio recordings were manually segmented to the utterance level, whereas each utterance contained at least one phrase. A large number of human annotators were involved with rating the data (17 annotators for one half of the data, six for the other). The labeling bases on a discrete five scale for three dimensions mapped onto the interval of $[-1, 1]$: the average results for the standard deviation are 0.29, 0.34, and 0.31 for valence, activation, and dominance. The averages for the correlation between the evaluators are 0.49, 0.72, and 0.61, respectively. The correlation coefficients for activation and dominance show suitable values, whereas the moderate value for valence indicates that this emotion primitive was more difficult to evaluate, but may partly also be a result of the smaller variance of valence.

4.2 Experimental Setup

Most of the experiments are based on the first emotion recognition challenge, i.e., the INTERSPEECH 2009 Emotion Challenge. Hence, the overview of the training set and the test set of the dataset used for this challenge is particularly shown in Table 4.3. Apart from the database, the feature set is introduced in Section 4.2.2.

It is well known that training a neural network is difficult and time-consuming. Autoencoders also have the same difficulty. To facilitate training an autoencoder, the

Table 4.3: Number of examples for the 2-class problem of the FAU Aibo Emotion Corpus (FAU AEC), which was used for the INTERSPEECH 2009 Emotion Challenge [25]. Negative emotions (NEG); Neutral and positive emotions (IDL).

#	NEG	IDL	$\sum$
Train	3 358	6 601	9 959
Test	2 465	5 792	8 257
$\sum$	5 823	12 393	18 216

toolkit minFunc[1] was applied which implements L-BFGS to optimize the parameters of the autoencoders. Logistic sigmoid functions are always chosen as the activation function for autoencoders. UAR is always chosen as a primary performance metric, which has also been the competition measure of the first challenge on emotion recognition from speech [25] and follow-up ones. It equals the sum of the recalls per class divided by the number of the classes, and appears more meaningful than overall accuracy in the case of presence of class imbalance. Besides, WAR is used as the secondary metric. To validate statistical significance of the results, a one-sided z-test is taken. As for the basic supervised learner in the classification step, SVMs with the $L2$-regularized $L2$-loss support vector classifier implemented in LIBLINEAR [99] are used. Throughout the experiments, a fixed penalty factor $C = 0.5$ for the linear SVMs is used.

For appropriately selecting the hyper-parameters of the autoencoders, k-fold cross validation is adopted. Therefore, the training set is split into four folds ($k = 4$) and each model is trained four times with a different fold held out as validation data. The predictions made by the four models are used to obtain a UAR value when reporting test set results. According to the performance on the validation data, the best particular model in each family of models is chosen.

4.2.1 Mapping of Emotions

In order to generate a unified set of labels across databases, the diverse emotion groups are mapped onto the valence axis in the dimensional emotion model. Based on psychology theory, categorical emotions can be decomposed into valence and arousal (or activation in some studies) in continuous dimensions [245, 246]. Valence (i.e., positive vs. negative) subjectively describes a feeling of pleasantness or unpleasantness, while arousal (i.e., low vs. high) subjectively describes a state of feeling activated or deactivated. Valence and arousal are the best established and widely used emotional dimensions at present [20]. In this thesis, valence is mainly

[1]`http://www.di.ens.fr/~mschmidt/Software/minFunc.html`

Table 4.4: Emotion categories mapping onto negative and positive valence for the eight selected databases.

Corpus	Negative	Positive
ABC	aggressive, nervous, tired	cheerful, intoxicated, neutral
AVIC	boredom	neutral, joyful
EMO-DB	anger, boredom, fear, disgust, sadness	joy, neutral
eNTERFACE	anger, disgust, fear, sadness	joy, surprise
FAU AEC[a]	negative	idle
GEWEC	anger, fear	happiness, neutral
SUSAS	high stress, screaming, fear	medium stress, neutral
VAM[b]	q4, q3	q2, q1

[a] Label negative and idle correspond to the 2-class labels defined in the FAU AEC database.
[b] Abbreviations q1-q4: quadrants in the arousal-valence plane.

investigated simply because this thesis sticks to the INTERSPEECH 2009 Emotion Challenge two-class task [25], where binary valence was featured. Hence, Table 4.4 gives these mapping only regarding valence. But both, valence and arousal are considered when the GEWEC database (Section 4.1.6) is selected. In fact, these mappings are based on the original mappings, as suggested in [247] and adopted for cross-corpus experiments [12, 24]. It is worth noting, that the controversial issue of the mapping of neutral may arise, since in theory, it should be projected onto a third state rather than positive valence. However, because not all databases included a neutral state, it was decided for a binary mapping here in order to be able to evaluate performances across database using the same labels and have two more balanced binary classes for each database. Hence, neutral is popularly mapped to low arousal and positive valence. Thus, as shown in Table 4.4 for the FAU-AEU database (Section 4.1.5), the idle label belongs to the positive valence label.

4.2.2 Features

As for acoustic features, a standardized feature set is chosen as is provided by the INTERSPEECH 2009 Emotion Challenge [25] which contains 12 functionals applied to 2×16 acoustic LLDs including their first order delta regression coefficients (Δ) as shown in Table 4.5. In detail, the 16 LLDs are MFCC 1–12, RMS frame energy,

Table 4.5: Overview of the standardized INTERSPEECH 2009 Emotion Challenge feature set [25].

LLDs (16×2)	Functionals (12)
(Δ) MFCC 1–12	Arithmetic Mean
(Δ) RMS Energy	Moments: SD, kurtosis, skewness
(Δ) ZCR	Extremes: value, rel. position, range
(Δ) Prob. of voicing, F0	Linear Regression: offset, slope, MSE

Zero Crossing Rate (ZCR) from the time signal, probability of voicing from the autocorrelation function, and the pitch frequency F0 (normalized to 500 Hz). Then, 12 functionals – arithmetic mean, moments including the *Standard Deviation* (SD), kurtosis, and skewness, four extremes (i.e., minimum and maximum value, relative position, and ranges) as well as two linear regression coefficients with their *Mean Square Error* (MSE) – are applied to the LLDs and their deltas. Thus, the total feature vector per utterance contains $16 \times 2 \times 12 = 384$ attributes. To ensure reproducibility, the open source openSMILE[2] toolkit version 2.0 [23, 52], which has matured to be a standard for feature extraction in automatic speech emotion recognition, was used with the pre-defined challenge configuration. More details on the feature set can be found in [24, 25].

4.3 SAE Feature Transfer Learning

4.3.1 Experiments

Sparse Autoencoder (SAE) feature transfer learning (Section 3.4.3) assumes that a small amount of labeled data from the target domain are available. In the experiments, six standard databases are chosen: The FAU AEC set is treated as target set, which consists of a training and test partition (roughly half and half) naturally given by recordings at different elementary schools, while eNTERFACE, SUSAS, EMO-DB, VAM, and AVIC seve as source set. To implement the sparse autoencoder algorithm, a small part of examples (the size ranging from 50 to 950 chunks) are randomly chosen from the FAU AEC training set to obtain a common feature structure, where the same number of examples are chosen from positive valence and negative valence. In the sparse autoencoder learning process, the number of hidden units was fixed to 200, and the sparsity level ρ was set to 0.01. The reported performance in UAR is the average over 20 runs to avoid 'lucky' or 'unlucky' selection. Then, the common feature structure is used to reconstruct each source database, as described

[2]`http://sourceforge.net/projects/opensmile/`

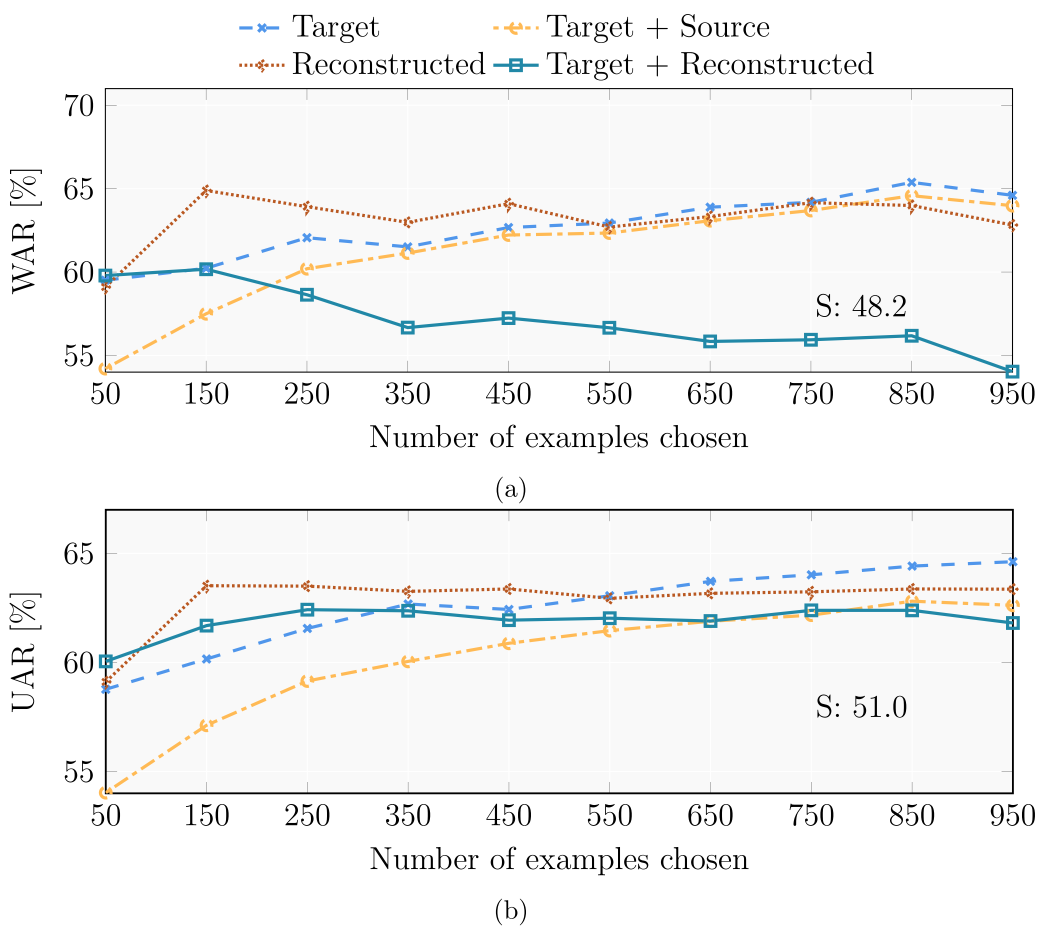
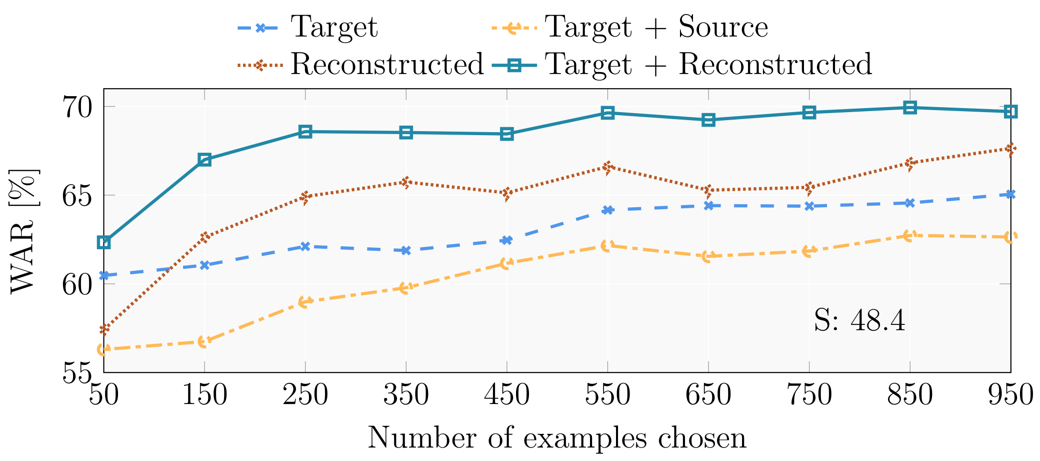


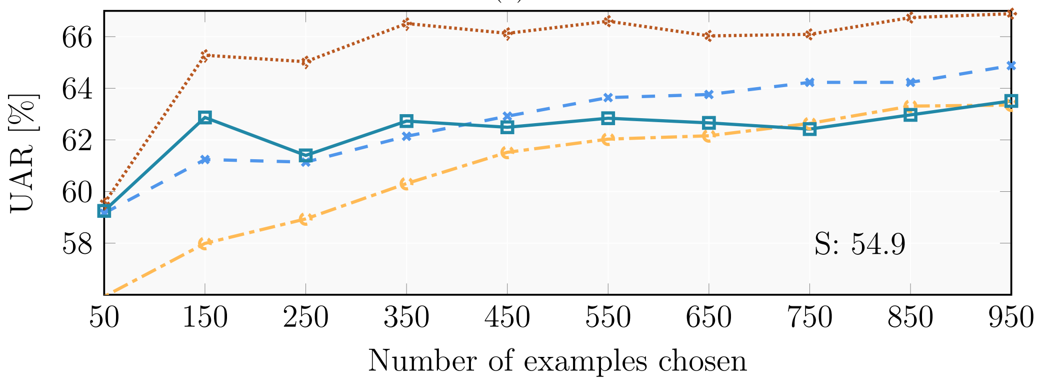
Figure 4.1: WAR and UAR comparison for the increase of number of examples chosen from the FAU AEC training set for the source data eNTERFACE. S:##.# is the WAR and UAR if only using source data. Reconstructed: classifier trained on reconstructed source data by a sparse autoencoder method. Target + Reconstructed: classifier trained on target and reconstructed source data. Target: classifier trained on target data. Target + Source: classifier trained on target and original source data.

in Algorithm 3.3. Finally, the FAU AEC test data are classified by the classifier trained on the reconstructed data.

As for metrics, both UAR and WAR are selected. Furthermore, the hyper-parameters of all SVMs are chosen by cross-validation on the training set. Before training SVMs, furthermore, the *Synthetic Minority Oversampling Technique* (SMOTE) [248] is always applied to balance training examples between the positive and negative classes. For a two-class problem, the chance level thus always resembles 50.0 % UAR. Following the setup given in Section 4.2, here the baseline UAR for the



(a)



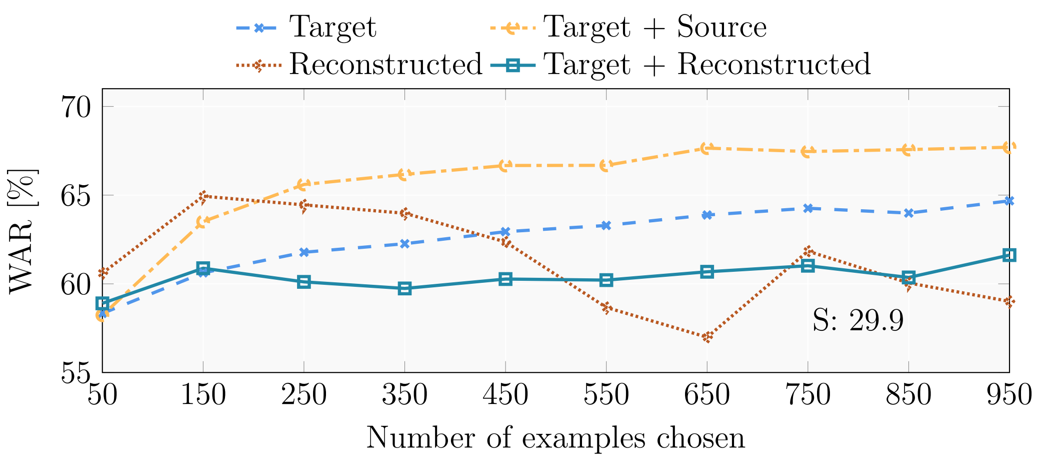
(b)

Figure 4.2: WAR and UAR comparison for the increase of number of examples chosen from the FAU AEC training set for the source data SUSAS. Explanations: cf. Figure 4.1.

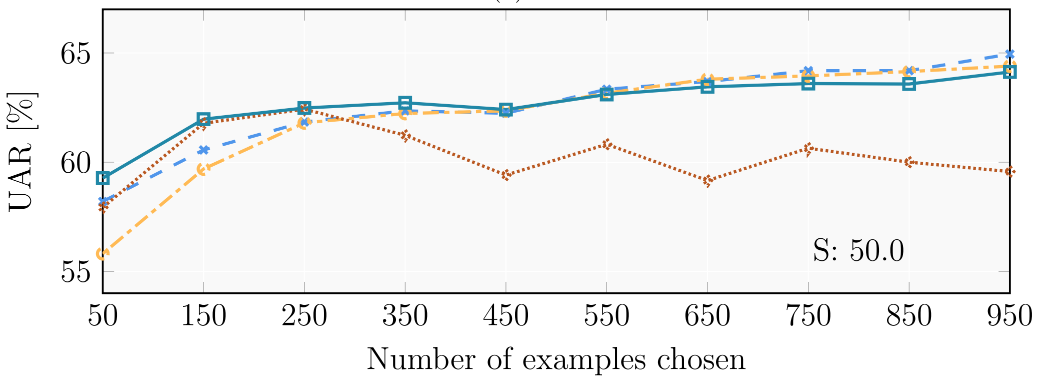
FAU AEC two-class task is 66.9 %.

4.3.2 Experimental Results

During the evaluation, a variety of combinations of the target data, the reconstructed data, and the source data, were considered in order to provide a full picture of the suggested method's effects. Figures 4.1 and 4.2 report the results for the source data being eNTERFACE and SUSAS. Reconstructed data by the sparse autoencoder, possibly in combination with target data, significantly (one sided z-test) outperform the target data alone. For the eNTERFACE database with induced emotion types, sparse autoencoder data achieves mostly the highest test UAR and WAR when the number of chosen examples is in the range of 50 to 550. For instance, the reconstructed data's UAR obtains 63.5 % compared with only the



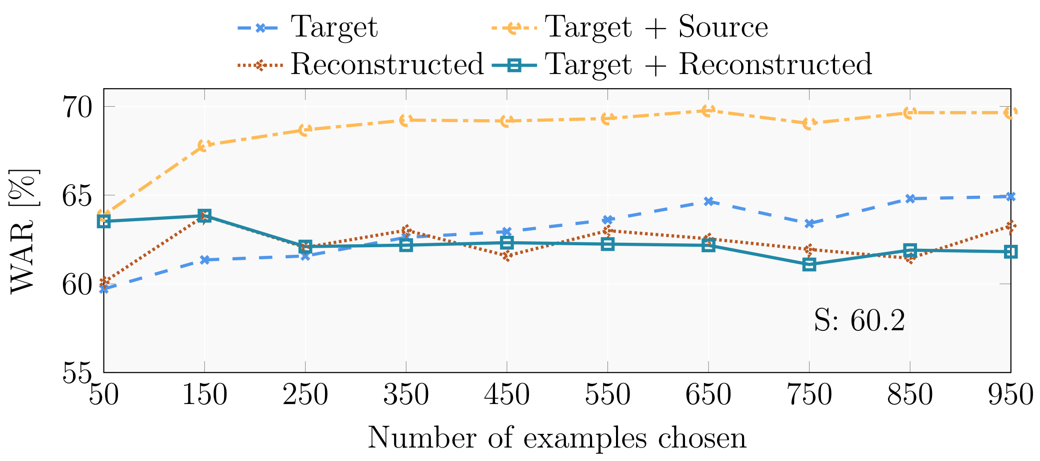
(a)



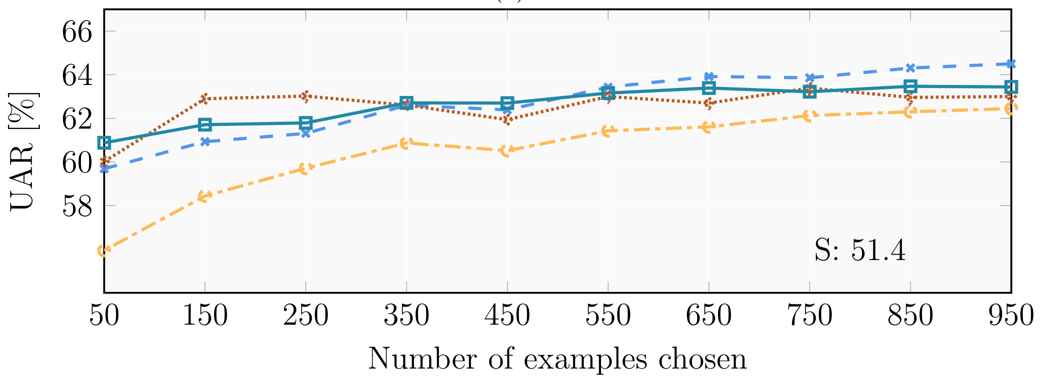
(b)

Figure 4.3: WAR and UAR comparison for the increase of number of examples chosen from the FAU AEC training set for the source data EMO-DB. Explanations: cf. Figure 4.1.

target's UAR of 60.1 %, the target and the reconstructed data's UAR of 61.6 %, and the target and the source data's UAR of 57.1 %, while 150 target examples are used. Afterwards, when the size of target training continues increasing, the performance of target data gradually overtakes the sparse autoencoder data since no more extra information in the eNTERFACE can be transferred to the FAU AEC target domain. In comparison with eNTERFACE, SUSAS's actual stress data, which is collected in a noisy recording, always obtains the highest test UAR. At 150 target examples available, the reconstructed data's UAR reaches 65.2 % which is sharply larger than only the target's UAR of 61.2 %, the target and reconstructed data's UAR of 62.8 %, and the target and source data's UAR of 57.9 %. It is worth noting that, with the increase of target training size, its UAR stably goes up to 66.8 % at 950 target examples available, which approaches the baseline UAR 66.9 % with the whole FAU



(a)

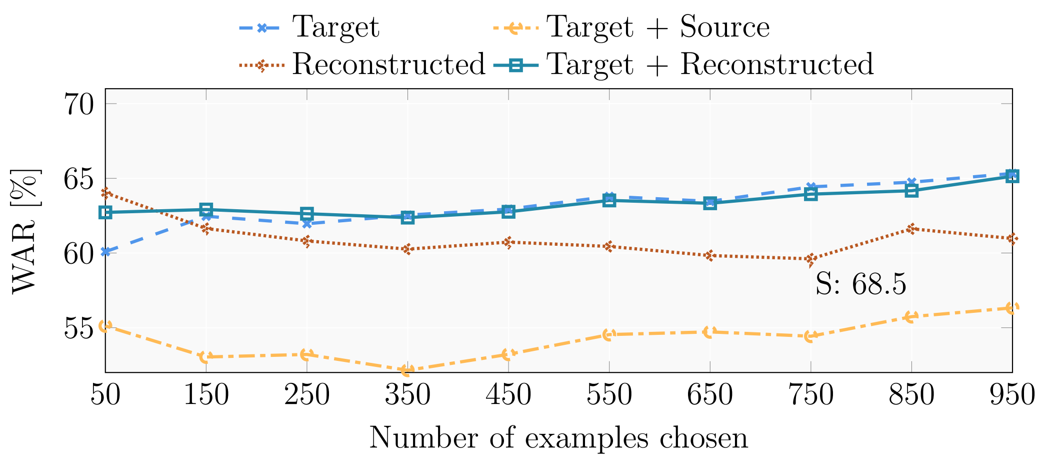


(b)

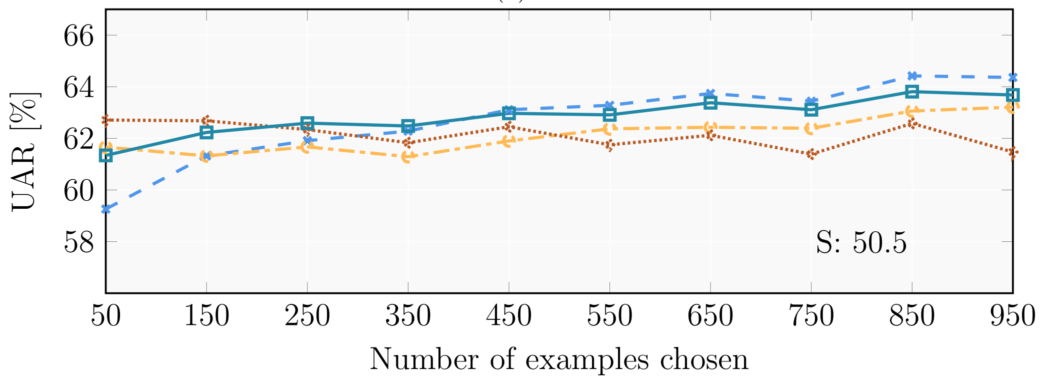
Figure 4.4: WAR and UAR comparison for the increase of number of examples chosen from the FAU AEC training set for the source data VAM. Explanations: cf. Figure 4.1.

AEC training set (9 958 examples) applied. Surprisingly, the WAR comparison for the increase of number of examples has the similar trend as the UAR trend. For eNTERFACE, for example, the data reconstructed by the sparse autoencoder reaches the WAR of 64.9 % with the 150 target examples available when compared to as the source's WAR of 48.2 %.

Experimental results on the source data EMO-DB, VAM, and TUM AVIC are shown in Figures 4.3 to 4.5. As for EMO-DB with acted emotion, note that, the sparse autoencoder method cannot transfer more useful information from the source with the increase of target training size. Instead, its performance decreases unexpectedly, which might indicate that negative transfer happens because EMO-DB and FAU AEC are highly dissimilar. However, the method of combining target data with reconstructed data steadily rises in line with the size of the target data. For



(a)



(b)

Figure 4.5: WAR and UAR comparison for the increase of number of examples chosen from the FAU AEC training set for the source data AVIC. Explanations: cf. Figure 4.1.

the source data being VAM, the sparse autoencoder method performs better within the range of target size ranging from 150 to 350. Afterwards, the performance of the sparse autoencoder is still comparable with the method of using target data. For the final source database TUM AVIC in the English language, there is no significant improvement compared to the method of using target data. Nevertheless, it is worth noting that, its UAR of the reconstructed data fluctuates around 62.5 %, and this UAR value (62.7 %) at 50 target examples available is dramatically more than the average UAR values over the other source data (59.1 %). If only a small number of data are available in the target domain, e.g., only 50 examples, Figure 4.6 shows UAR values for each source data and the corresponding reconstructed data. As can be seen from Figure 4.6, when those source data as training set are input to an emotion recognition system, respectively, only the chance level UAR is obtained for

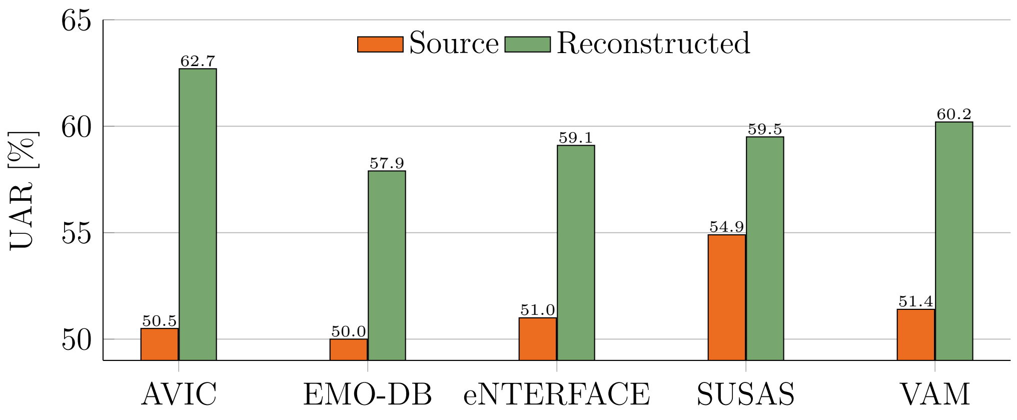


Figure 4.6: UAR comparison when 50 examples are chosen from the target data. Average: 51.6 % UAR (source) and 59.9 % UAR (reconstructed).

the two-class task of FAU AEC. But the reconstructed data (average UAR 59.9 %) significantly outperform the original source data (average UAR 51.6 %), which means that knowledge transferred by the sparse autoencoder is useful for the classification in emotion recognition. The performance improvement over each original source data is large even though very few target data examples are used.

4.3.3 Conclusions

In this section, the proposed sparse autoencoder feature transfer learning method, which uses a single-layer autoencoder to find a common structure in small target data and then applies such structure to reconstruct source data in order to complete useful knowledge transfer from source data into a target task was evaluated on six publicly available corpora. In this method, each single-layer autoencoder focuses on discovering nonlinear generalization of class-specific target examples. The reconstructed data are to build a speech emotion recognition engine for a real-life task as given by the INTERSPEECH 2009 Emotion Challenge. Experimental results show that the proposed algorithm effectively transfers knowledge and further enhances the classification accuracy. Besides, the results confirm some observations that transfer learning can deliver a higher start as well as speed up the growth of performance expected (see Section 3.2).

4.4 SHLA Feature Transfer Learning

4.4.1 Experiments

Unlike sparse autoencoder feature transfer learning, shared-hidden-layer autoencoder (SHLA) feature transfer learning copes with the typical inherent mismatch between different corpora in acoustic emotion when no labeled data from the target domain exists. As stated in Section 3.4.5, this approach learns common feature representations shared across the training and test set in order to reduce the discrepancy in them. To exemplify effectiveness of this approach, the INTERSPEECH Emotion Challenge's FAU AEC is selected as test database and two other databases are used as training set for extensive evaluation.

In the SHLA learning process, the number of hidden units m was fixed to 200, and attempted hyper-parameter γ and weight decay values λ were the following : $\gamma \in \{0.1, 0.3, 0.5, 1, 2, 3\}$, $\lambda \in \{0.0001, 0.001, 0.01, 0.1\}$.

Models for Comparison

The following methods are provided for comparison:

- *Matched Instance Number Training* (MINT): randomly (repeated ten times) picks a number of examples from the FAU AEC training set to train an SVM, i.e., without the need of transferring in intra-corpus scenario. For comparison, this number is given by the number of learning examples as in the ABC or SUSAS sets, respectively.

- *Cross Training* (CT): uses ABC or SUSAS to train the standard (SVM) classifier, i.e., without using SHLA-based representation learning.

- KMM: utilizes the KMM (see Section 3.2.1.1) on the ABC and SUSAS database for covariate shift adaptation. The 'tuning parameters' in KMM follow heuristics adopted in [170, 172].

- DAE: employs denoising autoencoders for representation learning in order to match training examples to test examples (Section 3.4.4.1), which was successfully applied to the transfer learning challenge and domain adaptation [166, 210].

- SHLA: uses the proposed SHLA to extract common features on the training and target test set, then trains standard SVMs using the learned features and labels in the training set.

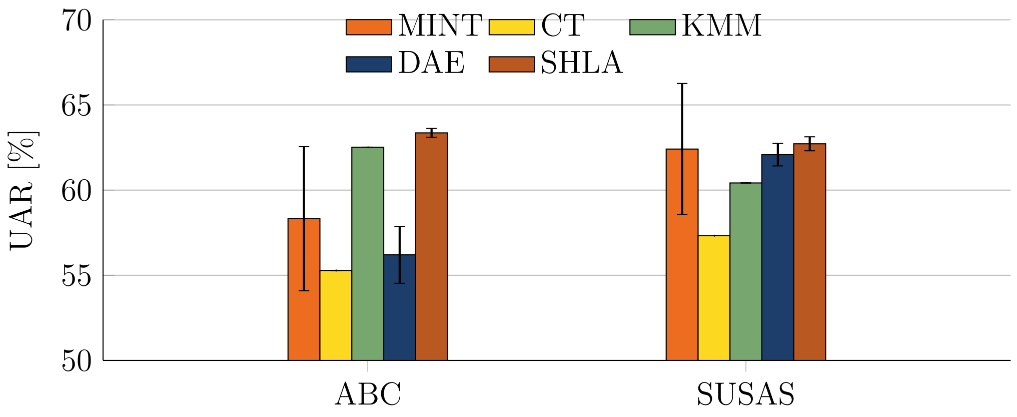


Figure 4.7: Cross-corpus average UAR over ten trials using matched instance number training (MINT), cross training (CT), the covariate shift adaptation KMM, and the proposed SHLA for ABC and SUSAS.

Table 4.6: Cross-corpus average UAR over ten trials for the training sets ABC and SUSAS.

UAR [%]	MINT	CT	KMM	DAE	SHLA
ABC	58.32	55.28	62.52	56.20	**63.36**
SUSAS	62.41	57.32	60.41	62.08	**62.72**

4.4.2 Experimental Results

First, the cross-corpus scenario is evaluated, where acoustic emotion recognition models are trained on ABC or SUSAS while tested on the FAU AEC test set (except the MINT condition that uses FAU AEC data for training). The experiments are run ten times for MINT, DAE, and SHLA methods that involve random sampling. The averaged UAR over the ten trials is visualized in Figure 4.7, including the error bar, and given quantitatively in Table 4.6. As can be seen, the SHLA method outperforms all the other approaches.

More specifically, for the small database ABC, one can easily see that, the two standard methods (CT and MINT) only obtain average UAR around the chance level (55.28 % and 58.32 %). When the accuracy obtained by the DAE method reaches 56.20 %, the covariate shift adaptation KMM can boost the accuracy to 62.52 %. However, with SHLA one reaches 63.36 %. This improvement has a statistical significance at the $p < 0.001$ level compared with the baselines CT and MINT.

In comparison with ABC, although SUSAS's average UAR from the CT method is still close to chance level, it is worth noting that, the average UAR achieved by the MINT method increases sharply to 62.41 % because of the larger size of SUSAS leading to more examples chosen from the FAU AEC training set. But SUSAS cannot obtain a great benefit from the covariate shift adaptation KMM, like ABC. Nevertheless, the SHLA method still gives an average UAR of 62.72 %, which is slightly larger than the maximum average UAR obtained by the MINT. Compared with the four methods in use, the proposed SHLA method passes the significant t-test at the $p < 0.01$ and $p < 0.02$ level against the CT and KMM methods.

Finally, the intra-corpus scenario is considered, which means that the SHLA method carries out the representation learning between the FAU AEC training set and its test set. In this case, the SHLA obtains an average UAR of 68.29% compared to the baseline (the standard SVM) UAR of 67.04%. The improvement is significant at the $p < 0.05$ level.

Overall, SHLA-based representation learning could be shown as useful in reducing the difference for cross-corpus recognition.

4.4.3 Conclusions

The 'shared-hidden-layer autoencoder' (SHLA) shared across training and target corpora was proposed for feature transfer learning. In this method, the SHLA is used to explore the common feature representation in order to compensate for the differences in corpora caused by language, speaker, and acoustic conditions. Such learned representations were successfully applied to a standard machine learning task: acoustic emotion recognition. Experimental results on three publicly available corpora demonstrate that, the proposed method effectively and significantly enhances the emotion classification accuracy and competes well with other domain adaptation methods.

4.5 A-DAE Feature Transfer Learning

4.5.1 Experiments

This section performs experiments to assess the adaptive denoising autoencoder (A-DAE) feature transfer learning algorithm (Section 3.4.6), where prior knowledge learned from a target set is used to regularize the training on a source set. Its goal is to achieve a matched feature space representation for the target and source sets while ensuring target domain knowledge transfer. Here, the method is evaluated on the FAU AEC as target corpus and two other standard speech emotion corpora as sources.

For the training of the autoencoders, the toolkit minFunc was applied which implements L-BFGS to optimize the parameters of DAEs and A-DAEs. For training of the DAE, masking noise with a variance of 0.01 was injected to generate a corrupted input. For the parameters of the DAE, the weight decay values λ were set to 0.0001, the number of epochs of DAE training was set to 250. In the A-DAE learning process, the hyper-parameter β was fixed to 0.05.

Note that, the FAU AEC official test partition is always used as the 'target' test set, its official training partition is partially used as 'target' training set, and the further two corpora are exclusively used as additional 'source' training sets.

Models for Comparison

The following methods are used to evaluate the proposed approach in the context of the current state-of-the-art: (1) MINT: randomly (repeated ten times) picks a number of examples from the FAU AEC training set to train an SVM, i.e., without the need of transferring to an intra-corpus scenario. For fair comparison, this number is set by the number of training examples of the the ABC or SUSAS sets, respectively. (2) CT: uses ABC or SUSAS to train the standard (SVM) classifier, which is the 'classical' cross-corpus testing, i.e., it involves no adaptation. (3) KLIEP [168], (4) uLSIF [169], and (5) KMM [172]: utilize the modern domain adaption methods on the ABC and SUSAS database for covariate shift adaptation, respectively. (6) DAE: employs denoising autoencoders for representation learning in order to match training examples to test examples; this was successfully applied to the transfer learning challenge and domain adaptation [166, 210].

4.5.2 Experimental Results

For the cross-corpus setting, where acoustic emotion recognition models are trained on ABC or SUSAS while evaluated on the FAU AEC test set (except the MINT condition that uses FAU AEC data for training), the number of hidden units is fixed to 256. Results of the averaged UAR over ten trials are reported in Table 4.7. As can be seen, the A-DAE approach always shows a comparable performance to other approaches [166, 168, 169, 172].

For the small database ABC, the two standard methods (CT and MINT) only yield an average UAR around chance level (55.28 % and 58.32 %). With the benefits of compensation for the existent mismatch, the covariate shift adaptation KMM can achieve the accuracy of 62.52 %. The proposed A-DAE method outperforms all other methods with 64.18 % UAR. This improvement has a statistical significance at $p < 0.001$ with a one-sided z-test when compared to CT and MINT.

On the SUSAS database, the proposed A-DAE method shows a significant improvement over other methods. Specifically, the A-DAE method gives an average UAR of 62.74 %, which is slightly higher than the maximum average UAR obtained by MINT. Moreover, it passes the significance test at $p < 0.001$ and $p < 0.002$ against the CT and KMM methods, respectively. Also, it is worth noting that the average UAR obtained by MINT increases dramatically to 62.42 % just due to the larger size

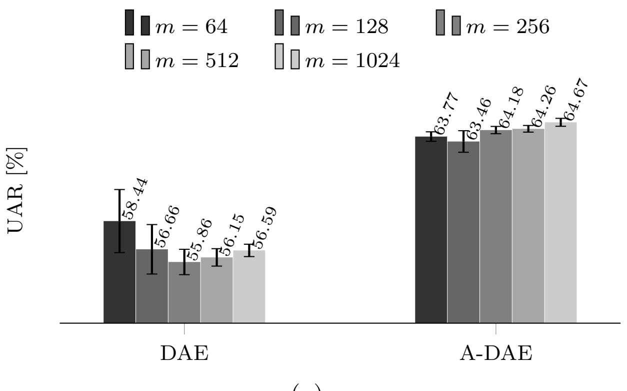
Table 4.7: Average UAR over ten trials: Matched instance number training (MINT), Cross training (CT), covariate shift adaptation methods KLIEP, uLSIF, and KMM, DAE-based representation learning, and the proposed A-DAE method related to training with ABC and SUSAS.

UAR [%]	ABC	SUSAS
MINT	58.32 ± 4.23	62.41 ± 3.85
CT	55.28 ± 0.00	57.32 ± 0.00
KLIEP [168]	55.07 ± 3.81	58.11 ± 3.56
uLSIF [169]	53.75 ± 1.68	57.94 ± 0.60
KMM [172]	62.52 ± 0.00	60.43 ± 0.00
DAE	55.86 ± 0.80	62.03 ± 0.69
A-DAE	$\mathbf{64.18 \pm 0.23}$	$\mathbf{62.74 \pm 0.27}$

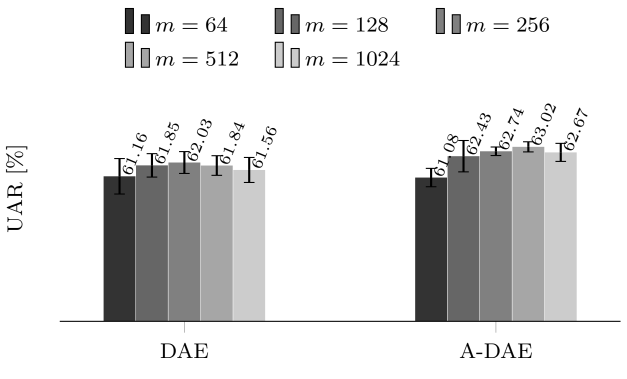
of SUSAS leading to more examples being chosen from the FAU AEC training set in comparison to ABC.

A-DAE vs. DAE

A comparison between the A-DAE and DAE methods is now made in detail because A-DAE has its roots in DAE. Figure 4.8 provides UAR for different numbers of hidden units m, where performance changes for different parameter settings are analyzed. Based on Figure 4.8, it is worth noting that the proposed method obtains the highest UAR of 64.67 % for ABC and of 63.02 % for SUSAS at $m = 1\,024$ and $m = 512$, respectively. Surprisingly, one could not obtain a sustained performance growth with more hidden units for SUSAS. One reason is that the utterances of ABC are more complex and have more variance (length and content) than those of SUSAS which contain pre-defined short commands. Therefore, the increase in hidden units potentially yields more generalization performance for ABC than for SUSAS. In contrast, increasing the number of hidden units to $m = 1024$ in the case of SUSAS reduces the corresponding performance because overfitting occurs. Nevertheless, increasing the number of hidden units leads to additional improvement indeed, which confirms that an over-complete first hidden layer works better than an under-complete one when using unsupervised pre-training as in the theory of deep architectures [249].



(a)



(b)

Figure 4.8: Average UAR with standard deviation over ten trials with varying number of hidden units (m) using DAE or A-DAE.

4.5.3 Conclusions

In this section, the unsupervised domain adaptation method based on adaptive denoising autoencoders is examined by affective speech signal analysis. The method is capable of reducing the discrepancy between training and test sets due to different conditions (e.g., different corpora). A denoising autoencoder is first built on the target domain adaptation set without using any label information with the aim to encode the target data in an optimal way. These encoding parameters are used as prior information to regularize the training process of an A-DAE on the training set. In this way, a trade-off between the reconstruction error on the training data and a knowledge transfer to the target domain is found, effectively reducing the existing mismatch between the training and testing conditions in an unsupervised way. Results with three publicly available corpora show that, the proposed method effectively and significantly enhances the emotion classification accuracy in mismatched training and test conditions when compared to other domain adaptation methods.

4.6 Emotion Recognition Based on Feature Transfer Learning in Subspace

4.6.1 Experiments

The feature transfer learning in subspace method introduced in Section 3.4.8 uses denoising autoencoders to build high-order subspaces of the source and target corpora, where features in the source domain are transferred to the target domain by an additional neural network. This method is referred to as DAE-NN. To exemplify effectiveness of this approach, three common emotional speech corpora, i.e., FAU AEC, ABC, and SUSAS, are selected for extensive and reproducible evaluation.

The DAE-NN method involves the parameter optimization of DAEs and a regression neural network. For training of the DAE, masking noise with a variance of 0.01 was used to generate the corrupted input. The number of hidden units m was fixed to 200 for the two DAEs and the NN, and the weight decay technique with a defined value of 0.0001 was considered. The number of epochs for the DAEs was set to 250 and the number for NNs was decreased to 50.

Models for Comparison

To evaluate this presented approach in the context of the current state-of-the-art, the following methods are considered for comparison:

- MINT: in this reference 'method' we randomly (repeated ten times to reduce singularity effects) pick a number of examples from the FAU AEC official training set to train an SVM, i.e., without the need of transferring in an intra-corpus scenario. For a fair comparison, the number is given by picking the same number of learning examples as is given by the ABC or SUSAS sets, respectively. In other words, this can be considered as baseline reference using exclusively target-type data, but each with the same amount of training examples as will later be used coming from non-target data.

- CT: uses the source corpora ABC or SUSAS to train the standard (SVM) classifier directly, i.e., without using any methods to reduce the mismatch between source and target data. This is the 'classical' cross-corpus testing.

- KMM: utilizes the KMM method (see Section 3.2.1.1) on the ABC and SUSAS database for covariate shift adaptation. It is thus the first reference with application of transfer learning.

- DAE: employs denoising autoencoders for representation learning in order to match training examples to test examples, which was successfully applied to the transfer learning challenge and domain adaptation [166, 210], and may

Table 4.8: Average UAR over ten trials: MINT, CT, covariate shift adaptation KMM, DAE feature transfer representation learning, and the proposed DAE-NN method related to training with ABC and SUSAS.

UAR [%]	MINT	CT	KMM	DAE	DAE-NN
ABC	58.32	55.28	62.52	56.20	**63.63**
SUSAS	62.41	57.32	60.41	62.08	**64.73**

be considered as close reference from a method point of view, as a DAE is also used, yet, without the linking between source and target domain during transferring as is proposed in the DAE-NN method.

- DAE-NN: finally uses the DAE-NN method to compensate for the mismatch between the features on the training and test sets, then trains standard SVMs using the compensated features and labels in the training set.

4.6.2 Experimental Results

In the case of a cross-corpus scenario, emotion recognition engines are trained on ABC or SUSAS while tested on the FAU AEC test set. The experiments are run ten times for each training set, and we evaluate the performance by UAR. When using ABC or SUSAS, averaged UAR over the ten trials is visualised in Figure 4.9, including the error bars, and given quantitatively in Table 4.8 for reference comparison. As can been seen, the DAE-NN method always achieves larger average UAR for ABC and SUSAS when compared to the MINT and CT cases. It also exceeds the UAR achieved by the DAE method and the covariate shift adaptation KMM.

More specifically, for the small database ABC (composed of only 430 examples), one can easily see that, the standard method (CT) only obtains an average UAR around the chance level (55.28 %) due to the inherent mismatch between the ABC used for training and the FAU AEC test set. The accuracy obtained by the DAE reaches 56.20 %, the covariate shift adaptation KMM can boost the accuracy to 62.52 %. However, with DAE-NN one reaches 63.63 %, which yields 1.11 % absolute improvement when compared to KMM. This improvement has a high statistical significance at the $p < 0.001$ level compared with the baseline CT and even the one of MINT, i. e., even when using 430 target domain examples.

In comparison with ABC, SUSAS's average UAR in 'classical do-nothing' cross-corpus testing (CT method) is also close to chance level. Here, however, it is worth noting that, the average UAR achieved by training with an equivalent number of target domain examples as found in SUSAS (i. e., 3.6 k examples, MINT method) increases sharply to 62.41 % because of the eight times larger size of SUSAS than

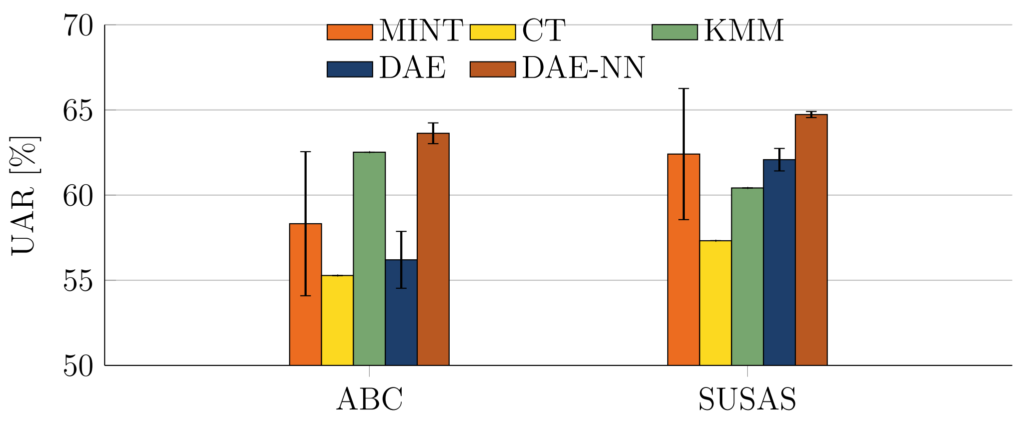


Figure 4.9: Cross-corpus average UAR over ten trials using MINT, CT, the covariate shift adaptation KMM, the DAE feature transfer learning, and the feature transfer learning DAE-NN for ABC and SUSAS.

ABC leading to eight times more examples chosen from the FAU AEC training set. Unlike ABC, SUSAS cannot obtain a great benefit from the covariate shift adaptation KMM but can achieve a comparable performance by DAE. Again, the DAE-NN method gives the highest average UAR of 64.73 %, which is again surprisingly even exceeding the average UAR obtained by the MINT 'method'. Compared with all four reference methods, the superiority of the proposed DAE-NN method passes the significant test at the $p < 0.001$ level.

It is worth noting that, the UAR from the DAE-NN for the SUSAS database is slightly larger than the one for the ABC. Thus, it is believed that this can partially be attributed to the larger size of the SUSAS corpus. Overall, the DAE-NN-based feature transfer learning could be shown as highly useful in reducing the difference for cross-corpus recognition.

4.6.3 Conclusions

A feature transfer learning method, referred to as DAE-NN, was proposed to address a situation where training and test set come from different corpora. The method uses denoising autoencoders to build a subspace for the source domain and the target domain, and makes use of regression neural networks in order to reduce the mismatch between target data and source data on subspace feature level. The proposed method was successfully applied to the speech emotion recognition task. Experimental results with three publicly available corpora demonstrate that, the presented method remarkably improves the emotion classification accuracy and competes well with other domain adaptation methods.

4.7 Recognizing Whispered Emotions by Feature Transfer Learning

4.7.1 Experiments

Whispered speech, as an alternative speaking style for 'normal phonated' (non-whispered) speech, has so far received little attention in speech emotion recognition. Currently, speech emotion recognition systems are exclusively designed to process 'normal' phonated speech, and can result in significantly degraded performance on whispered speech because of the fundamental differences between these two – normal phonated speech and whispered speech – in vocal excitation and the vocal tract transfer function. This section, motivated by the recent successes of feature transfer learning, sheds some light on this topic by proposing three feature transfer learning methods based on denoising autoencoders (Section 3.4.4), shared-hidden-layer autoencoders (Section 3.4.5), and extreme learning machines autoencoders (Section 3.4.7). Without the availability of labeled whispered speech data in the training phase, in turn, the three proposed methods can help modern emotion recognition models trained on normal phonated speech to reliably handle also whispered speech. Throughout extensive experiments on the GEWEC data and the EMO-DB data, the three methods are compared to alternative methods reported to perform well for a wide range of speech emotion recognition tasks and find that the proposed methods provide significant superior performance on both, normal phonated *and* whispered speech.

Whispered Speech Emotion Recognition

SER has grown into a major research topic in speech processing, human-computer interaction, and computer-mediated human communication over the last decades (see [7, 8, 9, 10]). In general, it focuses on using machine learning methods to automatically predict 'correct' emotional states from speech. Apart from normal phonated speech for which current studies mainly have made considerable efforts to date, *whispered speech* is another common form of speaking to communicate; it is produced by speaking with high breathiness and no periodic excitation. With the absence of periodic vibration of the vocal folds during the production, whispered speech's structure is significantly altered which results in reduced perceptibly and a significant reduction in intelligibility. In the meantime, it was already found that whispered speech can encode prosodic information, and thereby still convey clues carrying emotion information [250, 251]. Naturally, whispered speech plays an important role in our daily life in order to intentionally confine the hearing of speech to listeners who are nearby. For example, people whisper to the user interface over the cellphone when offering privacy information in terms of date of birth, credit card information, billing address to make hotel, flight, and table reservations. Further, one may at times wish not to disturb others around as when preferring to whisper information. Another area of interest are patients with speech disabilities who are affected by a temporary or long-term limitation in the vocal fold structure or disease of the vocal system such as functional aphonia or laryngeal disorders [252] and therefore can only produce whisper-like sounds. For SER, however, only a handful of efforts have been devoted to recognizing whispered speech by now [253, 254]. Especially, the issue of how to build a practically feasible emotion recognition system tailored for whispered speech has not been addressed, yet, as past work mainly analyzed the differences of the prosodic features in emotions of Chinese whispered speech [253, 254]. Hence, to be more useful in practice, it would be highly desirable to enable an emotion recognition system to process whispered speech as well with promising accuracy.

In the speech community, there has been a considerable amount of related work on whispered speech [252, 255, 256, 257, 258, 259, 260]. In [259], the authors addressed F0 modeling in whisper-to-audible speech conversion and then proposed a hybrid unit selection approach for whisper-to-speech conversion based on the finding that F0 contours can be derived from the mapped spectral vectors. Furthermore, to improve the intelligibility of whispered speech in various noise contexts, an unsupervised learning of phonemes was proposed based on convolutive non-negative matrix factorization [257]. For the task of acoustic voice analysis in computer laryngeal diagnostics, the authors in [252] developed three methods for fundamental frequency determination of the voice of patients with laryngeal disorders, including auto-correlation, spectral, and cepstral methods. Moreover, these methods were combined in a system for acoustic analysis and screening of the pathological voices in the everyday clinical practice.

Collecting naturalistic real-life emotional speech is always challenging mainly due to privacy reasons. Obviously, this factor is amplified by an order of magnitude in the case of whispered emotional speech. Luckily, rather than tediously collecting and labeling whispered speech and designing a dedicated system from scratch, past studies also have shown that a workable scheme in an attempt to deal with whispered speech is to explore normal phonated speech data to create and develop systems that would be much more robust against variability and shifts in speech modes (e.g., normal phonated and whispered modes) [255, 261]. For example, the authors in [255] recently considered a feature transformation estimation method in the training phase which results in a more robust speaker model for speaker identification on whispered speech. Three estimation methods are proposed to model the transformation from normal phonated speech to whispered speech. This solution seems also reasonably feasible and worthwhile in SER, because it allows for one single recognition system to process both, normal phonated and whispered speech simultaneously. Another important reason is that massively available normal phonated speech is a potential benefit of the recognition system in the era of big data considering the alluded to scarceness of real-life whispered emotional data. For these reasons, this strategy, i.e.,

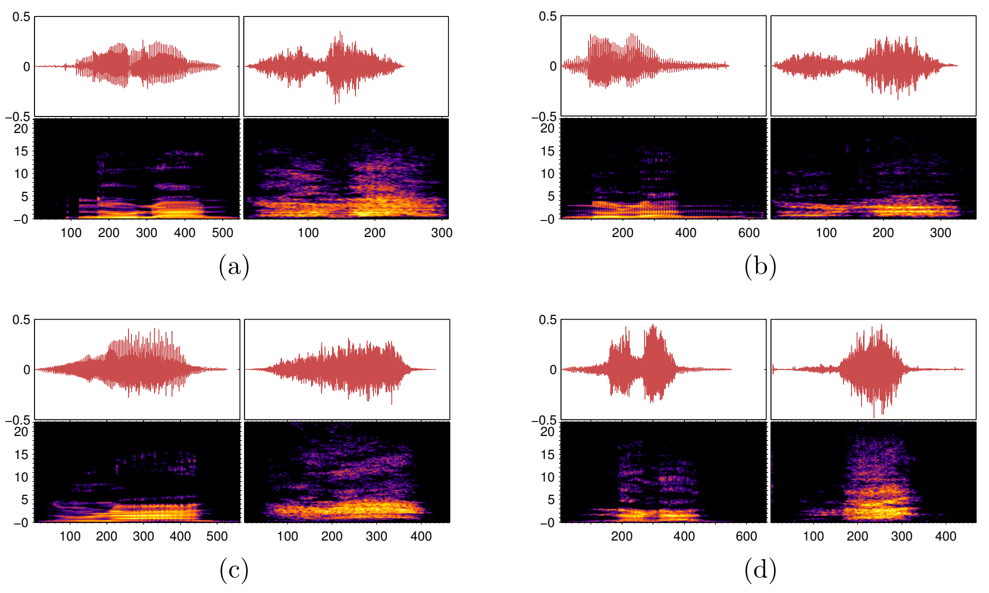


Figure 4.10: Examples of waveforms (top row) and spectrograms (bottom row) for the same speech signal “nolan” expressed in two emotional states *neutral* and *anger* by different genders in normal (left column) and whispered speech (right column) styles taken from GEWEC. (a) *Neutral* emotion for the female speaker. (b) *Neutral* emotion for the male speaker. (c) *Anger* emotion for the same female speaker. (d) *Anger* emotion for the same male speaker.

deploying normal phonated speech data for whispered speech-based tasks, is adopted in this study for creating a whispered speech emotion recognition system.

A major concern of a whispered speech emotion recognition system is as follows: normal phonated speech fundamentally differs from whispered speech in their use of the spectrum both perceptually and for speech production. Figure 4.10 visualizes the differences for normal phonated speech and whispered speech expressed in two given emotional states. Specifically, the absence of periodic vibration of the vocal folds during production of whispered speech leads to a lack of 1) voiced excitation, 2) harmonic structure, 3) acoustic cues signaling the fundamental frequency (F0), 4) shifted formant locations, and 5) changes in formant band width (see [255, 262, 263, 264, 265, 266]). Speech emotion systems built with ‘normal’ phonated speech signals are thus challenged, and can deliver significantly degraded performance, when they encounter whispered speech that accordingly differs from the limited conditions under which they were originally developed and ‘trained’. Hence, such differences between the test data and training data render whispered speech emotion recognition a challenging task.

There has been a considerable amount of related work to overcome the problem of training/test feature distribution mismatch in the field of SER [267, 268] in general.

For example, an iterative feature normalization scheme designed to reduce the speaker variability, while preserving the signal information critical to discriminate between emotional states, is proposed in [267]. Furthermore, the authors in [268] analyzed how speaker variability affects the feature distribution in detail, and further presented a speaker normalization approach based on joint factor analysis to compensate for some of the effects identified.

A generic approach for reducing the mismatch problem in SER is based on IW (Section 3.2.1). The essential idea is to assign more weight to those training examples that are most similar to the test data, and less weight to those that poorly reflect the distribution of the test data. With this idea on mind, the authors in [168] proposed unconstrained least-squares importance fitting (uLSIF) to estimate the importance weights by a linear model. Additionally, one can model the importance function by a linear (or kernel) model, which leads to a convex optimization problem with a sparse solution, called the Kullback-Leibler importance estimation procedure, or KLIEP [169]. Kernel mean matching (KMM) was proposed to directly estimate the resampling weights by matching training and test distribution feature means in a reproducing kernel Hilbert space [172]. The three methods have recently been shown to lead to significant improvement in SER when the authors in [170] first considered to explicitly compensate for acoustic and speaker differences between training and test databases.

Another possible solution to address the problem of these differences is to deploy *feature learning* (or representation learning). Feature learning, i.e., learning suited transformations of the data that render it easier to extract salient information when building classifiers or other predictors, has been considered from many perspectives within the realm of machine learning [146, 165, 206, 207]. The key idea of feature learning is to employ deep architectures, resulting in an abstract representation. Generally, more abstract concepts are invariant to most local changes of the input. Following the concept of feature learning, *feature transfer learning* has been proposed to deal with the problem of how to reuse the knowledge learned previously from 'other' data or features [16]. This rather essential characteristic suggests that feature transfer learning would be well suited for the scenarios where the data distribution in the test domain is different from the one in the training domain but the task remains the same [165, 206]. For example, feature transfer learning based on a sparse autoencoder for discovering knowledge in acoustic features from small labeled target data to improve performance of SER when applying the knowledge to source data was proposed in [165].

Motivated by feature transfer learning, the present section will demonstrate that such a concept considerably benefits an emotion recognition system for whispered speech when it uses normal phonated speech data for training as well. Specifically, this work is centered around the idea of a transformation from the normal phonated speech domain to the whispered speech domain without the need of labels for the whispered data. The resulting transformation can alleviate the disparity between the two

Table 4.9: Overview of the selected two databases for whispered emotion recognition.

[#]	Emotion[a]				Valence[b]		Arousal[c]	
	A	F	H	N	−	+	−	+
GEWEC:								
Normal	160	160	160	160	320	320	160	480
Whispered	160	160	160	160	320	320	160	480
EMO-DB:								
Normal	127	55	64	78	182	142	78	246

[a] Emotion categories: anger (A), fear (F), happiness (H), and neutral (N).
[b] Binary valence: negative (−), positive (+).
[c] Binary arousal: low (−), high (+).

datasets and then support effective supervised learning in building a whispered speech emotion recognition system. Accordingly, the focus of the present work is placed on exploring standard, but powerful feature transfer learning techniques based on autoencoders including *denoising autoencoders* (DAE) [197] (cf. Section 3.4.4) , a more recent variant, i.e., *shared-hidden-layer autoencoders* (SHLA) [206] (cf. Section 3.4.5), and *extreme learning machine autoencoders* (ELM-AE) [223] (cf. Section 3.4.7). As a result, the proposed feature transfer learning methods successfully endow a speech emotion model that can adapt to a range of speech modalities, including in particular normal phonated speech and whispered speech.

Experimental Setup

The GEWEC and EMO-DB databases are chosen for evaluation, which are shown in Table 4.9. In order to run experiments based on a common set of emotional states, only those emotional states in EMO-DB appearing in the GEWEC data are retained for experiments. In this way, EMO-DB here ends up consisting of 322 utterances as shown in Table 4.9.

For optimization of the parameters in the autoencoders such as DAE and SHLA, we applied the third party software minFunc implementing L-BFGS gradient descent. In the experiments, attempted hyper-parameters for DAE and SHLA are the following: the maximum iteration number $iter_{max} \in \{20, 40, 50, 100, \ldots, 300\}$, the number of hidden units $m \in \{64, 128, \ldots, 1\,024\}$, the weight decay value $\lambda_{tr}(\lambda_{te}) \in \{10^{-3}, 10^{-2}, 10^{-1}\}$, and the hyper-parameter for SHLA $\gamma \in \{0, 0.1, \ldots, 1\}$. In addition, masking noise with a variance of 0.01 is injected to inputs during the training of the DAE and the SHLA. For the ELM-AE, the number of hidden units $m \in \{64, 128, \ldots, 1\,024, 2\,000, \ldots, 7\,000\}$, and the regularization term

$C \in \{10^{-5}, 10^{-4}, \ldots, 10^{8}\}$ are attempted.

4.7.2 Experimental Results

Acoustic Feature Analysis

Feature selection on GEWEC is now considered to reveal which features derived from the two different speech modes are important for the task of interest. By means of one feature selection algorithm for ranking using the information gain with respect to the class implemented in the WEKA toolkit [101], we compare the features obtained on the normal phonated speech with those obtained on the whispered speech from the GEWEC data in Figure 4.11.

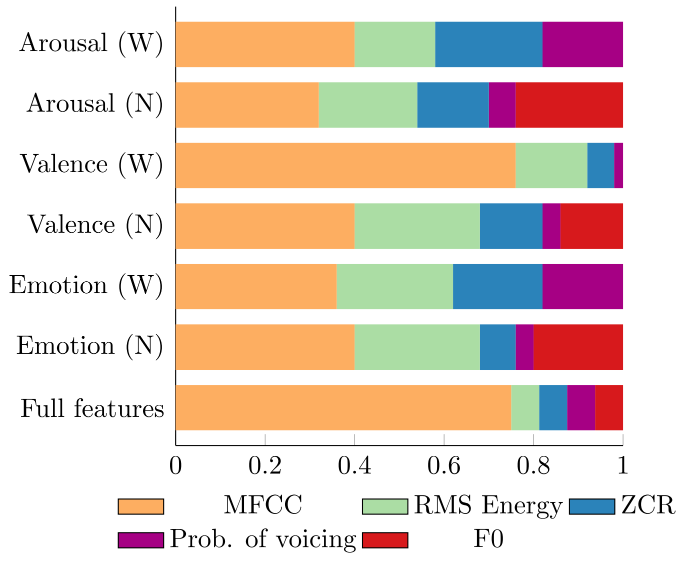
For all tasks, it can be observed that the relative importance of LLDs remarkably differs between the two speech modes. For instance, the F0-related features appear crucial for normal phonated speech while they appear entirely irrelevant for whispered speech, which is expected due to the absence of the fundamental frequency in whispered speech. Besides, the probability of voicing and ZCR for whispered speech show increased relevance in the emotion and arousal cases. One possible reason causing such change is that for whispered speech, discrimination performance is mainly affected by the high-frequency region whereas for normal phonated speech, discrimination performance is mainly affected by the low-frequency region [266].

As for the types of functionals, it can be observed that, the relative importance of means and moments increases for whispered speech when compared to normal phonated speech, whereas the relevance of regression coefficients decreases. This can be explained by the fact that the means and moments are more robust to extract, whereas computing reliable regression coefficients may be more difficult in a more noisy setting as is given for most LLDs in case of whispered speech.

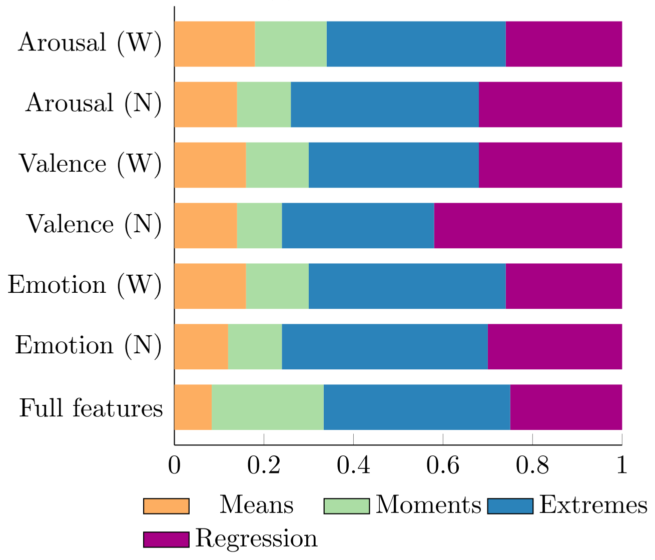
Overall, the acoustic feature analysis shows that relevant features for use in a speech emotion recognition model construction are different from normal phonated speech and whispered speech, and using normal phonated speech as training set to recognize emotional states from whispered speech can be assumed as challenging.

Benchmark Tests on GEWEC

A number of experiments are first run where the training and the test set varies in the combinations of normal phonated speech and whispered speech within the data GEWEC. These include matched and mismatched as well as multi-condition training and testing. Table 4.10 lists all nine different training and test set combinations. Apart from emotion categories, the discrimination between binary valence and the discrimination between binary arousal is also evaluated. A practical and challenging leave-one-speaker-out cross validation strategy is used to meet speaker independent criteria.



(a) LLDs



(b) Functionals

Figure 4.11: Full INTERSPEECH 09 feature set (Full, bottom) vs. 50 best features selected by measuring the information gain with respect to the class on whispered (W) speech and normal phonated speech (N) in the GEWEC data for binary arousal and valence, and quaternary emotion classification. The percentage of the selected low-level descriptors (LLDs) and types of functionals is shown.

As may be expected and can be seen from Table 4.10, the recognition system using supra-segmental features works best when both the training and test data are entirely drawn from normal phonated speech, leading to the highest UAR of 74.1 %

Table 4.10: Recognition results for emotion categories and binary valence and binary arousal in leave-one-speaker-out testing for different train/test combinations.

UAR [%]	Train on		
Test on	Normal	Whispered	Both
Emotion:			
Normal	74.1	41.7	58.3
Whispered	44.5	46.7	50.6
Both	59.3	44.2	59.5
Valence:			
Normal	73.1	61.7	70.5
Whispered	57.2	57.6	59.4
Both	65.2	59.0	64.9
Arousal:			
Normal	58.6	60.1	61.9
Whispered	62.2	57.6	59.4
Both	60.4	58.9	60.6

for the four-class emotion classification problem. Further—also as one may expect—, whispered speech (in matched condition) reaches a significantly lower UAR of 46.7 %. Using whispered speech for training seems to downgrade in particular the recognition of valence. It seems plausible that a training set drawn from whispered speech should be a better way for whispered speech emotion recognition (i.e., matched condition learning). However, there is no significant reduction in the system using normal phonated speech based on Table 4.10. For binary valence and binary arousal, it is even surprisingly observed that, the system trained with normal phonated speech sometimes obtains slightly higher UAR than the ones when trained with whispered speech when testing on whispered speech. Further, a multi-condition training is only truly beneficial for whispered speech.

Results on GEWEC

Here the results obtained by the emotion recognition system using the proposed and further domain adaptation methods are reported.

A basic model without any adaptation and the three 'IW' methods are used for comparison, listed as follows, to evaluate the three feature transfer learning approaches:

Table 4.11: Average UAR over ten trials on GEWEC: no Transfer Learning ('none'), and the methods KLIEP, uLSIF, and KMM, and the feature transfer learning methods DAE, SHLA, and ELM-AE. Significant results (p-value < 0.05, one-sided z-test) are marked with an asterisk, judged relative to 'none'. The best UAR is highlighted in bold. Speaker-independent classification by SVM.

GEWEC	Whispered (test), Normal (train)		
UAR [%]	Emotion	Valence	Arousal
None	45.3 ± 0.0	63.0 ± 0.0	65.1 ± 0.0
KLIEP [168]	46.1 ± 0.6	63.8 ± 0.4	60.9 ± 0.9
uLSIF [169]	45.1 ± 0.4	63.0 ± 0.5	64.7 ± 0.5
KMM [172]	47.8 ± 0.0	62.8 ± 0.0	65.0 ± 0.0
DAE	$*53.7 \pm 1.6$	63.6 ± 1.1	67.2 ± 2.1
SHLA	$*\mathbf{54.5 \pm 1.6}$	63.5 ± 1.2	$*70.6 \pm 2.9$
ELM-AE	$*52.3 \pm 0.4$	$\mathbf{65.0 \pm 0.9}$	$*\mathbf{74.6 \pm 1.0}$

Table 4.12: Average UAR over ten trials on GEWEC: no Transfer Learning ('none'), and the methods KLIEP, uLSIF, and KMM, and the feature transfer learning methods DAE, SHLA, and ELM-AE.

GEWEC	Normal (test), Whispered (train)		
UAR [%]	Emotion	Valence	Arousal
None	52.2 ± 0.0	62.8 ± 0.0	69.0 ± 0.0
KLIEP [168]	56.7 ± 0.6	62.6 ± 0.6	65.4 ± 1.4
uLSIF [169]	49.2 ± 0.2	62.9 ± 0.4	67.1 ± 0.3
KMM [172]	55.8 ± 0.0	66.6 ± 0.0	72.7 ± 0.0
DAE	53.1 ± 1.7	66.9 ± 1.8	$*76.4 \pm 3.9$
SHLA	$*58.3 \pm 1.8$	66.0 ± 1.9	$*81.1 \pm 5.2$
ELM-AE	$*\mathbf{63.7 \pm 0.7}$	$*\mathbf{72.9 \pm 0.7}$	$*\mathbf{85.6 \pm 0.2}$

- 'None': employs a conventional speech emotion system, i.e., involving no adaptation, to predict emotions for a given whispered utterance.

- KLIEP [168], uLSIF [169], and KMM [172]: utilizes these modern domain adaptation methods for covariate shift adaptation, respectively before SVM classification.

First, similar to a cross-corpus setting, speech emotion recognition models are trained on normal phonated speech while tested on whispered speech. Because of the random initialization in the autoencoders and the IW methods, the results of the averaged UAR over ten trials, along with a significance level computed by a one-sided z-test, are given in Table 4.11. It is found that, the DAE, SHLA, and ELM-AE outperform all the other approaches. In detail, the best performing methods for the three tasks, which achieve UARs of 54.5 %, 65.0 %, and 74.6 %, respectively, use autoencoders. For all the three tasks, the IW methods just achieve similar results as in the 'None' condition. On two of the three tasks, however, the autoencoder-based methods exhibit a statistically significant improvement over the 'None' condition. Note that, the SHLA improves on the DAE, showing that it can leverage information both from the training set and the test set in a more effective way. However, the ELM-AE generally outperforms the SHLA, which may indicate that the ELM-AE tends to attain more generalization performance.

To further test the effectiveness of the feature transfer learning methods at reducing the mismatch problem, more experiments for recognizing emotions from normal phonated speech are considered, specifically in which the training data is whispered speech, and the test data is normal phonated speech. Table 4.12 summarizes these results. It shows that, the feature transfer learning methods consistently outperform all the other methods since they achieve the highest UARs for the three tasks as well. In other words, they are also found effective for normal phonated speech emotion recognition systems when a mismatch to the training data is given.

Results on EMO-DB

Although the transfer-learning system was originally intended for coping with whispered speech, one would be curious to see if such an approach can be suitable also for normal phonated speech in a cross-corpus setting since normal phonated speech is a more common way in our daily life. Therefore, the feature transfer learning methods are further tested on normal phonated speech. In doing so, the recognition models obtained by the feature transfer learning methods as well as the other concurrent methods for comparison, which are originally developed for whispered speech in Section 4.7.2, are now evaluated to make predictions on the data of the EMO-DB corpus. Note that, these models are trained in a way of transferring the knowledge from whispered speech to normal speech within GEWEC. Following the experimental settings, these results are presented in Table 4.13.

It is found that, the feature transfer learning methods can retain a competing performance as in the 'None' condition (i.e., a usual cross-corpus setting training on one corpus and testing on another), where the training and test data come from normal phonated speech, whereas all of the IW methods lead to a significant reduction in performance. In addition, for the emotion and arousal tasks, the feature transfer

Table 4.13: Average UAR on EMO-DB: All the models are originally trained to transfer the knowledge from whispered speech to normal speech within GEWEC while directly testing on EMO-DB.

UAR [%]	Emotion	Valence	Arousal
None	49.9 ± 0.0	$\mathbf{84.2 \pm 0.0}$	75.0 ± 0.0
KLIEP [168]	17.1 ± 0.9	66.4 ± 0.6	34.9 ± 0.7
uLSIF [169]	19.7 ± 0.6	67.8 ± 0.3	34.1 ± 0.3
KMM [172]	15.8 ± 0.0	68.2 ± 0.0	27.4 ± 0.0
DAE	54.5 ± 1.5	72.2 ± 1.9	78.6 ± 3.9
SHLA	55.4 ± 1.6	69.5 ± 1.8	$*82.1 \pm 1.8$
ELM-AE	$*\mathbf{57.4 \pm 0.4}$	76.0 ± 0.3	$*\mathbf{85.5 \pm 1.0}$

Table 4.14: Comparison of running time (s) of the DAE, SHLA, and the ELM-AE on GEWEC.

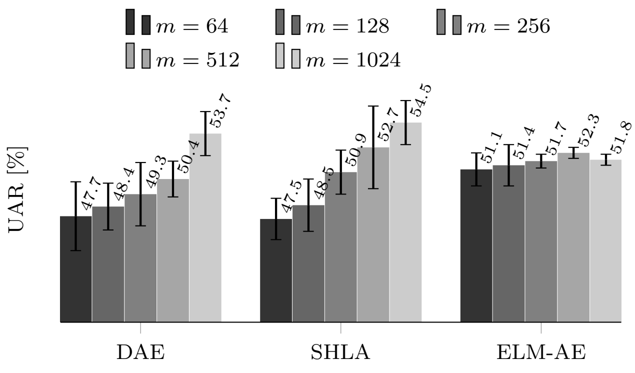
Hidden units	DAE	SHLA	ELM-AE
64	4.734 ± 0.500	11.348 ± 2.091	0.020 ± 0.005
128	9.098 ± 2.808	15.799 ± 1.543	0.035 ± 0.005
256	14.023 ± 2.904	27.159 ± 3.885	0.086 ± 0.0569
512	23.062 ± 4.828	42.896 ± 8.322	0.154 ± 0.029
1 024	35.952 ± 11.126	61.502 ± 14.604	0.337 ± 0.040

learning methods significantly improve the performance in UAR over the ‘None’ condition, which may indicate that the knowledge of whispered speech automatically found by the proposed methods might be beneficial for normal phonated speech recognition to some degree. Overall, these findings may suggest that the autoencoder-based methods have great advantages to generate feature representations which are common to or invariant across both whispered and normal phonated speech.

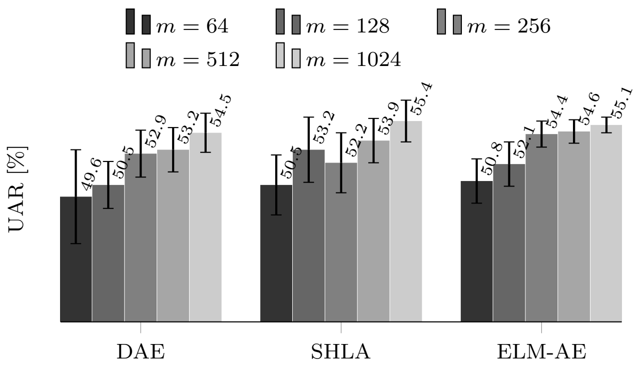
Comparison between the Autoencoder-based Methods

Furthermore, the running time of DAE, SHLA, and ELM-AE on GEWEC are compared. As can be seen from Table 4.14, the ELM-AE has the least amount of needed running time with respect to the DAE and SHLA, simply because its training phase avoids tuning the parameters iteratively.

Another set experiments are next conducted to take a closer look at the differences between the autoencoder methods. Figure 4.12 demonstrates how the number of hidden units m influences the performance of the different autoencoder-based methods on GEWEC and EMO-DB. It can been seen that, the change in the number of hidden units – within a particular range – has a strong influence on the proposed methods. That is, it is possible to obtain a sustained performance growth with more hidden units.



(a)



(b)

Figure 4.12: Average UAR with standard deviation over ten trials obtained by DAE, SHLA, and ELM-AE with changes in the number of hidden units (m) for the emotion labeling scheme. (a) On GEWEC. (b) On EMO-DB.

4.7.3 Conclusions

Autoencoder-based feature transfer learning has been successfully applied to SER primarily for cross-corpus classification of emotions (see Sections 4.3 to 4.6), rather than whispered speech classification. This section extended these works by 1) showing how autoencoder-based feature transfer learning can be applied to create a recognition engine with a completely trainable architecture that can adapt itself to other speech modalities, such as normal phonated speech and whispered speech, and 2) by considering novel autoencoder realizations by ELM.

To reach the goal of this work, i.e., developing an emotion recognition system which is trained on normal phonated speech that can offer reliable performance also for whispered emotional speech, this section applied three feature transfer learning methods using denoising autoencoders, shared-hidden-layer autoencoders, and extreme learning machines autoencoders for whispered speech emotion recognition. The results demonstrate that such feature transfer learning methods can significantly enhance the prediction accuracy on a range of emotion tasks and are able to outperform alternative methods. At the same time, the proposed methods do not hurt system performance on normal phonated speech.

It was further found that, autoencoder-based feature transfer learning not only can reduce the mismatch between the training set and test set by discovering common features across multiple modes or different corpora, which has been repeatedly shown in previous work like [166, 192], but also can greatly improve the learning performance of a target task by transferring useful information in one source task to the target task in an unsupervised way. Note that, here, whispered speech as the source obviously offers helpful information so as to improve the target task of normal phonated speech emotion recognition. Such benefit has been constantly demonstrated in other transfer learning methods and been widely applied in a number of applications such as web-document classification [269] and WiFi-based indoor localization [270], but has never been found for autoencoder-based feature transfer learning before. Hence, this work provides a new insight into autoencoder-based feature transfer learning.

By that, the results are very encouraging beyond the original intentions: Not only could it be possible to implement a speech emotion recognition engine robust to whispering, but it could also be possible to exploit whispered speech to improve the recognition of normal speech. In a similar fashion, one can now imagine to use all sorts of atypical and inhomogenous databases of emotional speech to train on the various corpora available an engine that is showing better performance simply by having been trained on ‘much more data’ that has, however, been transferred in a meaningful way to a target domain.

5 Summary and Outlook

This chapter summarizes the results obtained and concludes this thesis (Section 5.1). Further, it sketches possible future directions of research (Section 5.2).

5.1 Summary

This thesis was strongly oriented to the practical problem of the distribution mismatch within emotional speech corpora (cf. Sections 1.1 and 3.1). To approach this problem, this thesis advanced the state-of-the-art in transfer learning with autoencoders (cf. Section 3.4). To be more specific, six novel feature transfer learning methods based on autoencoders were proposed to find common features across different data domains in this thesis. Every method was systematically evaluated on a wide range of emotional speech corpora, covering the distribution mismatches caused by the cross-language setting, the cross-speaker setting, the cross-age setting, and the cross-speech-mode setting. Although all methods were evaluated for *Speech Emotion Recognition* (SER) tasks, they are general methods, which are applicable to any task beyond SER.

The first proposed method, incorporating sparse autoencoders in feature transfer learning (Section 3.4.3.1), was found dramatically efficient for transferring knowledge in the real-life situation where a small amount of labeled target domain data are available and rich data from other domains are at hand. This proposed feature transfer learning method was systematically evaluated on six databases for SER tasks (Section 4.3).

Next, this thesis was fully dedicated to even more challenging unsupervised transfer learning. For unsupervised transfer learning, there is no labeled target domain data. In this case, this thesis promoted autoencoders for unsupervised transfer learning with the inspiration of feature learning [146], multitask learning [156], extreme learning machine [221], as well as subspace learning [227]. As a consequence, five unsupervised feature transfer learning methods using denoising autoencoders and the variants were proposed in Sections 3.4.4 to 3.4.8 and extensively evaluated in

Chapter 4, respectively. In particular, these methods include denoising autoencoder feature transfer learning (Section 3.4.4), shared-hidden-layer autoencoder feature transfer learning (Section 3.4.5), adaptive denoising autoencoder feature transfer learning (Section 3.4.6), extreme learning machine autoencoder feature transfer learning (Section 3.4.7), and feature transfer learning in subspace (Section 3.4.8). Experimental results showed that these methods could achieve significant improvement in comparison with the state-of-the-art transfer learning methods.

Finally, the focus of the thesis was placed on the application of unsupervised feature transfer learning for whispered speech emotion recognition (Section 4.7), which has drawn little attention in the filed of SER so far. Three of feature transfer learning methods introduced in Section 3.4, namely feature transfer learning using denoising autoencoders, shared-hidden-layer autoencoders, and extreme learning machine autoencoders, were further extended to develop a whispered speech emotion recognition system. Such a system was evaluated on two public databases containing normal phonated speech and whispered speech. The results indicate that these feature transfer learning algorithms are promising and efficient methods to make the recognition system adapt rapidly to different speech modalities, and outperform alternative transfer learning methods (e.g., kernel mean matching [171] and unconstrained least-squares importance fitting [168]) by a large margin.

5.2 Outlook

This thesis has proposed a set of autoencoder-based feature transfer learning algorithms for automatically exploring abstract representations to reduce the distribution mismatch. Despite their significant achievement for a wide range of novel and real-life SER problems, there are several ideas along this line of research worth being investigated in future.

It is natural to believe that deep architectures, for example, deep neural networks [102] and deep autoencoders [166], are able to extract complex structure and build internal representation from rich inputs. Thus, one potential direction is to expand the shallow architecture of the proposed algorithms to a deep architecture. Furthermore, this thesis has only made use of static modeling (i.e., SVMs), which does not take account of the fact that human emotion is slowly varying and highly context-dependent, therefore, another line of research would improve performance by using long-term context modeling, such as deep LSTM neural networks, to exploit contextual information.

With the popularity of the Internet, the amount of data will constantly increase, and the diversity of data will be significantly enhanced. While a large volume of data may maximize benefits achievable using feature transfer learning methods for a pattern recognition engine, they also pose problems: There are *dirty data* with incompleteness, misspellings, differential precision, and obsolete information. Besides automatic learning of features for transfer learning, these problems give rise to the necessity of data selection techniques, such as cooperative learning [163] and active learning [55], since the accurate predictions of a recognition engine heavily depend on good quality data. Thus, the next step is to integrate autoencoder-based feature transfer learning in such data selection techniques for building good systems, which can identify and clean dirty data.

This thesis has shed light on whispered speech emotion recognition, and has investigated the use of enacted data for the experiments, which is known to lead to overestimated performance. In the future, spontaneous (whispered) data should be considered, which will make recognition systems even more applicable in real-file settings. Obviously, however, collecting spontaneous whispered emotion large quantities will remain quite a challenge.

Overall, all developed algorithms in this thesis are aligned to reach the ultimate goal, i.e., to reduce the distribution mismatch. Hopefully, the research presented here can inspire others to pursue more projects on both algorithmic development and real-life application towards building a universal analysis system like Schuller et al. [164] envision in the iHEARu project.

Acronyms

ABC	Aircraft Behaviour Corpus
A-DAE	Adaptive Denoising Autoencoder
FAU AEC	FAU Aibo Emotion Corpus
AM-FM	Amplitude Modulation-Frequency Modulation
ANN	Artificial Neural Network
ASR	Automatic Speech Recognition
AVIC	Audio-Visual Interest Corpus
BP	Backpropagation
CMN	Cepstral Mean Normalization
CNN	Convolutional Neural Network
CT	Cross Training
DAE	Denoising Autoencoder
DCT	Discrete Cosine Transformation
DNN	Deep Neural Network
ELM	Extreme Learning Machine
ELM-AE	Extreme Learning Machine Autoencoder
EM	Expectation Maximization
EMO-DB	Berlin Emotional Speech Database

FFNN	Feedforward Neural Network
GEMEP	Geneva Multimodal Emotional Portrayals
GEWEC	Geneva Whispered Emotion Corpus
GMM	Gaussian Mixture Model
GPU	Graphics Processing Unit
HMM	Hidden Markov Model
IW	Importance Weighting
***k*-NN**	*k*-Nearest Neighbors
KL	Kullback-Leibler
KLIEP	Kullback-Leibler Importance Estimation Procedure
KMM	Kernel Mean Matching
L-BFGS	Limited-memory Broyden-Fletcher-Goldfarb-Shanno
LLD	Low Level Descriptor
LOI	Level of Interest
LP	Linear Prediction
LPC	Linear Prediction Coding
LPC	Linear Prediction Coding
LPCC	Linear Prediction Cepstral Coefficient
LSTM	Long Short-Term Memory
MFCC	Mel-Frequency Cepstral Coefficient
MINT	Matched Instance Number Training
MLLR	Maximum Likelihood Linear Regression
MLP	Multilayer Perceptron
MNIST	Mixed National Institute of Standards and Technology
MSE	Mean Square Error

MV	Majority Voting
NLL	Negative Log-Likelihood
NN	Neural Network
PCA	Principal Component Analysis
PLP	Perceptual Linear Prediction
RBM	Restricted Boltzmann Machine
ReLU	Rectified Linear Unit
RMS	Root Mean Square
RNN	Recurrent Neural Network
SAE	Sparse Autoencoder
SD	Standard Deviation
SER	Speech Emotion Recognition
SGD	Stochastic Gradient Descent
SHLA	Shared-hidden-layer Autoencoder
SMO	Sequential Minimal Optimization
SMOTE	Synthetic Minority Oversampling Technique
SPL	Sound Pressure Level
SSE	Sum of Squared Error
SUSAS	Speech Under Simulated and Actual Stress
SVM	Support Vector Machine
UAR	Unweighted Average Recall
uLSIF	Unconstrained Least-Squares Importance Fitting
VAM	Vera am Mittag
VTLN	Vocal Tract Length Normalization
WAR	Weighted Average Recall
WOZ	Wizard of Oz
ZCR	Zero Crossing Rate

List of Symbols

Acoustic Features

F_0	Fundamental frequency
f	Linear frequency scale in Hz
$mel(f)$	Mel scale

Support Vector Machines

n	Input feature size
i	Index of sampling
$\mathbf{b}$	Offset bias vector for SVMs
$\mathbf{w}$	Weight matrix for SVMs
$\phi(\cdot)$	Nonlinear feature space mapping function
N	Number of examples
y	Referenced binary labels
$k(x_i, x_j)$	Kernel function
C	Penalty parameter
ξ_i	Slack variables
σ	Width of the Gaussian kernel

Neural Networks

l	Layer index
n_l	Number of layers
$f(\cdot)$	Nonlinear activation function
$f'(\cdot)$	Derivative of a function
$\mathbf{z}$	Activation vector
$\mathbf{h}^{(l)}$	Representation from the $l-$th hidden layer
$\mathbf{W}$	Weight matrix
$\mathbf{x}$	Example
$\mathbf{y}$	Predicted output vector
$\mathcal{J}(\mathbf{W}, \mathbf{b})$	Objective function
δ	Error term
τ	Iteration step
η	Learning rate
λ	Weight decay
μ	Momentum

Classification Evaluation

$\mathbf{X}$	Data set
$\mathbf{x}$	Example
f	Mapping functin
y	Predicted label
$\mathbf{X}^{te}$	Test set
t	Target label
N_t	Number of examples belonging to class t in the test set
$\lvert \cdot \rvert$	Cardinality of a set
$\#$	Total number of examples
p_t	Prior probability of class t in the test set

p_A	Probability of correct classification for system A
p_{AB}	Average probability of correct classification for system A and B
N_c	Number of correct classifications on the test set
N	Number of the test examples
$\mathrm{Bin}(\cdot)$	Binomial distribution function
z	Standard score
p	p-value
α	Significance level
$\Phi(\cdot)$	Standard normal cumulative distribution function

Feature Transfer Learning

$\beta(\cdot)$	Importance weights
$p(\cdot)$	Example density
$\Phi(\cdot)$	Canonical mapping functions
$\mathcal{J}$	Objective function
k	Kernel function
n_{tr}	Number of training examples
n_{te}	Number of test examples
C	Constant variable
t	Index of target domain
s	Index of source domain
$\mathcal{D}$	Data domain
$\mathbf{t}$	Label set
$\mathbf{X}^t$	Target domain data
$\mathbf{X}^s$	Source domain data
$\mathbf{x}$	Feature vector
n	Dimension of features
c	Label
n_c	Number of classes

m	Number of hidden units
$\mathbf{z}$	Activation vector
$\mathbf{h}$	Hidden representation
$f(\cdot)$	Activation function
$\mathbf{y}$	Reconstruction
$\mathbf{W}$	Weight matrix
$\mathbf{b}$	Bias vector
θ	Set of tuning parameters
δ	Error term
$\mathrm{SP}(\rho\|\|\hat{\rho}_j)$	KL divergence
ρ	Sparsity level
β	Sparsity penalty term
$q_{\mathcal{D}}$	Corrupting function
p_n	Given corruption level
$\mathrm{Bin}(\cdot)$	Binomial distribution function
γ	Hyper-parameter for SHLAs
λ	Weight decay
β	Hyper-parameter for adaptive DAEs
$\mathbf{I}$	Identity matrix
C	Regularization term for ELM-AEs
$\tilde{\mathbf{H}}$	Representations learned by ELM-AEs
$\mathbf{T}$	Target subspace
$\mathbf{S}$	Source subspace

Bibliography

[1] D. O'Shaughnessy, "Acoustic analysis for automatic speech recognition," *Proceedings of the IEEE*, vol. 101, no. 5, pp. 1038–1053, 2013.

[2] P. N. Juslin and P. Laukka, "Communication of emotions in vocal expression and music performance: Different channels, same code?" *Psychological bulletin*, vol. 129, no. 5, p. 770, 2003.

[3] B. Schuller, E. Marchi, S. Baron-Cohen, H. O'Reilly, D. Pigat, P. Robinson, I. Davies, O. Golan, S. Fridenson, S. Tal, S. Newman, N. Meir, R. Shillo, A. Camurri, S. Piana, A. Staglianò, S. Bölte, D. Lundqvist, S. Berggren, A. Baranger, and N. Sullings, "The state of play of ASC-inclusion: An integrated internet-based environment for social inclusion of children with autism spectrum conditions," in *Proc. IDGEI*. Haifa, Israel: ACM, 2014.

[4] L. Shen, M. Wang, and R. Shen, "Affective e-learning: Using emotional data to improve learning in pervasive learning environment," *Journal of Educational Technology & Society*, vol. 12, no. 2, pp. 176–189, 2009.

[5] C. Breazeal and L. Aryananda, "Recognition of affective communicative intent in robot-directed speech," *Autonomous robots*, vol. 12, no. 1, pp. 83–104, 2002.

[6] R. W. Picard, *Affective computing.* MIT Press, 1997.

[7] 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*, vol. 53, no. 9-10, pp. 1062–1087, 2011.

[8] R. A. Calvo and S. D'Mello, "Affect detection: An interdisciplinary review of models, methods, and their applications," *Affective Computing, IEEE Transactions on*, vol. 1, no. 1, pp. 18–37, 2010.

[9] M. El Ayadi, M. S. Kamel, and F. Karray, "Survey on speech emotion recognition: Features, classification schemes, and databases," *Pattern Recognition*, vol. 44, no. 3, pp. 572–587, 2011.

[10] C.-N. Anagnostopoulos, T. Iliou, and I. Giannoukos, "Features and classifiers for emotion recognition from speech: A survey from 2000 to 2011," *Artificial Intelligence Review*, pp. 1–23, 2012.

[11] J. Deng, Z. Zhang, F. Eyben, and B. Schuller, "Autoencoder-based unsupervised domain adaptation for speech emotion recognition," *Signal Processing Letters, IEEE*, vol. 21, no. 9, pp. 1068–1072, 2014.

[12] B. Schuller, B. Vlasenko, F. Eyben, M. Wöllmer, A. Stuhlsatz, A. Wendemuth, and G. Rigoll, "Cross-corpus acoustic emotion recognition: Variances and strategies," *Affective Computing, IEEE Transactions on*, vol. 1, no. 2, pp. 119–131, 2010.

[13] F. Eyben, A. Batliner, B. Schuller, D. Seppi, and S. Steidl, "Cross-corpus classification of realistic emotions – some pilot experiments," in *Proc. 3rd International Workshop on EMOTION (satellite of LREC): Corpora for Research on Emotion and Affect.* Valletta, Malta: ELRA, 2010, pp. 77–82.

[14] A. Halevy, P. Norvig, and F. Pereira, "The unreasonable effectiveness of data," *Intelligent Systems, IEEE*, vol. 24, no. 2, pp. 8–12, 2009.

[15] L. Torrey and J. Shavlik, "Transfer learning," *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, vol. 1, pp. 242–264, 2009.

[16] S. J. Pan and Q. Yang, "A survey on transfer learning," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 22, no. 10, pp. 1345–1359, 2010.

[17] B. Schuller, S. Steidl, A. Batliner, F. Burkhardt, L. Devillers, C. Müller, and S. Narayanan, "Paralinguistics in speech and language – state-of-the-art and the challenge," *Computer Speech & Language*, vol. 27, no. 1, pp. 4–39, 2013.

[18] B. Schuller, "The computational paralinguistics challenge," *IEEE Signal Processing Magazine*, vol. 29, no. 4, pp. 97–101, 2012.

[19] T. L. Nwe, S. W. Foo, and L. C. De Silva, "Speech emotion recognition using hidden Markov models," *Speech communication*, vol. 41, no. 4, pp. 603–623, 2003.

[20] B. Schuller and A. Batliner, *Computational Paralinguistics: Emotion, Affect and Personality in Speech and Language Processing.* Wiley, November 2013.

[21] B. S. Atal, “Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification,” *The Journal of the Acoustical Society of America*, vol. 55, no. 6, pp. 1304–1312, 1974.

[22] S. B. Davis and P. Mermelstein, “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,” *Acoustic, Speech and Signal Processing, IEEE Transactions on*, vol. 28, no. 4, pp. 357–366, 1980.

[23] F. Eyben, M. Wöllmer, and B. Schuller, “OpenSMILE: The Munich versatile and fast open-source audio feature extractor,” in *Proc. ACM-MM*, Florence, Italy, 2010, pp. 1459–1462.

[24] F. A. Eyben, “Real-time speech and music classification by large audio feature speech extraction,” Ph.D. dissertation, Technische Universität München, 2015.

[25] B. Schuller, S. Steidl, and A. Batliner, “The INTERSPEECH 2009 emotion challenge,” in *Proc. INTERSPEECH*, Brighton, UK, 2009, pp. 312–315.

[26] B. Schuller, M. Valstar, R. Cowie, and M. Pantic, “AVEC 2012: The continuous audio/visual emotion challenge – an introduction,” in *Proc. ICMI*, Santa Monica, CA, 2012, pp. 361–362.

[27] B. Schuller, S. Steidl, A. Batliner, J. Epps, F. Eyben, F. Ringeval, E. Marchi, and Y. Zhang, “The INTERSPEECH 2014 computational paralinguistics challenge: Cognitive & physical load,” in *Proc. INTERSPEECH*, 2014.

[28] A. Dhall, R. Goecke, J. Joshi, K. Sikka, and T. Gedeon, “Emotion recognition in the wild challenge 2014: Baseline, data and protocol,” in *Proc. ICMI*, New York, NY, USA, 2014, pp. 461–466.

[29] R. Fernandez and R. W. Picard, “Modeling drivers speech under stress,” *Speech Communication*, vol. 40, no. 1, pp. 145–159, 2003.

[30] G. Zhou, J. H. Hansen, and J. F. Kaiser, “Nonlinear feature based classification of speech under stress,” *Speech and Audio Processing, IEEE Transactions on*, vol. 9, no. 3, pp. 201–216, 2001.

[31] M. J. Alam, Y. Attabi, P. Dumouchel, P. Kenny, and D. D. O’Shaughnessy, “Amplitude modulation features for emotion recognition from speech,” in *Proc. INTERSPEECH*, Lyon, France, 2013, pp. 2420–2424.

[32] S. Young, G. Evermann, M. Gales, T. Hain, D.Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, *The HTK Book*, for HTK version 3.4 ed. Cambridge, MA: Cambridge University Engineering Department, 2006.

[33] T. Ganchev, N. Fakotakis, and G. Kokkinakis, “Comparative evaluation of various MFCC implementations on the speaker verification task,” in *Proc. SPECOM*, 2005, pp. 191–194.

[34] B. Logan *et al.*, “Mel frequency cepstral coefficients for music modeling.” in *Proc. ISMIR*, 2000.

[35] G. Tzanetakis and P. Cook, “Musical genre classification of audio signals,” *Speech and Audio Processing, IEEE transactions on*, vol. 10, no. 5, pp. 293–302, 2002.

[36] D. Neiberg, K. Elenius, and K. Laskowski, “Emotion recognition in spontaneous speech using GMMs.” in *Proc. INTERSPEECH*, 2006.

[37] D. Le and E. M. Provost, “Emotion recognition from spontaneous speech using hidden Markov models with deep belief networks,” in *Proc. ASRU*. IEEE, 2013, pp. 216–221.

[38] Y. Attabi, M. J. Alam, P. Dumouchel, P. Kenny, and D. O’Shaughnessy, “Multiple windowed spectral features for emotion recognition,” in *Proc. ICASSP*. IEEE, 2013, pp. 7527–7531.

[39] S. E. Kahou, C. Pal, X. Bouthillier, P. Froumenty, Ç. Gülçehre, R. Memisevic, P. Vincent, A. Courville, Y. Bengio, R. C. Ferrari *et al.*, “Combining modality specific deep neural networks for emotion recognition in video,” in *Proc. ICMI*. ACM, 2013, pp. 543–550.

[40] S. E. Kahou, X. Bouthillier, P. Lamblin, C. Gulcehre, V. Michalski, K. Konda, S. Jean, P. Froumenty, A. Courville, P. Vincent *et al.*, “EmoNets: Multimodal deep learning approaches for emotion recognition in video,” *arXiv preprint arXiv:1503.01800*, 2015.

[41] C. Busso, Z. Deng, S. Yildirim, M. Bulut, C. M. Lee, A. Kazemzadeh, S. Lee, U. Neumann, and S. Narayanan, “Analysis of emotion recognition using facial expressions, speech and multimodal information,” in *Proc. ICMI*. ACM, 2004, pp. 205–211.

[42] J. Kim and M. Clements, “Multimodal affect classification at various temporal lengths,” *Affective Computing, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2015.

[43] M. Wöllmer, B. Schuller, F. Eyben, and G. Rigoll, “Combining long short-term memory and dynamic bayesian networks for incremental emotion-sensitive artificial listening,” *Selected Topics in Signal Processing, IEEE Journal of*, vol. 4, no. 5, pp. 867–881, 2010.

[44] S. Greenberg and B. E. Kingsbury, "The modulation spectrogram: In pursuit of an invariant representation of speech," in *Proc. ICASSP*, vol. 3. IEEE, 1997, pp. 1647–1650.

[45] S. Wu, T. H. Falk, and W.-Y. Chan, "Automatic speech emotion recognition using modulation spectral features," *Speech Communication*, vol. 53, no. 5, pp. 768–785, 2011.

[46] N. Cummins, J. Epps, and E. Ambikairajah, "Spectro-temporal analysis of speech affected by depression and psychomotor retardation," in *Proc. ICASSP*. IEEE, 2013, pp. 7542–7546.

[47] T. Kinnunen, K.-A. Lee, and H. Li, "Dimension reduction of the modulation spectrogram for speaker verification." in *Proc. Odyssey*, 2008, p. 30.

[48] Z. Tüske, P. Golik, R. Schlüter, and H. Ney, "Acoustic modeling with deep neural networks using raw time signal for LVCSR," in *Proc. INTERSPEECH*, 2014, pp. 890–894.

[49] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proc. ICML*, 2014, pp. 1764–1772.

[50] D. Palaz, M. Magimai-Doss, and R. Collobert, "Convolutional neural networks-based continuous speech recognition using raw speech signal," in *Proc. ICASSP*. IEEE, 2015, pp. 4295–4299.

[51] D. Ververidis and C. Kotropoulos, "Emotional speech recognition: Resources, features, and methods," *Speech communication*, vol. 48, no. 9, pp. 1162–1181, 2006.

[52] F. Eyben, F. Weninger, F. Groß, and B. Schuller, "Recent developments in openSMILE, the Munich open-source multimedia feature extractor," in *Proc. ACM-MM*, Barcelona, Spain, 2013, pp. 835–838.

[53] 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. ICASSP*, vol. I, Montreal, Canada, May 2004, pp. 577–580.

[54] X. Xu, C. Huang, C. Wu, Q. Wang, and L. Zhao, "Graph learning based speaker independent speech emotion recognition," *Advances in Electrical and Computer Engineering*, vol. 14, no. 2, pp. 17–22, 2014.

[55] Z. Zhang, J. Deng, E. Marchi, and B. Schuller, "Active learning by label uncertainty for acoustic emotion recognition," in *Proc. INTERSPECCH*, Lyon, France, August 2013, pp. 2841–2845.

[56] Z. Zhang, F. Eyben, J. Deng, and B. Schuller, “An agreement and sparseness-based learning instance selection and its application to subjective speech phenomena,” in *Proc. ES^{3}LOD*, Reykjavik, Iceland, May 2014, pp. 21–26.

[57] Z. Zhang, E. Coutinho, J. Deng, and B. Schuller, “Distributing recognition in computational paralinguistics,” *Affective Computing, IEEE Transactions on*, vol. 5, no. 4, pp. 406–417, Oct 2014.

[58] D. Chazan, R. Hoory, G. Cohen, and M. Zibulski, “Speech reconstruction from mel frequency cepstral coefficients and pitch frequency,” in *Proc. of ICASSP*, vol. 3. IEEE, 2000, pp. 1299–1302.

[59] B. Milner and X. Shao, “Clean speech reconstruction from MFCC vectors and fundamental frequency using an integrated front-end,” *Speech Communication*, vol. 48, no. 6, pp. 697–715, 2006.

[60] S. McGilloway, R. Cowie, E. Douglas-Cowie, S. Gielen, M. Westerdijk, and S. Stroeve, “Approaching automatic recognition of emotion from voice: A rough benchmark,” in *Proc. ITRW Speech and Emotion*, 2000.

[61] D. Ververidis and C. Kotropoulos, “Emotional speech classification using Gaussian mixture models and the sequential floating forward selection algorithm,” in *Proc. ICME.* IEEE, 2005, pp. 1500–1503.

[62] L. Breiman, “Statistical modeling: The two cultures,” *Statistical Science*, vol. 16, no. 3, pp. 199–231, 2001.

[63] B. D. Womack and J. H. Hansen, “N-channel hidden Markov models for combined stressed speech classification and recognition,” *Speech and Audio Processing, IEEE Transactions on*, vol. 7, no. 6, pp. 668–677, 1999.

[64] B. Schuller, G. Rigoll, and M. Lang, “Hidden Markov model-based speech emotion recognition,” in *Proc. ICASSP*, Hong Kong, China, 2003, pp. 1–4.

[65] H. Meng and N. Bianchi-Berthouze, “Affective state level recognition in naturalistic facial and vocal expressions,” *Cybernetics, IEEE Transactions on*, vol. 44, no. 3, pp. 315–328, 2014.

[66] M. E. Ayadi, M. S. Kamel, and F. Karray, “Speech emotion recognition using Gaussian mixture vector autoregressive models,” in *Proc. ICASSP*, Las Vegas, NV, 2007, pp. 957–960.

[67] C. Busso, S. Lee, and S. Narayanan, “Analysis of emotionally salient aspects of fundamental frequency for emotion detection,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 17, no. 4, pp. 582–596, 2009.

[68] P. Dumouchel, N. Dehak, Y. Attabi, R. Dehak, and N. Boufaden, “Cepstral and long-term features for emotion recognition.” in *Proc. INTERSPEECH*, 2009, pp. 344–347.

[69] A. Janicki, “Non-linguistic vocalisation recognition based on hybrid GMM-SVM approach.” in *Proc. INTERSPEECH*, 2013, pp. 153–157.

[70] R. Xia, J. Deng, B. Schuller, and Y. Liu, “Modeling gender information for emotion recognition using denoising autoencoders,” in *Proc. ICASSP*, Florence, Italy, 2014, pp. 990–994.

[71] B. Schuller and L. Devillers, “Incremental acoustic valence recognition: An inter-corpus perspective on features, matching, and performance in a gating paradigm,” in *Proc. INTERSPEECH*, Makuhari, Japan, September 2010, pp. 2794–2797.

[72] R. Brückner and B. Schuller, “Likability classification – a not so deep neural network approach,” in *Proc. INTERSPEECH*, Portland, OR, 2012.

[73] Y. Kim, H. Lee, and E. M. Provost, “Deep learning for robust feature generation in audiovisual emotion recognition,” in *Proc. ICASSP.* IEEE, 2013, pp. 3687–3691.

[74] A. Stuhlsatz, C. Meyer, F. Eyben, T. Zielke, G. Meier, and B. Schuller, “Deep neural networks for acoustic emotion recognition: Raising the benchmarks,” in *Proc. ICASSP*, Prague, Czech Republic, 2011, pp. 5688–5691.

[75] G. Caridakis, L. Malatesta, L. Kessous, N. Amir, A. Raouzaiou, and K. Karpouzis, “Modeling naturalistic affective states via facial and vocal expressions recognition,” in *Proc. ICMI.* ACM, 2006, pp. 146–154.

[76] C.-H. Wu and W.-B. Liang, “Emotion recognition of affective speech based on multiple classifiers using acoustic-prosodic information and semantic labels,” *Affective Computing, IEEE Transactions on*, vol. 2, no. 1, pp. 10–21, 2011.

[77] D. Bone, T. Chaspari, K. Audhkhasi, J. Gibson, A. Tsiartas, M. Van Segbroeck, M. Li, S. Lee, and S. Narayanan, “Classifying language-related developmental disorders from speech cues: The promise and the potential confounds.” in *Proc. INTERSPEECH*, 2013, pp. 182–186.

[78] O. Pierre-Yves, “The production and recognition of emotions in speech: Features and algorithms,” *International Journal of Human-Computer Studies*, vol. 59, no. 1, pp. 157–183, 2003.

[79] C.-C. Lee, E. Mower, C. Busso, S. Lee, and S. Narayanan, “Emotion recognition using a hierarchical binary decision tree approach,” *Speech Communication*, vol. 53, no. 9, pp. 1162–1171, 2011.

[80] B. Schuller, S. Reiter, R. Müller, M. Al-Hames, M. Lang, and G. Rigoll, “Speaker independent speech emotion recognition by ensemble classification,” in *Proc. ICME*, Amsterdam, The Netherlands, July 2005, pp. 864–867.

[81] D. Morrison, R. Wang, and L. C. De Silva, “Ensemble methods for spoken emotion recognition in call-centres,” *Speech communication*, vol. 49, no. 2, pp. 98–112, 2007.

[82] K. Han, D. Yu, and I. Tashev, “Speech emotion recognition using deep neural network and extreme learning machine,” in *Proc. INTERSPEECH*, 2014, pp. 223–227.

[83] H. Kaya and A. A. Salah, “Combining modality-specific extreme learning machines for emotion recognition in the wild,” in *Proc. ICMI*, 2014, pp. 487–493.

[84] L. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[85] C. Bishop, *Pattern Recognition and Machine Learning.* Springer, 2006.

[86] N. Cristianini and J. Shawe-Taylor, “An introduction to support vector machines,” 2000.

[87] B. D. Womack and J. H. Hansen, “Classification of speech under stress using target driven features,” *Speech Communication*, vol. 20, no. 1, pp. 131–150, 1996.

[88] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[89] Q. Mao, M. Dong, Z. Huang, and Y. Zhan, “Learning salient features for speech emotion recognition using convolutional neural networks,” *Multimedia, IEEE Transactions on*, vol. 16, no. 8, pp. 2203–2213, Dec 2014.

[90] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim, “Do we need hundreds of classifiers to solve real world classification problems?” *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3133–3181, 2014.

[91] B. Schuller, *Intelligent Audio Analysis*, ser. Signals and Communication Technology. Springer, 2013, 350 pages.

[92] B. Schuller, S. Steidl, A. Batliner, S. Hantke, F. Hönig, J. R. Orozco-Arroyave, E. Nöth, Y. Zhang, and F. Weninger, "The INTERSPEECH 2015 computational paralinguistics challenge: Degree of nativeness, parkinson's & eating condition," in *Proc. INTERSPEECH*, Dresden, Germany, 2015, 5 pages, to appear.

[93] F. Eyben and B. Schuller, "openSMILE: The Munich open-source large-scale multimedia feature extractor," *ACM SIGMM Records*, vol. 6, no. 4, December 2014.

[94] V. Vapnik, *The nature of statistical learning theory*, 2nd ed. Berlin: Springer, 1999.

[95] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[96] S. Boyd and L. Vandenberghe, *Convex Optimization.* New York, NY, USA: Cambridge University Press, 2004.

[97] 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.

[98] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *Neural Networks, IEEE Transactions on*, vol. 13, no. 2, pp. 415–425, 2002.

[99] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *The Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[100] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 1–27, 2011.

[101] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, 2009.

[102] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[103] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *Proc. ICDAR*, vol. 2. IEEE Computer Society, 2003, pp. 958–958.

[104] P. Golik, P. Doetsch, and H. Ney, “Cross-entropy vs. squared error training: A theoretical and experimental comparison.” in *Proc. INTERSPEECH*, 2013, pp. 1756–1760.

[105] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, “Efficient backprop,” in *Neural networks: Tricks of the trade.* Springer, 2012, pp. 9–48.

[106] X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier networks,” in *Proc. AISTATS*, vol. 15, 2011, pp. 315–323.

[107] C. Dugas, Y. Bengio, F. Bélisle, C. Nadeau, and R. Garcia, “Incorporating second-order functional knowledge for better option pricing,” in *Proc. NIPS*, 2001, pp. 472–478.

[108] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, “Maxout networks,” in *Proc. ICML*, 2013, pp. 1319–1327.

[109] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification,” *arXiv preprint arXiv:1502.01852*, 2015.

[110] D. E. Rumelhart, G. E. Hintont, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, pp. 533–536, 1986.

[111] D. B. Parker, “Learning-logic,” Center for Comp. Research in Economics and Management Sci., MIT, Tech. Rep. TR-47, 1985.

[112] Y. LeCun, “Une procédure d’apprentissage pour réseau à seuil asymétrique,” *Proc. Cognitiva5, Paris*, pp. 599–604, 1985.

[113] ——, “A theoretical framework for back-propagation,” in *Proc. CMSS*, D. Touretzky, G. Hinton, and T. Sejnowski, Eds. CMU, Pittsburgh, Pa, USA: Morgan Kaufmann, 1988, pp. 21–28.

[114] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, “A limited memory algorithm for bound constrained optimization,” *SIAM Journal on Scientific Computing*, vol. 16, no. 5, pp. 1190–1208, 1995.

[115] J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, Q. V. Le, and A. Y. Ng, “On optimization methods for deep learning,” in *Proc. ICML*, 2011, pp. 265–272.

[116] L. Bottou and O. Bousquet, “The tradeoffs of large scale learning,” in *Proc. NIPS*, 2008, pp. 161–168.

[117] C. M. Bishop, *Neural networks for pattern recognition.* Oxford university press, 1995.

[118] Y. Bengio, P. Simard, and P. Frasconi, “Learning long-term dependencies with gradient descent is difficult,” *Neural Networks, IEEE Transactions on*, vol. 5, no. 2, pp. 157–166, 1994.

[119] Y. Bengio, I. J. Goodfellow, and A. Courville, “Deep learning,” 2015, book in preparation for MIT Press. [Online]. Available: http://www.iro.umontreal.ca/~bengioy/dlbook

[120] B. T. Polyak, “Some methods of speeding up the convergence of iteration methods,” *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, 1964.

[121] Y. Nesterov, “A method of solving a convex programming problem with convergence rate $\mathcal{O}(1/\sqrt{k})$,” *Soviet Mathematics Doklady*, vol. 27, no. 2, 1983.

[122] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, “On the importance of initialization and momentum in deep learning,” in *Proc. ICML*, 2013, pp. 1139–1147.

[123] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” *arXiv preprint arXiv:1207.0580*, 2012.

[124] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[125] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[126] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, “Greedy layer-wise training of deep networks,” in *Proc. NIPS*, Vancouver, Canada, 2007, pp. 153–160.

[127] M. Ranzato, C. Poultney, S. Chopra, and Y. LeCun, “Efficient learning of sparse representations with an energy-based model,” in *Proc. NIPS*, 2006, pp. 1137–1144.

[128] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[129] T. N. Sainath, A.-r. Mohamed, B. Kingsbury, and B. Ramabhadran, “Deep convolutional neural networks for LVCSR,” in *Proc. ICASSP*. IEEE, 2013, pp. 8614–8618.

[130] T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. Černockỳ, “Strategies for training large scale neural network language models,” in *Proc. ASRU*, 2011, pp. 196–201.

[131] F. Weninger, J. Geiger, M. Wöllmer, B. Schuller, and G. Rigoll, “Feature enhancement by deep LSTM networks for ASR in reverberant multisource environments,” *Computer Speech and Language*, vol. 28, no. 4, pp. 888–902, July 2014.

[132] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Proc. NIPS*, 2012, pp. 1097–1105.

[133] C. Szegedy, A. Toshev, and D. Erhan, “Deep neural networks for object detection,” in *Proc. NIPS*, 2013, pp. 2553–2561.

[134] J. Ma, R. P. Sheridan, A. Liaw, G. E. Dahl, and V. Svetnik, “Deep neural nets as a method for quantitative structure–activity relationships,” *Journal of chemical information and modeling*, vol. 55, no. 2, pp. 263–274, 2015.

[135] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, “Natural language processing (almost) from scratch,” *The Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.

[136] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Proc. NIPS*, 2014, pp. 3104–3112.

[137] R. Brückner and B. Schuller, “Social signal classification using deep BLSTM recurrent neural networks,” in *Proc. ICASSP*, Florence, Italy, May 2014, pp. 4856–4860.

[138] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proc. CVPR*, 2015, pp. 1–9.

[139] G. E. Hinton, S. Osindero, and Y.-W. Teh, “A fast learning algorithm for deep belief nets,” *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[140] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” *arXiv preprint arXiv:1408.5093*, 2014.

[141] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, “Theano: A CPU and GPU math expression compiler,” in *Proc. SciPy*, Jun. 2010.

[142] F. Weninger, J. Bergmann, and B. Schuller, “Introducing CURRENNT: the Munich open-source CUDA recurrent neural network toolkit,” *The Journal of Machine Learning Research*, vol. 16, pp. 547–551, 2015.

[143] R. J. Williams and D. Zipser, “A learning algorithm for continually running fully recurrent neural networks,” *Neural computation*, vol. 1, no. 2, pp. 270–280, 1989.

[144] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[145] Y. Bengio, “Learning deep architectures for AI,” *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[146] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 8, pp. 1798–1828, 2013.

[147] L. Deng, “A tutorial survey of architectures, algorithms, and applications for deep learning,” *APSIPA Transactions on Signal and Information Processing*, vol. 3, p. e2, 2014.

[148] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural Networks*, vol. 61, pp. 85–117, 2015.

[149] K. Kroschel, G. Rigoll, and B. Schuller, *Statistische informationstechnik*, 5th ed. Springer, 2011.

[150] T. G. Dietterich, “Approximate statistical tests for comparing supervised classification learning algorithms,” *Neural Computation*, vol. 10, pp. 1895–1923, 1998.

[151] A. Torralba and A. A. Efros, “Unbiased look at dataset bias,” in *Proc. CVPR.* IEEE, 2011, pp. 1521–1528.

[152] J. Ponce, T. L. Berg, M. Everingham, D. A. Forsyth, M. Hebert, S. Lazebnik, M. Marszalek, C. Schmid, B. C. Russell, A. Torralba, C. K. I. Williams, J. Zhang, and A. Zisserman, “Dataset issues in object recognition,” in *Toward category-level object recognition.* Springer, 2006, pp. 29–48.

[153] A. Margolis, K. Livescu, and M. Ostendorf, “Domain adaptation with unlabeled data for dialog act tagging,” in *Proc. Workshop on Domain Adaptation for Natural Language Processing.* Association for Computational Linguistics, 2010, pp. 45–52.

[154] E. L. Thorndike and R. S. Woodworth, “The influence of improvement in one mental function upon the efficiency of other functions. II. the estimation of magnitudes.” *Psychological Review*, vol. 8, pp. 247–261, 1901.

[155] B. F. Skinner, *Science and human behavior.* Simon and Schuster, 1953.

[156] R. Caruana, “Multitask learning,” *Machine Learning*, vol. 28, no. 1, pp. 41–75, 1997.

[157] R. Xia and Y. Liu, “Leveraging valence and activation information via multi-task learning for categorical emotion recognition,” in *Proc. ICASSP*, 2015.

[158] M. E. Taylor and P. Stone, “Transfer learning for reinforcement learning domains: A survey,” *The Journal of Machine Learning Research*, vol. 10, pp. 1633–1685, 2009.

[159] F.-F. Li, R. Fergus, and P. Perona, “One-shot learning of object categories,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 4, pp. 594–611, 2006.

[160] R. Socher, M. Ganjoo, C. D. Manning, and A. Ng, “Zero-shot learning through cross-modal transfer,” in *Proc. NIPS*, 2013, pp. 935–943.

[161] Z. Zhang, “Semi-autonomous data enrichment and optimisation for intelligent speech analysis,” Ph.D. dissertation, Technische Universität München, 2015.

[162] Z. Zhang, J. Deng, and B. Schuller, “Co-training succeeds in computational paralinguistics,” in *Proc. ICASSP*, Vancouver, Canada, May 2013, pp. 8505–8509.

[163] Z. Zhang, E. Coutinho, J. Deng, and B. Schuller, “Cooperative learning and its application to emotion recognition from speech,” *Speech and Language Processing, IEEE/ACM Transactions on Audio*, vol. 23, no. 1, pp. 115–126, 2015.

[164] B. Schuller, Y. Zhang, F. Eyben, and F. Weninger, “Intelligent systems’ holistic evolving analysis of real-life universal speaker characteristics,” in *Proc. ES^{3}LOD*, B. Schuller, P. Buitelaar, L. Devillers, C. Pelachaud, T. Declerck, A. Batliner, P. Rosso, and S. Gaines, Eds., Reykjavik, Iceland, 2014, pp. 14–20.

[165] J. Deng, Z. Zhang, E. Marchi, and B. Schuller, “Sparse autoencoder-based feature transfer learning for speech emotion recognition,” in *Proc. ACII*, Geneva, Switzerland, 2013, pp. 511–516.

[166] X. Glorot, A. Bordes, and Y. Bengio, “Domain adaptation for large-scale sentiment classification: A deep learning approach,” in *Proc. ICML*, Bellevue, Washington, USA, 2011, pp. 513–520.

[167] R. Gopalan, R. Li, and R. Chellappa, “Unsupervised adaptation across domain shifts by generating intermediate data representations,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 11, pp. 2288–2302, 2014.

[168] T. Kanamori, S. Hido, and M. Sugiyama, “Efficient direct density ratio estimation for non-stationarity adaptation and outlier detection,” in *Proc. NIPS*, Vancouver, BC, Canada, 2008, pp. 809–816.

[169] M. Sugiyama, S. Nakajima, H. Kashima, P. von Bünau, and M. Kawanabe, “Direct importance estimation with model selectionand its application to covariate shift adaptation,” in *Proc. NIPS*, Vancouver, BC, Canada, 2007, pp. 1433–1440.

[170] A. Hassan, R. Damper, and M. Niranjan, “On acoustic emotion recognition: Compensating for covariate shift,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 21, no. 7, pp. 1458–1468, 2013.

[171] J. Huang, A. Gretton, K. M. Borgwardt, B. Schölkopf, and A. J. Smola, “Correcting sample selection bias by unlabeled data,” in *Proc. NIPS*, 2006, pp. 601–608.

[172] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, “Covariate shift by kernel mean matching,” *Dataset shift in machine learning*, vol. 3, no. 4, pp. 131–160, 2009.

[173] T. Kanamori, S. Hido, and M. Sugiyama, “A least-squares approach to direct importance estimation,” *The Journal of Machine Learning Research*, vol. 10, pp. 1391–1445, 2009.

[174] L. Lee and R. C. Rose, “Speaker normalization using efficient frequency warping procedures,” in *Proc. ICASSP*, vol. 1. IEEE, 1996, pp. 353–356.

[175] N. Jaitly and G. E. Hinton, “Vocal tract length perturbation (VTLP) improves speech recognition,” in *Proc. ICML Workshop on Deep Learning for Audio, Speech and Language*, 2013.

[176] C. J. Leggetter and P. C. Woodland, “Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models,” *Computer Speech & Language*, vol. 9, no. 2, pp. 171–185, 1995.

[177] J. S. Bridle and S. J. Cox, "Recnorm: Simultaneous normalization and classification applied to speech recognition," in *Proc. NIPS*, 1990, pp. 234–240.

[178] S. P. Kishore and B. Yegnanarayana, "Speaker verification: Minimizing the channel effects using autoassociative neural network models," in *Proc. ICASSP*, vol. 2, Istanbul, Turkey, 2000, pp. 1101–1104.

[179] S. Garimella, S. H. Mallidi, and H. Hermansky, "Regularized auto-associative neural networks for speaker verification," *Signal Processing Letters, IEEE*, vol. 19, no. 12, pp. 841–844, 2012.

[180] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *Proc. ASRU*, 2013, pp. 55–59.

[181] O. Abdel-Hamid and H. Jiang, "Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code," in *Proc. ICASSP*, 2013, pp. 7942–7946.

[182] J. Neto, L. Almeida, M. Hochberg, C. Martins, L. Nunes, S. Renals, and T. Robinson, "Speaker-adaptation for hybrid HMM-ANN continuous speech recognition system," in *Proc. Eurospeech*, 1995, pp. 2171–2174.

[183] Z. Zhang, F. Weninger, M. Wöllmer, and B. Schuller, "Unsupervised learning in cross-corpus acoustic emotion recognition," in *Proc. ASRU*, Big Island, HY, 2011, pp. 523–528.

[184] E. Coutinho, J. Deng, and B. Schuller, "Transfer learning emotion manifestation across music and speech," in *Proc. IJCNN*, Beijing, China, July 2014, pp. 3592–3598.

[185] H. Daumé III, "Frustratingly easy domain adaptation," in *Proc. ACL*, 2007.

[186] J. Blitzer, R. McDonald, and F. Pereira, "Domain adaptation with structural correspondence learning," in *Proc. EMNLP*. Association for Computational Linguistics, 2006, pp. 120–128.

[187] A. Argyriou and T. Evgeniou, "Multi-task feature learning," *Proc. NIPS*, vol. 19, pp. 41–48, 2007.

[188] S. Si, D. Tao, and B. Geng, "Bregman divergence-based regularization for transfer subspace learning," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 22, no. 7, pp. 929–942, 2010.

[189] B. A. Olshausen and J. D. Field, “Emergence of simple-cell receptive field properties by learning a sparse code for natural images,” *Nature*, vol. 381, no. 6583, pp. 607–609, 1996.

[190] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, “Self-taught learning: Transfer learning from unlabeled data,” in *Proc. ICML*, 2007, pp. 759–766.

[191] A. Maurer, M. Pontil, and B. Romera-Paredes, “Sparse coding for multitask and transfer learning,” in *Proc. ICML*, 2013, pp. 343–351.

[192] M. Chen, Z. Xu, K. Weinberger, and F. Sha, “Marginalized stacked denoising autoencoders for domain adaptation,” in *Proc. ICML*, Edinburgh, Scotland, 2012.

[193] Y. LeCun, “Modeles connexionnistes de l’apprentissage (connectionist learning models),” Ph.D. dissertation, Université P. et M. Curie (Paris 6), June 1987.

[194] H. Bourlard and Y. Kamp, “Auto-association by multilayer perceptrons and singular value decomposition,” *Biological cybernetics*, vol. 59, no. 4-5, pp. 291–294, 1988.

[195] G. E. Hinton and R. S. Zemel, “Autoencoders, minimum description length, and Helmholtz free energy,” in *Proc. NIPS*, Denver, Colorado, U. S. A., 1993, pp. 3–10.

[196] I. Goodfellow, H. Lee, Q. V. Le, A. Saxe, and A. Y. Ng, “Measuring invariances in deep networks,” in *Proc. NIPS*, Vancouver, Canada, 2009, pp. 646–654.

[197] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” in *Proc. ICML*, Helsinki, Finland, 2008, pp. 1096–1103.

[198] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, “Contractive auto-encoders: Explicit invariance during feature extraction,” in *Proc. ICML*, 2011, pp. 833–840.

[199] G. Alain and Y. Bengio, “What regularized auto-encoders learn from the data-generating distribution,” *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3563–3593, 2014.

[200] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, “Stacked convolutional auto-encoders for hierarchical feature extraction,” in *Proc. ICANN*. Springer, 2011, pp. 52–59.

[201] L. Deng, “Computational models for speech production,” in *Computational models of speech pattern processing*. Springer, 1999, pp. 199–213.

[202] ——, "Switching dynamic system models for speech articulation and acoustics," in *Mathematical Foundations of Speech and Language Processing.* Springer, 2004, pp. 115–133.

[203] H. Lee, C. Ekanadham, and Y. A. Ng, "Sparse deep belief net model for visual area V2," in *Proc. NIPS*, 2008, pp. 873–880.

[204] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, pp. 79–86, 1951.

[205] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 30–42, 2012.

[206] J. Deng, R. Xia, Z. Zhang, Y. Liu, and B. Schuller, "Introducing shared-hidden-layer autoencoders for transfer learning and their application in acoustic emotion recognition," in *Proc. ICASSP*, Florence, Italy, 2014, pp. 4851–4855.

[207] R. Xia and Y. Liu, "Using denoising autoencoder for emotion recognition," in *Proc. INTERSPEECH*, Lyon, France, 2013, pp. 2886–2889.

[208] E. Marchi, F. Vesperini, F. Eyben, S. Squartini, and B. Schuller, "A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional lstm neural networks," in *Proc. ICASSP.* Brisbane, Australia: IEEE, 2015.

[209] C. Kandaswamy, L. M. Silva, L. Alexandre, R. Sousa, J. M. Santos, J. M. de Sá *et al.*, "Improving transfer learning accuracy by reusing stacked denoising autoencoders," in *Proc. SMC.* IEEE, 2014, pp. 1380–1387.

[210] Y. Bengio, "Deep learning of representations for unsupervised and transfer learning," in *Proc. ICML*, Bellevue, Washington, USA, 2011, pp. 17–36.

[211] S. J. Hwang, F. Sha, and K. Grauman, "Sharing features between objects and their attributes," in *Proc. CVPR.* IEEE, 2011, pp. 1761–1768.

[212] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," in *Proc. ICML*, 2011, pp. 689–696.

[213] N. Srivastava and R. R. Salakhutdinov, "Multimodal learning with deep boltzmann machines," in *Proc. NIPS*, 2012, pp. 2222–2230.

[214] T. Tommasi, F. Orabona, and B. Caputo, "Safety in numbers: Learning categories from few examples with multi model knowledge transfer," in *Proc. CVPR*, 2010.

[215] Y. Aytar and A. Zisserman, "Tabula rasa: Model transfer for object category detection," in *Proc. ICCV*, Barcelona, Spain, 2011, pp. 2252–2259.

[216] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Proc. NIPS Deep Learning Workshop*, 2015.

[217] J. Li, R. Zhao, J.-T. Huang, and Y. Gong, "Learning small-size DNN with output-distribution-based criteria," in *Proc. INTERSPEECH*, 2014.

[218] J. Ba and R. Caruana, "Do deep nets really need to be deep?" in *Proc. NIPS*, 2014, pp. 2654–2662.

[219] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," in *Proc. ICLR*, 2015.

[220] C. Bucilua, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proc. SIGKDD.* ACM, 2006, pp. 535–541.

[221] G. Huang, G.-B. Huang, S. Song, and K. You, "Trends in extreme learning machines: A review," *Neural Networks*, vol. 61, pp. 32–48, 2015.

[222] X. Xu, J. Deng, W. Zheng, L. Zhao, and B. Schuller, "Dimensionality reduction for speech emotion features by multiscale kernels," in *Proc. INTERSPEECH*, 2015.

[223] L. L. C. Kasun, , H. Zhou, and G.-B. Huang, "Representational learning with ELMs for big data," *Intelligent Systems, IEEE*, vol. 28, no. 6, pp. 31–34, Nov 2013.

[224] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 42, no. 2, pp. 513–529, 2012.

[225] J. Deng, Z. Zhang, and B. Schuller, "Linked source and target domain subspace feature transfer learning – Exemplified by speech emotion recognition," in *Proc. ICPR*, Stockholm, Sweden, August 2014, pp. 761–766.

[226] M. Shao, D. Kit, and Y. Fu, "Generalized transfer subspace learning through low-rank constraint," *International Journal of Computer Vision*, vol. 109, no. 1-2, pp. 74–93, 2014.

[227] T. Nakashika, R. Takashima, T. Takiguchi, and Y. Ariki, "Voice conversion in high-order eigen space using deep belief nets." in *Proc. INTERSPEECH*, 2013, pp. 369–372.

[228] T. Nakashika, T. Takiguchi, and Y. Ariki, “Voice conversion using RNN pre-trained by recurrent temporal restricted Boltzmann machines,” *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, vol. 23, no. 3, pp. 580–587, 2015.

[229] ——, “Voice conversion using speaker-dependent conditional restricted Boltzmann machine,” *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2015, no. 1, pp. 1–12, 2015.

[230] J. Deng and B. Schuller, “Confidence measures in speech emotion recognition based on semi-supervised learning,” in *Proc. INTERSPEECH*, Portland, OR, September 2012, 4 pages.

[231] J. Deng, W. Han, and B. Schuller, “Confidence measures for speech emotion recognition: A start,” in *Proc. ITG Symposium*, Braunschweig, Germany, September 2012, pp. 1–4.

[232] B. Schuller, D. Arsic, G. Rigoll, M. Wimmer, and B. Radig, “Audiovisual behavior modeling by combined feature spaces,” in *Proc. ICASSP*, Honolulu, HY, 2007, pp. 733–736.

[233] 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, Special Issue on Visual and Multimodal Analysis of Human Spontaneous Behavior*, vol. 27, no. 12, pp. 1760–1774, November 2009.

[234] B. Schuller, S. Steidl, A. Batliner, F. Burkhardt, L. Devillers, C. Müller, and S. Narayanan, “The INTERSPEECH 2010 paralinguistic challenge,” in *Proc. INTERSPEECH*, Makuhari, Japan, 2010, pp. 2794–2797.

[235] F. Burkhardt, A. Paeschke, M. Rolfes, W. Sendlmeier, and B. Weiss, “A database of German emotional speech,” in *Proc. INTERSPEECH*, Lissabon, Portuga, 2005, pp. 1517–1520.

[236] H. Meng, J. Pittermann, A. Pittermann, and W. Minker, “Combined speech-emotion recognition for spoken human-computer interfaces,” in *Proc. SPCOM*, Dubai, United Emirates, 2007, pp. 1179–1182.

[237] V. Slavova, W. Verhelst, and H. Sahli, “A cognitive science reasoning in recognition of emotions in audio-visual speech,” *International Journal Information Technologies and Knowledge*, vol. 2, pp. 324–334, 2008.

[238] B. Schuller, M. Wimmer, L. Msenlechner, C. Kern, and G. Rigoll, “Brute-forcing hierarchical functionals for paralinguistics: A waste of feature space?” in *Proc. ICASSP*, Las Vegas, Nevada, USA, 2008, pp. 4501–4504.

[239] O. Martin, I. Kotsia, B. Macq, and I. Pitas, “The eNTERFACE’05 audio-visual emotion database,” in *Proc. IEEE Workshop on Multimedia Database Management*, 2006.

[240] S. Steidl, A. Batliner, B. Schuller, and D. Seppi, “The hinterland of emotions: Facing the open-microphone challenge,” in *Proc. ACII.* IEEE, 2009, pp. 1–8.

[241] S. Steidl, *Automatic Classification of Emotion Related User States in Spontaneous Children’s Speech.* Logos Verlag, 2009.

[242] B. Schuller, S. Steidl, A. Batliner, A. Vinciarelli, K. Scherer, F. Ringeval, M. Chetouani, F. Weninger, F. Eyben, E. Marchi, M. Mortillaro, H. Salamin, A. Polychroniou, F. Valente, and S. Kim, “The INTERSPEECH 2013 computational paralinguistics challenge: Social signals, conflict, emotion, autism,” in *Proc. INTERSPEECH*, Lyon, France, 2013, pp. 148–152.

[243] J. Hansen and S. Bou-Ghazale, “Getting started with SUSAS: A speech under simulated and actual stress database,” in *Proc. EUROSPEECH*, Rhodes, Greece, 1997.

[244] M. Grimm, K. Kroschel, and S. Narayanan, “The Vera am Mittag German audio-visual emotional speech database,” in *Proc. ICME*, Hannover, Germany, 2008, pp. 865–868.

[245] H. Schlosberg, “Three dimensions of emotion.” *Psychological review*, vol. 61, no. 2, p. 81, 1954.

[246] J. A. Russell, “A circumplex model of affect,” *Journal of Personality and Social Psychology*, vol. 39, no. 6, pp. 1161–1178, 1980.

[247] B. Schuller, B. Vlasenko, F. Eyben, G. Rigoll, and A. Wendemuth, “Acoustic emotion recognition: A benchmark comparison of performances,” in *Proc. ASRU*, IEEE. Merano, Italy: IEEE, December 2009, pp. 552–557.

[248] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, pp. 321–357, 2002.

[249] Y. Bengio, “Practical recommendations for gradient-based training of deep architectures,” in *Neural Networks: Tricks of the Trade*, 2012, pp. 437–478.

[250] F. H. Knower, “Analysis of some experimental variations of simulated vocal expressions of the emotions,” *The Journal of Social Psychology*, vol. 14, no. 2, pp. 369–372, 1941.

[251] I. R. Murray and J. L. Arnott, “Toward the simulation of emotion in synthetic speech: A review of the literature on human vocal emotion,” *The Journal of the Acoustical Society of America*, vol. 93, no. 2, pp. 1097–1108, 1993.

[252] P. Mitev and S. Hadjitodorov, “Fundamental frequency estimation of voice of patients with laryngeal disorders,” *Information Sciences*, vol. 156, no. 1, pp. 3–19, 2003.

[253] Y. Jin, Y. Zhao, C. Huang, and L. Zhao, “Study on the emotion recognition of whispered speech,” in *Proc. GCIS*, vol. 3, Xiamen, China, 2009, pp. 242–246.

[254] C. Gong, H. Zhao, W. Zou, Y. Wang, and M. Wang, “A preliminary study on emotions of Chinese whispered speech,” in *Proc. IFCSTA*, vol. 2, Chongqing, China, 2009, pp. 429–433.

[255] X. Fan and J. H. Hansen, “Acoustic analysis and feature transformation from neutral to whisper for speaker identification within whispered speech audio streams,” *Speech communication*, vol. 55, no. 1, pp. 119–134, 2013.

[256] C. Zhang and J. H. Hansen, “An advanced entropy-based feature with a frame-level vocal effort likelihood space modeling for distant whisper-island detection,” *Speech Communication*, vol. 66, pp. 107–117, 2015.

[257] J. Zhou, R. Liang, L. Zhao, L. Tao, and C. Zou, “Unsupervised learning of phonemes of whispered speech in a noisy environment based on convolutive non-negative matrix factorization,” *Information Sciences*, vol. 257, pp. 115–126, 2014.

[258] T. Irino, Y. Aoki, H. Kawahara, and R. D. Patterson, “Comparison of performance with voiced and whispered speech in word recognition and mean-formant-frequency discrimination,” *Speech Communication*, vol. 54, no. 9, pp. 998–1013, 2012.

[259] M. Janke, M. Wand, T. Heistermann, T. Schultz, and K. Prahallad, “Fundamental frequency generation for whisper-to-audible speech conversion,” in *Proc. ICASSP*, Florence, Italy, 2014, pp. 2598–2602.

[260] H. R. Sharifzadeh, I. V. McLoughlin, and M. J. Russell, “A comprehensive vowel space for whispered speech,” *Journal of Voice*, vol. 26, no. 2, pp. e49–e56, 2012.

[261] S. Bou-Ghazale and J. Hansen, “Stress perturbation of neutral speech for synthesis based on hidden Markov models,” *Speech and Audio Processing, IEEE Transactions on*, vol. 6, no. 3, pp. 201–216, 1998.

[262] T. Ito, K. Takeda, and F. Itakura, “Analysis and recognition of whispered speech,” *Speech Communication*, vol. 45, no. 2, pp. 139–152, 2005.

[263] R. W. Morris and M. A. Clements, “Reconstruction of speech from whispers,” *Medical Engineering & Physics*, vol. 24, no. 7, pp. 515–520, 2002.

[264] M. Matsuda and H. Kasuya, “Acoustic nature of the whisper,” in *Proc. Eurospeech*, Budapest, Hungary, 1999, pp. 133–136.

[265] S. T. Jovičić, “Formant feature differences between whispered and voiced sustained vowels,” *Acta Acustica united with Acustica*, vol. 84, no. 4, pp. 739–743, 1998.

[266] W. F. Heeren and C. Lorenzi, “Perception of prosody in normal and whispered french,” *The Journal of the Acoustical Society of America*, vol. 135, no. 4, pp. 2026–2040, 2014.

[267] C. Busso, A. Metallinou, and S. S. Narayanan, “Iterative feature normalization for emotional speech detection,” in *Proc. ICASSP*, 2011, pp. 5692–5695.

[268] V. Sethu, J. Epps, and E. Ambikairajah, “Speaker variability in speech based emotion models-Analysis and normalisation,” in *Proc. ICASSP*, Vancouver, BC, Canada, 2013, pp. 7522–7526.

[269] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, “Boosting for transfer learning,” in *Proc. ICML*, Corvallis, OR, USA, 2007, pp. 193–200.

[270] S. J. Pan, V. W. Zheng, Q. Yang, and D. H. Hu, “Transfer learning for wifi-based indoor localization,” in *Proc. AAAI*, Chicago, Illinois, USA, 2008.