Tampere University

Einari Vaaras

AUTOMATIC EMOTIONAL SPEECH ANALYSIS FROM DAYLONG CHILD-CENTERED RECORDINGS FROM A NEONATAL INTENSIVE CARE UNIT

Faculty of Information Technology and Communication Sciences
Master of Science Thesis
April 2021

ABSTRACT

Einari Vaaras: Automatic Emotional Speech Analysis from Daylong Child-Centered Recordings from a Neonatal Intensive Care Unit
Master of Science Thesis
Tampere University
Master's Degree Programme in Electrical Engineering
April 2021

Speech emotion recognition (SER) is the task of recognizing the emotional state of the speaker from a speech signal. One potential field of application for SER is the study of the effect of parental proximity and communication to the early cognitive development of preterm infants. A crucial aspect in this kind of research is the analysis of the emotional content of speech that the preterm infants hear during intensive care. However, manual analysis of emotions in speech is highly time-consuming and expensive. Hence, an automatic SER system is essentially required for performing large-scale emotional speech analysis.

In the present study, a system which performs SER for real-life child-centered audio samples from a neonatal intensive care unit (NICU) was developed. Typically, with enough labeled training data, a traditional supervised machine learning approach could be taken to address this task. However, the primary audio material of the present experiments recorded in a NICU contains hundreds of hours of audio, and is thus far too large to be fully annotated manually. Therefore, alternative machine learning-based approaches, namely cross-corpus generalization, k-medoids clustering-based active learning (AL), and Wasserstein generative adversarial network-based domain adaptation (DA), are compared in the present experiments.

Since the dataset from the NICU was initially unannotated and the manual annotation of the recordings is laborious, simulations with four already existing SER corpora were first conducted to find out what would be the best approach for deploying a SER system on a novel unannotated corpus. Then, a subset of the NICU dataset was annotated, and the discovered solutions from the simulations were applied this subset to test how the simulated strategies would work in practice.

As a result, the DA method outperformed the cross-corpus generalization approach in situations when there are no labeled data available for the target corpus. With a moderate human annotation effort, the AL method was superior compared to the DA method for the classification of valence when approximately 4% of the NICU data was annotated. With the same number of annotated samples, the DA method slightly outperformed the AL method when classifying arousal. For a binary classification for valence, the best-performing model was a support vector machine classifier utilizing the AL method with a classification accuracy of 73.4% unweighted average recall (UAR). For arousal, the best model for a binary classification was a neural network-based classifier using the DA method with an accuracy of 73.2% UAR.

Keywords: speech emotion recognition, paralinguistic speech processing, active learning, domain adaptation, LENA recorder

TIIVISTELMÄ

Einari Vaaras: Automaattinen emotionaalisen puheen analyysi päivämittaisista lapsikeskeisistä äänitallenteista vastasyntyneiden teho-osastolta
Diplomityö
Tampereen yliopisto
Sähkötekniikan DI-ohjelma
Huhtikuu 2021

Puheen tunteiden tunnistuksessa (SER, Speech Emotion Recognition) tarkoituksena on tunnistaa puhujan emotionaalinen tila puhesignaalista. Yksi potentiaalinen soveltamisala SER:ille on tutkimus vanhempien läheisyyden ja kommunikaation vaikutuksesta keskosvauvojen varhaiseen kognitiiviseen kehitykseen. Yksi tärkeä näkökanta tällaisessa tutkimuksessa on analysoida emotionaalista sisältöä puheesta, jota keskoset kuulevat tehohoidon aikana. Puheen emotionaalisen sisällön manuaalinen analyysi on kuitenkin erittäin aikaavievää ja kallista. Täten olennaisesti tarvitaan automaattinen SER-systeemi laajamittaiseen puheen emootioanalyysiin.

Tässä tutkimuksessa tarkastellaan systeemiä, joka suorittaa SER:iä tosielämän lapsikeskeisille ääninäytteille vastasyntyneiden teho-osastolta. Tyypillisesti tällaista ongelmaa voitaisiin lähestyä ohjatun koneoppimisen menetelmin, mikäli riittävä määrä annotoitua opetusdataa on saatavilla. Tutkimuksen pääaineisto eli teho-osastonauhoitteet sisältävät kuitenkin satoja tunteja äänimateriaalia, joten aineisto on aivan liian suuri manuaalisesti annotoitavaksi. Tämän vuoksi vaihtoehtoisia koneoppimisen lähestymistapoja vertailtiin tutkimuksessa. Nämä lähestymistavat olivat ristikorpusopetus, k-medoids -klusterointialgoritmiin perustuva aktiivinen oppiminen (AL, Active Learning) sekä Wasserstein-generatiiviseen kilpailevaan verkostoon perustuva määrittelyjoukon adaptointi (DA, Domain Adaptation).

Koska tutkimuksen teho-osastonauhoitteista puuttuivat aluksi annotaatiot ja nauhoitteiden manuaalinen annotointi on erittäin työlästä, simulaatioita suoritettiin neljällä jo olemassa olevalla SER-korpuksilla jotta saataisiin selville, että mikä olisi parhain lähestymistapa kehittää SER-järjestelmää annotoimattomalle korpukselle. Tämän jälkeen osa teho-osastonauhoitteista annotoitiin ja näitä annotoituja nauhoitteita käytettiin arvioimaan simulaatioiden avulla saatujen löydösten toimivuutta käytännössä.

Tutkimuksen kokeissa DA-metodi suoriutui paremmin kuin ristikorpusopetus tapauksissa, joissa annotoitua dataa ei ole saatavilla kohdekorpukselle. Kohtalaisella annotoinnilla AL-metodi oli parempi kuin DA-metodi valenssin luokittelussa kun noin 4% teho-osastonauhoitteista oli annotoitu. Samalla määrällä annotaatioita DA-metodi suoriutui kuitenkin hieman paremmin kuin AL-metodi virittävyyden luokittelussa. Valenssin binäärisessä luokittelussa parhaiten suoriutunut koneoppimismalli oli AL-metodia hyödyntävä tukivektorikone, jonka luokittelutarkkuus oli 73.4% UAR (engl. unweighted average recall). Vastaavasti virittävyyden binäärisessä luokittelussa parhain koneoppimismalli oli DA-metodia hyödyntävä neuroverkkopohjainen luokitin, jonka tarkkuus oli 73.2% UAR.

Avainsanat: puheen tunteiden tunnistus, paralingvistinen puheenkäsittely, aktiivinen oppiminen, määrittelyjoukon adaptointi, LENA-nauhuri

PREFACE

The research work for this thesis was conducted at the Speech and Cognition research group at the Unit of Computing Sciences at Tampere University and the Department of Signal Processing and Acoustics at Aalto University. This work was partially supported by Academy of Finland project *Rhythms of Infant Brain* (RIB; project no. 314573 and 335872). When my supervisor, Prof. Okko Räsänen, presented me the current topic of my thesis as one possible option, it immediately struck me as a lightning to choose this topic without a doubt. I was born prematurely, so there was something special about writing a thesis which was related to my personal background. I would like to express my sincere gratitude to Okko for all the guidance and support he has given me throughout the entire project, and for constantly bringing new ideas to the table. I have already lost count of the number of times he has given me valuable advice during the project. All other members of the Speech and Cognition research group also deserve a big thank you for creating such a friendly atmosphere to work in.

I would also like to thank everyone else involved in the project, particularly those from the University of Turku. In addition, I would like to thank Prof. Joni Kämäräinen for indirectly guiding me towards studying signal processing and machine learning. Now that I come to think of it, no other major would have been a better fit for me. Furthermore, I would like to thank Prof. Tuomas Virtanen for all the great work he and his research group have done, and for the way he has led the Audio Research Group at Tampere University. Without active learning and domain adaptation methods developed by his research group, the present thesis would not be at the level where it is now.

Last but not least, I would like to thank all my family and friends for the support I have received during the project. Most sincerely, I would like to express my gratitude to Siru Peltoniemi for inspiring and motivating me on a daily basis throughout the entire project. I cannot even imagine how someone can provide me with such a vast amount of energy she gives me each and every day.

Tampere, 22nd April 2021

Einari Vaaras

CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

$\| \|$	Euclidean norm
$\langle a, b \rangle$	Inner product between elements a and b
$\mathbb{R}$	The set of real numbers
$\mathbb{Z}$	The set of integers
$\times$	Cartesian product
j	Imaginary unit

AL	Active Learning
APPLE	Auditory Environment by Parents of Preterm Infant; Language Development and Eye-movements
ASC	Acoustic Scene Classification
CNN	Convolutional Neural Network
DA	Domain Adaptation
DNN	Deep Neural Network
eGeMAPS	The Extended Geneva Minimalistic Acoustic Parameter Set
f-SPL	f-Similarity Preservation Loss
FNN	Feed-forward Neural Network
GAN	Generative Adversarial Network
GeMAPS	The Geneva Minimalistic Acoustic Parameter Set
GMM	Gaussian Mixture Model
GRU	Gated Recurrent Unit
GW	Gestational Weeks
HMM	Hidden Markov Model
KCCA	Kernel Canonical Correlation Analysis
LLD	Low-level Descriptor
LReLU	Leaky Rectified Linear Unit
LSTM	Long Short-term Memory

MAL	Medoid-based Active Learning
MLP	Multilayer Perceptron
MSE	Mean Squared Error
NICU	Neonatal Intensive Care Unit
NLP	Natural Language Processing
PCA	Principal Component Analysis
PSP	Paralinguistic Speech Processing
RBF	Radial Basis Function
ReLU	Rectified Linear Unit
RF	Random Forest
RNN	Recurrent Neural Network
SEM	Standard Error of the Mean
SER	Speech Emotion Recognition
STE	Short-time Energy
SVM	Support Vector Machine
UAR	Unweighted Average Recall
WDA	Wasserstein Distance-based Domain Adaptation
WGAN	Wasserstein Generative Adversarial Network

1. INTRODUCTION

The basic purpose of speech is to transmit a message using language [1, 2]. The formal structure of language is called the linguistic content of speech, which in turn consists of phonemes, words, and sentences. Speech is transmitted as acoustic waveforms produced by the human speech production system [2]. However, these acoustic waveforms contain much more information than only the linguistic content of speech, such as the speaker's personality, health state, attitude, speaking style, and the age of the speaker. Paralinguistic speech processing (PSP) refers to the digital analysis of speech beyond its linguistic content [1].

Perhaps the most well-known subcategory of PSP is speech emotion recognition (SER), in which the task is to recognize the emotional state of the speaker from an acoustic waveform [1, 3]. Although in some cases this might be an easy task for humans, machine-based automatic recognition of emotions is an ongoing subject of research in the PSP research community. Especially real-life audio recordings with various task-irrelevant characteristics such as noise and overlapping speakers have turned out to be difficult in PSP [4, 5].

Emotional speech is particularly interesting in the study of babies' cognitive development. Preterm infants can spend up to even four months at a hospital's neonatal intensive care unit (NICU) after birth. Since the baby is exposed to multiple environmental sources of stress during the intensive care, such as bright lights and noise, the stay at the NICU might negatively affect the early brain development of the child. To better understand the effect of parental proximity and communication on a child's development for prematurely born children, there is an ongoing joint research project conducted by Turku University Hospital and Tallinn Children's Hospital called *Auditory environment by Parents of Preterm infant; Language development and Eye-movements* (APPLE) [6]. The fundamental purpose of this thesis is to contribute to this research project by creating a system which performs SER as accurately as possible for hundreds of hours of real-life child-centered audio recordings collected from a NICU during this project. An automatic system that is capable of performing emotion analysis for recordings like these would help vastly in the study of how different emotional environments affect a child's cognitive development. For example, one hypothesis might be that a preterm infant, whose audio environment at the NICU mainly consists of speech with positive emotions, would be more likely to

have faster cognitive development later on than those infants whose audio environment consists of less positive emotions. Furthermore, in addition to the scientific study of child development, a functioning SER system could be utilized for intervention studies aiming to optimizing neonatal care [7]. Without an automatic SER system, the large-scale analysis of the emotional content of real-life audio recordings would be extremely expensive and time-consuming.

Traditionally, training a SER system using supervised machine learning requires large amounts of labeled data. These labels are often manually acquired from human annotators. However, the size of the present audio dataset from the NICU that is going to be analyzed is very large, and is thus too expensive to be fully annotated. Therefore, alternative machine learning-based techniques such as cross-corpus generalization, active learning (AL), and domain adaptation (DA) are required to tackle the absence of labeled training data.

The main research goal of this thesis is to create a well-performing SER model for the real-life child-centered audio recordings from a NICU. As already stated above, SER with real-life recordings is a difficult task. Also, the absence of a fully annotated dataset raises the question of how to most effectively deploy a SER system to a novel domain, where effectiveness can be measured in terms of system accuracy and the amount of required human effort to develop and validate the system. To this end, cross-corpus generalization and state-of-the-art AL and DA methods are compared in the present experiments. These methods have rarely been compared to each other directly. Moreover, the unique nature of the dataset also provides an excellent opportunity to explore the application of SER to a challenging real-world use case, where SER can be utilized for the scientific study of child development. Furthermore, the present study is one of the few studies in which SER is applied to real-world large-scale data.

This thesis is organized as follows. Chapter 2 presents a review of the main concepts of the present study. A theoretical foundation for the core methods of the thesis is given in Chapter 3. These include the cross-corpus generalization, AL, and DA methods used in the present experiments. Chapter 4 describes the experiments conducted in the present study, followed by the presentation and discussion of the results of these experiments in Chapter 5. Finally, Chapter 6 provides a summary of the present experiments and the main findings of the study. Additionally, future research questions related to SER and possible future improvements for the present work are briefly discussed.

2. BACKGROUND

This chapter presents a review of the main concepts of this thesis. First, Section 2.1 gives a brief introduction to machine learning. Then, Section 2.2 introduces the basics of preprocessing in speech processing, discusses about PSP and SER in general, and provides a review of the previous work done in the field of SER. Next, Section 2.3 gives an overview of the main features used in the present study, while Sections 2.4 and 2.5 provide a detailed overview of the different classification models used in the present experiments. Finally, Sections 2.6 and 2.7 give an overview of AL and DA, respectively. The basic concepts of the topics are discussed, and there is also a review of the roles of AL and DA in SER.

2.1 Introduction to machine learning

Machine learning is a subcategory of artificial intelligence in which the aim is to construct computer programs or algorithms that are able to automatically improve their performance with experience [8]. A machine-learning algorithm constructs a *model* based on training data to perform some given task. This process is also known as the training process. A model in machine learning is simply the outcome of the training process for some machine-learning algorithm. This model can be considered as a mathematical function that produces some output based on its input values [8].

Classification in machine learning is the task of predicting a categorical value or a set of categorical values based on the input of the model [8]. For example, predicting whether an image represents a dog or a cat is a classification task. *Regression* is the task of approximating a real-valued target function [8]. An example of a regression task is predicting the height of a person given his or her shoe size.

For machine-learning algorithms to learn, some representation of the input data should be given as an input to the algorithm. This representation is known as the input *features*. The most basic training scenario in machine learning is *supervised learning*, in which the training data contains both the input features and their respective target labels [9]. If there are no target labels available, then the training scenario is called *unsupervised learning*. A combination of these two is called *semi-supervised learning*, in which the premise is that there is a small amount of labeled data available and a large amount of unlabeled

data available [9].

2.2 Paralinguistic speech processing

Paralinguistic speech processing (PSP), also known as *computational paralinguistics*, is a relatively new area of speech processing [1]. Approximately 30 years ago the field was practically nonexistent, and neither did the term exist 20 years ago. In this context, the word *computational* means simply that something is done by a computer. The word *paralinguistics* is essentially the most relevant word here. Its first part, 'para', originates from the Greek preposition $\pi\alpha\rho\alpha$, which means 'alongside something'. The latter part, 'linguistics', refers to the linguistic content of speech, including e.g., the phonetics, the grammar, and the semantics of speech. Thus, PSP means digital speech processing where we are interested in analyzing or recognizing the way something is said instead of what is being said [1, 4].

The rest of this section is structured as follows. First, the basics of pre-processing in speech processing are introduced in Section 2.2.1. These pre-processing principles apply to practically any speech processing task, including PSP. Then, Section 2.2.2 focuses on the special characteristics of PSP that are different compared to other types of speech processing. Throughout the entire section, the book *Computational Paralinguistics: Emotion, Affect and Personality in Speech and Language Processing* by Schuller and Batliner [1] will be extensively referred to. Their book is the first, and so far the only, book which provides a comprehensive review of PSP, and is commonly used as a standard reference for PSP. Finally, Section 2.2.3 reviews the previous work done in the field of SER, which is a special case of PSP.

2.2.1 Pre-processing in speech processing

In speech processing, *windowing* is commonly the initial step in feature extraction. In windowing, a digital speech signal is split into short segments, also called audio frames, in which the signal is assumed to be stationary [2]. A signal, in turn, is a physical representation which carries data from one point to another [10]. The purpose of windowing is to split the time-varying speech signal into shorter segments within which the properties of the signal stay constant.

When a digital speech signal is split into short segments, the borders of the segments are discontinuous. Typically, to counter the effect of discontinuity, the segments are multiplied with a smooth windowing function that emphasises the values at the center of the segment and suppresses the values at the borders, i.e. the windowing function goes towards a small value or zero at the borders [2]. For features in the time domain, rectangular windows can also be used. However, the main problem with rectangular windows in the

time domain is that they produce a spectral leakage in the frequency domain due to large side lobes. Consequently, these side lobes may cause unwanted effects in the frequency domain, such as a ringing effect caused by a sinc function (the frequency response of a rectangular window) [2].

By using a smooth windowing function, the discontinuities near the borders of the windowed segments become negligible. The signal values outside of the frames can either be regarded as zero (stationary approach) or undefined (non-stationary approach) [1]. The window length should be determined so that it is long enough to model the desired property of the signal, but on the other hand, short enough for the signal to be stationary within the window. Voiced speech is often regarded as a *quasi-periodic* signal, which means that the signal is assumed to be periodic within a small time frame [2]. Additionally, the adjacent windowed frames are overlapped in order not to lose information within the signal. Typical window lengths in speech processing are 20–40-ms in length with time shifts of 10 ms [2]. Common windowing functions are Hamming and Hann windows because they both have the desired properties for analysis: they decay rapidly in the time domain, but also have a narrow and rectangular spectrum in the frequency domain [2]. Figure 2.1 demonstrates a 30-ms segment of a digital speech waveform and its Hann-windowed version along with the Hann window.

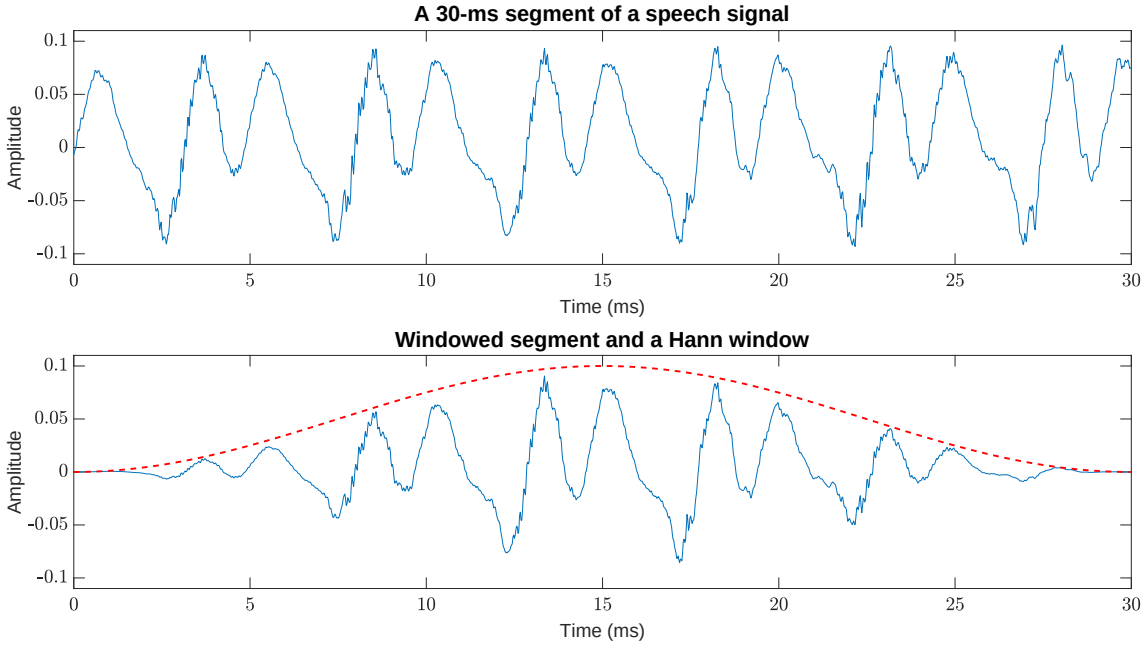


Figure 2.1. *A 30-ms segment of a speech signal corresponding to a vowel sound (upper image) and its Hann-windowed version along with a Hann window (lower image).*

After the speech signal is windowed, it is typical that frame-level acoustic features, also known as acoustic low-level descriptors (LLDs), are extracted from the windowed frames. Here the underlying assumption is that the signal properties of interest are constant within the frame of analysis, and that the feature of interest evolves at a slower rate than the rate

at which the adjacent overlapping windowed frames are located in the signal. This is also known as the short-time analysis of speech [2]. Typical frame-level features include mel-frequency cepstral coefficients (MFCCs), fundamental frequency (f_0), linear predictive coding (LPC) coefficients, the short-time autocorrelation function (STACF), and the short-time zero-crossing rate (STZCR) [2]. In addition, derived features of LLDs can be computed. These include e.g. first and second order delta features (see Section 2.3.1 for further details), filtered versions of LLDs, and LLDs with some nonlinear function applied [1, 2].

2.2.2 Fundamentals of paralinguistic speech processing

In PSP, an important concept is the distinction between speaker states and traits. Although both terms can mean similar things, in PSP traits refer to longer-lasting or permanent properties of the speaker, whereas states are shorter-term characteristics. Any characteristics with a duration of something between these longer-term traits and shorter-term states have been defined by Schuller and Batliner [1] as *medium-term between traits and states*, which we refer to in this text as 'intermediate straits'. Table 2.1 demonstrates examples of the three aforementioned cases of different time scales for paralinguistic phenomena. A common task for a PSP system is to either analyze, classify, or detect some paralinguistic phenomenon or phenomena (e.g. the examples in Table 2.1) [1]. When designing a model for a PSP task, the time scale of the paralinguistic phenomenon of interest should be taken into account. For example, a model for classifying personality should somehow exploit the fact that the time scale of personality-related phenomena in speech is long-term. Therefore, temporal dependencies related to personality in speech cannot be modeled using models that only consider a short time scale. The basic assumption when designing a PSP model is that the phenomenon of interest is constant for the entire time scale of analysis [1].

Typically, similarly to a classical machine-learning system, a PSP system consists of two separate parts; feature extraction and a PSP model [1, 4]. The first part, feature extraction, converts a digital speech signal into some feature representation. The second part, the PSP model, then performs classification or regression regarding the paralinguistic phenomenon of interest by means of supervised machine learning [1, 4]. More recently, however, it has become increasingly popular to use end-to-end PSP systems which combine these two parts. This can be achieved by using deep neural networks (DNNs) that are able to learn task-specific features directly from the training data while simultaneously training the DNN-based PSP model [11].

Since paralinguistic phenomena occur over time scales that are longer than typical features in speech processing, the features and models in PSP should also correspond to a time scale that is longer than frame-level. This time scale can range from the level of a

Table 2.1. *Examples of different paralinguistic phenomena divided into three different time scales [1, 4].*

Type	Subtype	Examples
long-term traits	biological trait primitives	height, weight, age, gender
	cultural trait primitives	race, culture, social class
	personality traits	personality, likeability
intermediate straits	(partly) self-induced more or less temporary states	sleepiness, intoxication, health state, mood
	structural signals	role in groups, friendship, attitude, interest, politeness
	discrepant signals	irony, sarcasm, lying
	mode (can also be long-term or short-term)	speaking style, voice quality
short-term states	emotional states	emotion, valence, arousal
	emotion-related states or affects	stress, confidence, uncertainty, frustration, pain

single utterance up to days or even longer periods of time [1]. Additionally, paralinguistic information is often hidden in the way LLDs evolve over time, and not in the individual frame-level features. Hence, there is a need for *suprasegmental* features (not to be confused with suprasegmentals in phonetics) which accumulate information over multiple frames. Moreover, as PSP deals with non-linguistic information, redundant information such as the actual linguistic content of the speech is reduced when using suprasegmental features [1].

By far the most common solution for obtaining suprasegmental features in PSP is to apply *functionals* to the time series of frame-level features [1, 4, 5]. Functionals are mathematical operations which map a time series of arbitrary length into a single value. These include e.g. mean values, statistical n^{th} order moments, extreme values, the range of the signal, percentile values and percentile ranges, and regression coefficients [1]. The functional mapping of a time series into a single value has the desirable property that audio samples of different lengths become mapped into constant-length feature vectors. This simplifies further analysis and processing of the features [1]. However, some information is lost when applying functionals to the data. Although this is often desirable for long audio signals, on some occasions the loss of information has to be minimized. In these cases, feature stacking and temporal models can be used, for example. In feature stacking, adjacent feature frames are stacked together to form a long feature vector. The main issue with this is that all input waveforms must have the same length in order to have a constant-length output. Moreover, feature stacking is not suitable for segments longer than a couple of seconds since otherwise the output feature vector will get too large for practical use cases [1]. More recently, temporal models like recurrent neural

networks (RNNs) have become more common in PSP [11]. These models are able to process longer-term information than by using feature stacking without reduction in information, but on the other hand they tend to require more training data. Additionally, RNN models are able to handle inputs of varying lengths which is a useful property for PSP [1, 11]. An example of a conventional feature extraction pipeline in PSP is demonstrated in Figure 2.2.

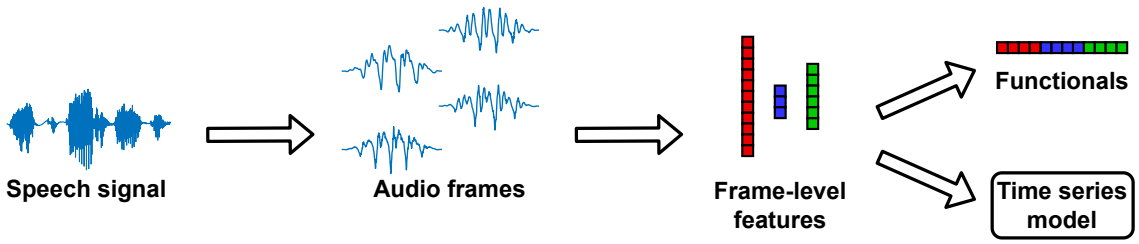


Figure 2.2. *An example of a typical feature extraction pipeline in PSP. First, a digital speech signal is split into audio frames using a windowing function. Then, one or multiple frame-level features are extracted from the windowed frames. Finally, either the frame-level features are directly used as an input to a time series model, or functionals are applied to the frame-level features to produce a constant-length feature vector.*

Selecting an optimal set of features for a PSP system is not a trivial task. The selection of features depends on the PSP model to be used, the amount and quality of the training data, and the knowledge of the task at hand [1, 5]. With much data available, DNNs can be utilized to learn a suitable feature representation directly from windowed segments or even from raw acoustic waveforms. These methods can be further improved if domain knowledge of the task at hand is available, e.g. by using some more advanced features as the initial features for the model [12]. However, this is not typically the case in PSP, where often high-quality annotated data is scarce [1, 11]. In this case, the option is to either utilize expert knowledge or to extract a large set of features that are not specific to the task at hand. By utilizing expert knowledge, it is possible to find acoustic and linguistic features that are relevant for a specific PSP task. However, so far nobody has found the perfect features for any PSP task, and often the ones that have been defined may not be easy to extract in practice in a systematic manner [1]. Thus, the most common method in PSP is to extract a large number of features that are not tailored for the specific task [1, 5, 13]. These features are chosen so that they can be extracted systematically, and are typically extracted using a feature extraction toolkit, such as openSMILE[1]. As a comparison, it is common to compute only one type of frame-level features for some given application in typical speech processing [2].

Generally, compared to other speech processing tasks, feature vectors in PSP are high-dimensional [1]. Large feature spaces lead to, among other things, more complex models which are prone to overfit the data, make the optimization algorithms more complex, and

[1] Open-source Speech and Music Interpretation by Large-space Extraction; for further reading, see [14].

thus lead to more extensive computation. Therefore, a common solution is to perform feature selection or to use a classifier that is robust to non-significant features [1, 15]. An example of such a classifier is the support vector machine (SVM) [1], which has the desired property of being able to handle high-dimensional feature spaces and can also withstand overfitting. These are some of the reasons why SVMs are among the most popular classifiers in PSP [1, 5]. Other common classifiers in PSP include random forests (RFs) [13] and neural network-based approaches, such as convolutional neural networks (CNNs), multilayer perceptrons (MLPs), RNNs, and combinations of these [1, 11]. Furthermore, other classifiers like hidden Markov models (HMMs), Gaussian mixture models (GMMs), and k-nearest neighbors (k-NN) have been used to some extent [5, 13, 15]. Out of all the classifiers in PSP, the neural network-based approaches have gained popularity in recent years [5, 11].

In general, many PSP tasks have turned out to be difficult, even for state-of-the-art machine-learning models[2] [5, 11, 13]. One of main bottlenecks in developing PSP models is the lack of high-quality annotated data for PSP tasks, which many modern DNN-based machine-learning models heavily depend on [11]. There are many reasons for the absence of annotated data in PSP. First, data collection in PSP may involve private or sensitive information within the recorded test subjects. This is also one of the reasons why many PSP corpora are not freely accessible within the research community, but instead, access to many PSP corpora is highly restricted [1, 4, 11]. Second, data collection for PSP tasks may be challenging, since many paralinguistic phenomena, such as personality, can be difficult to capture into a large-scale dataset in a systematic manner [4, 5]. Third, PSP corpora can be difficult and time-consuming to annotate, since many paralinguistic phenomena may not be transparent even for human experts [4, 5, 11]. To alleviate the lack of large-scale annotated PSP corpora which hinders the development of PSP models, multiple solutions have recently started to emerge in the field of PSP. These include e.g. crowdsourcing, AL, DA, pretrained DNN models, reinforcement learning, and utilizing synthesized speech [11].

2.2.3 Speech emotion recognition

The categorization of speech into different emotions, also known as speech emotion recognition (SER), is a PSP task which has interested researchers for years [3]. The automatic processing of emotions in speech started to evolve in the mid-nineties together with the development of the field of PSP in general. At first, only a few acted basic emotions were included in the studies, but later on, the analysis of more realistic portrayals of several emotions has become more prevalent [3, 16].

[2] Since 2009, different PSP tasks have been discussed annually in challenges held at INTERSPEECH conferences (`http://www.compare.openaudio.eu/`), which are the world's largest speech-related technical conferences.

Many of the common properties of PSP described in Section 2.2.2 apply in SER. Similar to PSP systems, a SER system is commonly constructed using supervised machine learning to either assign an input speech signal into one of predefined categorical emotional labels (classification), or to predict a continuous value on some predefined emotional scale (regression) [1, 3]. An example of classification in SER could be a case where a model predicts whether the emotion of a given utterance is either 'joy', 'sadness', 'anger', or 'neutral'. A regression task in SER could be, e.g., a case where a model predicts a value in the range from 0 to 1 for some given utterance, where the value 0 means that the expressed emotion is negative, and the value 1 means that the expressed emotion is positive.

The number of emotional categories in different SER corpora varies largely based on the intended use case of the SER corpus [3]. Commonly, since SER is in itself a difficult task even for human experts, less than 10 different emotional categories are used in SER corpora. When SER corpora are created, the assignment of audio samples into different emotional categories is usually conducted by human annotators [1, 3, 17].

In 1980, a psychologist named James A. Russell introduced a circular model for representing emotions [18]. He proposed that all emotions can be mapped into a two-dimensional plane with valence as one axis and arousal, also called activation in many studies, as the other axis. Valence is a measure of positive and negative affectivity, or in other words, pleasantness and unpleasantness, whereas arousal measures how calming or exciting the spoken information is [18]. This mapping of emotions into a valence-arousal plane has been used in a vast number of SER studies (e.g. [17], [19], [20], [21], [22], [23], [24], [25]) to harmonize differences between the emotional labels of different corpora. Indeed, some corpora have even only provided emotional labels in the valence-arousal plane. It should be noted, however, that mapping emotions into a valence-arousal plane is not a trivial task since no comprehensive one-to-one mapping has yet been defined [3, 4, 16, 17].

Besides unifying the divergence between corpora, mapping emotions into a valence-arousal plane has been commonly used to reduce the complexity of a given SER task by both reducing the number of possible classes and making the given classes more distinct from each other [16, 17]. Furthermore, classification in this plane can be regarded as two binary decisions if discrete emotional labels are considered. This works well with binary classifiers such as SVMs, which have been popularly used in SER [4, 5, 16]. Studies such as [5] and [26] have demonstrated that arousal is typically easier to classify than valence. However, as e.g. Schuller et al. [17] pointed out, for some SER corpora the classification of arousal is easier than it is for valence.

As with other PSP tasks in general, most phenomena related to SER are expressed in the way LLDs evolve over time. Emotions in speech are regarded as short-term states (see

Table 2.1), and the time scale of emotions in SER is commonly at the level of utterances (i.e., ranging from less than a second up to even tens of seconds) [3]. Thus, suprasegmental features, which accumulate information across multiple audio frames, are commonly used in SER. By far the most popular features in SER have been high-dimensional feature vectors that are not specifically tailored to the SER task, but instead are meant to capture properties of speech signals as diversely as possible [5, 14]. Studies such as [27], [28], and [29] utilized a 6373-dimensional feature set that was used as the standard feature set in the INTERSPEECH computational paralinguistics challenges from 2013 to 2017. Schuller et al. [19] and Zhang et al. [20] used 6552-dimensional suprasegmental features that consisted of 39 functionals of 56 LLDs. Another frequently occurring feature set in SER is the 384-dimensional INTERSPEECH 2009 emotion challenge baseline feature set, which has appeared in e.g. [21], [30], and [31].

Two minimalistic sets of features, the Geneva Minimalistic Acoustic Parameter Set (GeMAPS), and its extended version, the extended Geneva Minimalistic Acoustic Parameter Set (eGeMAPS), were proposed by Eyben et al. [32] as an attempt to unify features in the field of affective computing, including SER. Since then, studies such as [24], [26], and [33] have used the proposed features as baseline features. The GeMAPS and eGeMAPS features have provided performance that is comparable to and even better than large feature vectors that are not tailored for a specific task [32]. These features are discussed in further detail in Section 2.3.2.

In the past few years, more advanced features than those of the large suprasegmental feature vectors have become increasingly popular in SER. Many of these methods involve learning a task-specific feature representation in conjunction with training a SER model. Trigeorgis et al. [26] presented the first fully end-to-end SER model which utilized CNN and bidirectional long short-term memory (LSTM) layers to convert raw speech waveforms into predicted emotions. Cummins et al. [33] exploited a pretrained deep CNN, originally meant for image recognition, to convert spectrograms into a 4096-dimensional feature representation. Chen et al. [34] proposed a 3-D attention-based convolutional RNN that converted a log-mel spectrogram and its first and second order delta features directly into a predicted emotion. Etienne et al. [35] used a DNN approach where they converted log-spectrograms into high-level features by using four CNN layers and one bidirectional LSTM layer. Zhao et al. [36] processed 743-dimensional frame-level features using a fully convolutional network and a bidirectional attention-based LSTM network side by side. These processed features are then concatenated and fed to a fully connected network for predicting emotional labels. Zhang et al. [37] input log-mel filterbank energies into a temporal CNN model to predict soft emotional labels.

Since the most common features in SER have been large feature vectors which presumably contain many irrelevant elements, it seems natural that SVMs are the most popularly used classifiers in SER due to their properties (see Section 2.2.2 for a list of these prop-

erties) [4, 5, 16]. More recently, however, neural network-based models such as MLPs (e.g. [27], [31], [38]), CNNs (e.g. [33], [37]), RNNs (e.g. [22]), and combinations of these (e.g. [26], [34], [35], [36]) have become more and more widespread in the SER research community. Other commonplace classifiers in SER include GMMs, RFs, and naïve Bayes classifiers [5, 16].

Often, a SER system is intended to be applied in situations or conditions that are new to the system [3, 5, 16]. This not only considers new speakers from the same language that the system was trained on, but occasionally also speakers in different languages and varying recording conditions. In order to provide a classification model that is capable of performing emotion recognition with sufficient reliability over unseen samples, the model should not learn speaker-dependent or corpus-specific properties [3, 17]. Instead, an optimal SER model should solely learn emotion-related dependencies between the features and their corresponding labels. However, learning only emotion-related dependencies has proven to be a difficult task [16, 17].

Furthermore, emotions are portrayed differently across corpora and cultures. One might not even have a SER corpus available for a specific language. Thus, a common solution to create a well-generalizing SER model and to test its generalizability across e.g. different cultures, speakers, and recording conditions is to train the model on one or multiple SER corpora and test the model on some other SER corpus or corpora [17, 19]. This *cross-corpus generalization* SER setting has been examined in multiple studies (e.g. [17], [19], [20], [37]). Schuller et al. [17] have conducted the most extensive SER study regarding cross-corpus generalization thus far. They performed intra- and inter-corpus experiments using six frequently-used SER corpora of various languages, emotions, and test setups. Their experiments involved four different normalization strategies and different numbers of emotional classes. The study showed that reliable real-life emotion classification, let alone classification above chance level, was only feasible with certain corpora and only with certain emotional classes, even with corpora of similar cultural backgrounds. Their research also highlighted issues of SER corpora and cross-corpus emotion recognition at that time, of which many are still present today.

Schuller et al. [19] studied a cross-corpus generalization SER setting with the emphasis on comparing majority-voting between classifiers trained on a single dataset and training a classifier using a combination of multiple datasets. Their findings indicated that, on average, it is more beneficial to train a single classifier on multiple datasets than to use multiple separately trained classifiers, although the classification results varied considerably depending on the classifier that was used. Zhang et al. [20] used six different datasets as test sets and ten arrangements of labeled and unlabeled training sets for each test set to evaluate unsupervised learning on cross-corpus SER. In the study, three different normalization strategies were investigated. They discovered that adding normalized unlabeled data to agglomerated multi-corpus data enhanced classification performance.

This increase in performance was found to be approximately 50% of the performance increase if labeled data was added. Zhang et al. [37] proposed a family of loss functions called f-similarity preservation loss (f-SPL) for soft labels which are meant to preserve label similarities in a learned feature space. They combined f-SPL and cross-entropy classification loss and demonstrated in cross-corpus SER experiments that their method significantly outperformed a reference method which only utilized classification loss.

Only a few SER studies have been conducted on large-scale datasets. Jia et al. [39] randomly select 3000 utterances from a corpus of large-scale internet voice data containing a little under seven million utterances for manual annotation. Next, they proposed two novel methods for the emotion recognition task, a deep sparse neural network and a bidirectional LSTM, both of which were pretrained with 90,000 unlabeled utterances by using an autoencoder in an unsupervised manner. Then, these two methods were applied to the annotated utterances. Their experiments revealed that both proposed methods outperformed traditional SER models. When comparing the two methods, the bidirectional LSTM was more accurate than the deep sparse neural network at the cost of a notably longer training time. Fan et al. [40] presented a large-scale SER dataset with a little over 147,000 utterances from 820 subjects with a total duration of over 200 hours. They proposed a novel SER model containing pyramid convolutions which outperformed other models that were tested on the dataset. Additionally, they showed that existing models are prone to overfit to small-scale datasets which limits the ability of these models to generalize for real-life data.

When creating a novel SER corpus, the basic requirements of a good SER corpus include a large enough number of samples, a balanced distribution of different emotional categories, a large number of speakers, and an unequivocal distinction between emotional categories [1, 3]. Additionally, the corpus should represent the actual application of the SER system for which the system is intended to be used for. For example, a classification model which is trained using a SER corpus recorded in a clean recording environment is unlikely to perform well in a real-life noisy environment such as a crowded city street. Nevertheless, obtaining such high-quality annotated SER data that fulfills the basic requirements of a good SER corpus has turned out to be extremely difficult [1, 3, 16].

First of all, gathering large quantities of SER data for realistic applications is in itself troublesome. Not only is the data collection process time-consuming and expensive, but it is also extremely challenging or tedious to acquire data that is fully representative of the application that the corpus is intended to be used for [3, 16]. Besides, the frequency of emotional expressions from different emotional categories is highly imbalanced in real-world data for SER systems to be trained properly. It has been reported that neutral speech can account for over 90% of realistic speech content [3]. A common way to tackle the imbalance in the distribution of emotional labels is to use actors with acted emotions in the data collection process. This is not an optimal solution, since it is

generally acknowledged that one cannot model natural emotions adequately using acted emotions because of the different way that emotions are portrayed in acted speech [17, 41, 42, 43]. Also, speech in acted corpora is often not as diverse as in corpora containing realistic speech. Consequently, many SER studies show somewhat optimistic results since the corpora involved are using acted emotions instead of realistic portrayals of emotions [3, 16, 17, 40].

Second, there is no universal way of annotating emotions. It is not only that emotions can be expressed and perceived differently by people from different cultures, but also by people from the same culture. This results in the fact that utterances with the same emotional label between two distinct SER corpora can be very different from each other [1, 3, 17]. For example, utterances with the label 'neutral' might be restrained in one corpus and very lively in another. Furthermore, in many cases a ground truth label cannot be unequivocally determined, and often the agreement rates between different annotators, especially in realistic SER corpora, can be low even with domain experts [1, 3]. Again, resources are also a major limiting factor in the sizes of high-quality SER corpora, both real and acted, since the annotation process of emotions is both costly and laborious [3, 17, 27].

Third, as with other PSP corpora, data collection for SER tasks can involve private information within the test subjects. For instance, daylong audio recordings from real environments may contain sensitive information in the discussions of the participants. Hence, it is typical that realistic SER corpora are not freely distributable in the research community [3, 4, 16]. This considerably restricts the number of use cases of realistic SER datasets, resulting in many studies using freely accessible corpora which contain acted emotions despite the research community being well aware of their limitations [17, 41].

To conclude, SER is a subcategory of PSP for which many of the common PSP-related properties (Section 2.2.2) also apply. For instance, similar features and classifiers have been used in both PSP and SER. One of the key elements of what makes SER a challenging task is that it is difficult to set a clear threshold after which an emotion changes from one to another [3]. In addition, what makes SER even more challenging is that emotions can be expressed and perceived differently by individuals. To simplify a given SER task and to harmonize differences between SER corpora, emotions can be mapped into a two-dimensional valence-arousal plane [16, 17]. This mapping of emotions has been used in multiple studies related so SER (e.g. [19], [20], [21], [22], [23], [24], [25]).

2.3 Feature extraction methods

In machine learning, *feature extraction* is the process of converting data into some other representation. The main purpose of this representation is to make the actual task of the machine-learning model easier by removing information from the data which is not

relevant for the specific task [9]. For example, an acoustic speech signal contains a plethora of information which is not relevant to the majority of speech processing tasks, such as speaker- and recording device-dependent characteristics. Additionally, an ideal feature representation not only removes redundant information from the data, but also maximizes the informativeness of each sample regarding the task-at-hand [9, 44]. This section provides an overview of the main features used in the present study.

2.3.1 Log-mel

The log-mel spectrum is nowadays perhaps the most commonly-used feature in all audio analysis systems [44]. The procedure to obtain a log-mel spectrum of a digital signal begins with extracting the magnitude of the discrete short-time Fourier transform (STFT) of a windowed signal, i.e.

$$|STFT(t,\omega)| = \left| \sum_{n=0}^{N-1} x(n)w(t-n)e^{-j\omega n} \right|, \tag{2.1}$$

where t is the time instant of analysis, ω is the analysis frequency, $x(n)$ is the time domain signal, $w(n)$ is the windowing function, and N is the window length [2]. For the $|STFT(t,\omega)|$, a mel conversion of the frequency scale is applied using a mel-scale filter bank (a set of triangular filter responses in the magnitude domain with center frequencies that are evenly spaced on the mel scale). The mapping from hertz to mels is

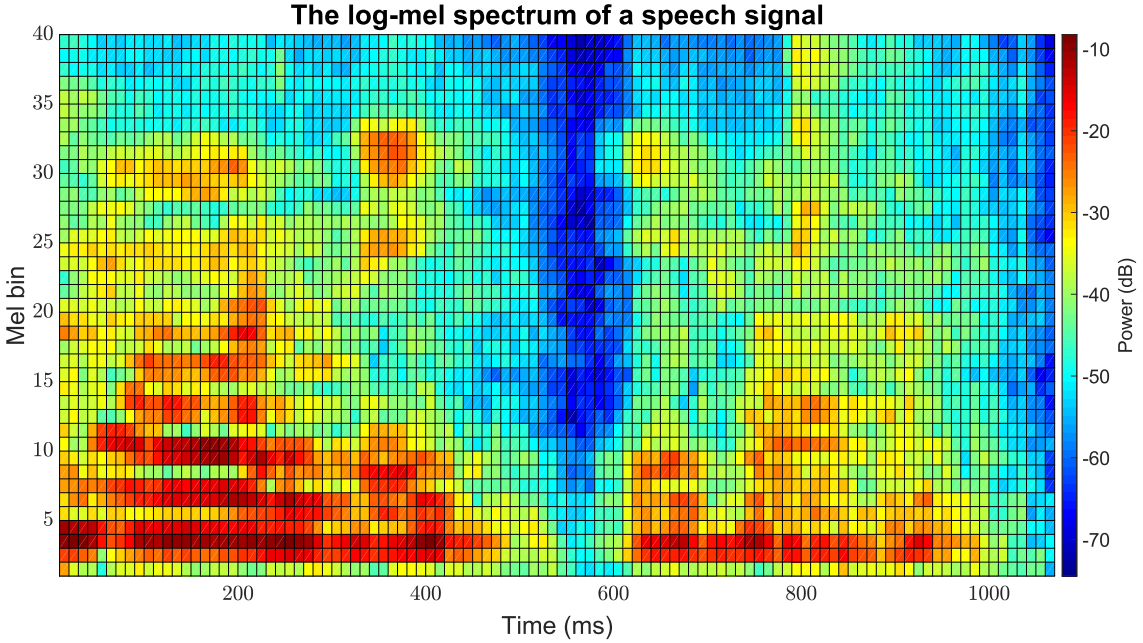


Figure 2.3. *The log-mel spectrum of a speech signal using 40 mel filters.*

defined as

$$mel(f) = 2595 \cdot log_{10}\left(1 + \frac{f}{700}\right), \tag{2.2}$$

where f is the frequency in Hz. Finally, after converting $|STFT(t,\omega)|$ into mels, the log-mel spectrum is obtained by taking the logarithm of the mel-filtered $|STFT(t,\omega)|$. The use of the nonlinear mel scale is motivated by the fact that the mel scale takes the human perception of sound into account, which makes it convenient for human-oriented audio classification tasks [2]. Figure 2.3 shows an example of a log-mel spectrum of a speech signal with a sampling frequency of 16 kHz using 40 mel filters.

It is typical to include first and second order time derivatives of log-mels (aka delta and deltadelta features) together with the original log-mel features. These delta features are used to account for temporal changes of the features between adjacent time frames. The first order delta features estimate the momentary evolution of the features, whereas the second order delta features estimate the rate of change of the features [44].

2.3.2 GeMAPS and eGeMAPS

A minimalistic set of features for various areas of voice analysis, the Geneva Minimalistic Acoustic Parameter Set (GeMAPS), was proposed by Eyben et al. in 2016 [32]. One of the main purposes of GeMAPS was to have a standardized feature set for voice research and affective computing for researchers working in various research areas. A standardized feature set helps, for example, in comparing different state-of-the-art methods, and also in combining and integrating different methods for voice analysis. Additionally, in contrast to the large brute-force feature sets that had been commonly used before GeMAPS in affective computing, the meaning of a small set of features is easier to interpret in a given task [32].

GeMAPS features were chosen in [32] according to three criteria:

1. The potential of an acoustic feature to index physiological changes in voice production during affective processes.
2. The frequency and success with which the feature has been used in the literature, as well as the automatic extractability of the feature.
3. The theoretical significance of the feature.

The GeMAPS feature set consists of prosodic, excitation, vocal tract, and spectral LLDs. To deal with different-length inputs, various functionals, such as the arithmetic mean and the standard deviation normalized by the arithmetic mean, are applied to the LLDs over the time dimension to get a feature output of constant length. After applying these func-

tionals, the GeMAPS feature set contains 62 parameters. An extension of GeMAPS, the extended Geneva Minimalistic Acoustic Parameter Set (eGeMAPS), adds functionals of cepstral LLDs to GeMAPS to better model affective states using a total of 88 parameters. For further details on the parameters and the functionals of GeMAPS and eGeMAPS, see Section 3 of [32]. The implementation to extract GeMAPS and eGeMAPS features is publicly available with the openSMILE toolkit [14].

2.4 Support vector machine

A support vector machine (SVM) is a non-probabilistic binary linear classifier. This classifier was first introduced by Boser et al. in 1992 [45] and it is based on the framework of the "Generalised Portrait Method" proposed by Vapnik and Chervonenkis in 1964 [46]. Their method was built on constructing a hyperplane which optimally separates the data points of two classes in the training data. In the case of SVMs, the optimal hyperplane is determined as the hyperplane which maximizes the margin between the two classes.

Following the SVM formulation of [47], let us have N linearly separable data points

$$(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N) \in \chi \times \{\pm 1\}, \qquad i = (1, \ldots, N), \tag{2.3}$$

where $\boldsymbol{x}_i$ are observations, y_i are their respective labels, and χ is a set containing all observations $\boldsymbol{x}_i$. For mathematical purposes, the two classes are labeled $+1$ and -1. A general hyperplane in some inner product space $\mathcal{H}$ can be written in the form

$$\langle \boldsymbol{w}, \boldsymbol{x} \rangle - b = 0, \tag{2.4}$$

where $\boldsymbol{w} \in \mathcal{H}$, and $b \in \mathbb{R}$. Among all of the possible hyperplanes in $\mathcal{H}$, there exists a unique optimal hyperplane, which maximizes the separating margin between any observation and the hyperplane. This optimal hyperplane can be found by solving the optimization problem

$$\underset{\boldsymbol{w} \in \mathcal{H},\ b \in \mathbb{R}}{\text{maximize}}\ \min\{\|\boldsymbol{x} - \boldsymbol{x}_i\| \ \mid\ \boldsymbol{x} \in \mathcal{H}, \langle \boldsymbol{w}, \boldsymbol{x} \rangle - b = 0,\ i = 1, \ldots, N\}. \tag{2.5}$$

By rescaling $\boldsymbol{w}$ and b so that the observations closest to the hyperplane satisfy the equation

$$|\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b| = 1, \tag{2.6}$$

we get a canonical form of the hyperplane, which satisfies

$$y_i(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b) \geq 1 \; . \tag{2.7}$$

Note that now the separating margin equals $\frac{1}{\|\boldsymbol{w}\|}$. Equation 2.7 can be split into two equations

$$\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b \geq 1 \quad \text{for} \quad \boldsymbol{y}_i = +1 \tag{2.8}$$

and

$$\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b \leq 1 \quad \text{for} \quad \boldsymbol{y}_i = -1, \tag{2.9}$$

which now enable the classification of unknown samples into the two classes. Now we can solve the optimization problem of 2.5 and construct the optimal hyperplane by solving

$$\text{minimize } \tau(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|^2 \text{ subject to } y_i(\langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b) \geq 1 \; . \tag{2.10}$$

The $\boldsymbol{x}_i$ that lie on the separating margin are called *support vectors*. An essential sidenote is that by following the Karush-Kuhn-Tucker (KKT) conditions of optimization theory, the optimal hyperplane is completely determined by its support vectors.

The aforementioned derivations hold only for linearly separable data points, whereas often the data points encountered in real-life machine-learning scenarios are not separable by a linear hyperplane [47]. To allow data points to violate the conditions in Equation 2.10, the objective function $\tau(\boldsymbol{w})$ is replaced by

$$\tau(\boldsymbol{w}, \boldsymbol{\xi}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} \xi_i \; , \tag{2.11}$$

where $C > 0$ is a penalty parameter, also known as the *box constraint*, and ξ_i are so-called slack variables. Now the SVM can be called a *soft margin* classifier since it no longer creates a clear threshold which separates the two classes perfectly. A classifier that generalizes well is obtained by adjusting the classifier's capacity with $\|\boldsymbol{w}\|$ and the sum $\sum_{i=1}^{N} \xi_i$. The box constraint C determines the trade-off between maximizing the class-separating margin and minimizing the training error [47].

In machine learning, feature spaces $\mathcal{H}$ typically require nonlinear class boundaries [47]. For these cases, Boser et. al. [45] propose mapping the data points into a higher-dimensional feature space Ω where target classes are linearly separable. This method is

popularly known as the *kernel trick*. With a suitable kernel, k, it is possible to compute inner products in higher-dimensional feature spaces without ever mapping the data points into that space. This can be achieved with k that are nonlinear in the original input feature space [9, 47]. For a class of kernels, k, which represent inner products in $\mathcal{H}$ through a mapping function Φ, i.e.

$$\Phi \colon \mathcal{H} \to \Omega, \quad x \mapsto \boldsymbol{x} \coloneqq \Phi(x), \tag{2.12}$$

we can represent a general kernel function as

$$k(x, x') = \langle \Phi(x), \Phi(x') \rangle \, . \tag{2.13}$$

There are multiple different kernel functions, the most popular being the Gaussian kernel, also known as the radial basis function (RBF) kernel. One of the reasons why the RBF kernel is popularly used is its universality in terms of its approximation capability [9, 47, 48]. The RBF kernel is defined as

$$k(x, x') = e^{-\frac{\|x - x'\|^2}{2\sigma^2}}, \tag{2.14}$$

where $\sigma > 0$. An interesting property of the RBF kernel is that Φ computes inner products in a feature space with infinite dimensionality. This follows from the property that the RBF Gram matrix has full rank and that there are no restrictions on the number of elements in χ [47]. Other popular kernels include the linear kernel $\langle x, x' \rangle$ and its extension, the polynomial kernel

$$k(x, x') = \langle x, x' \rangle^d, \tag{2.15}$$

where $d \in \mathbb{Z}^+$ [9, 47].

To avoid the inner product with some values of χ being dominant in the kernel computation, a kernel scale parameter, γ, is introduced. This parameter defines how far the influence of a single observation $\boldsymbol{x}_i$ extends. When applying this scaling parameter, all values of χ are divided by γ before computing the kernel mapping [47].

An SVM can be further extended into support vector regression (SVR), which was first presented by Drucker et al. [49]. Instead of the previous $y \in \{\pm 1\}$, we can have $y \in \mathbb{R}$ by introducing Vapnik's ϵ-insensitive loss function [47, 50]. Now, the loss can be determined

by predicting $f(x)$ instead of y as

$$c(x, y, f(x)) \coloneqq \max\{0, |y - f(x)| - \epsilon\}, \tag{2.16}$$

where y are the real observations, $f(x)$ are the predicted observations, and ϵ is a threshold parameter. No prediction $f(x)$ can be further than ϵ from y, i.e. a smaller value of ϵ results in the model being more sensitive to errors. To predict a linear regression

$$f(x) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b, \tag{2.17}$$

the objective function to minimize is now

$$\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} \max\{0, |y_i - f(x_i)| - \epsilon\} \ . \tag{2.18}$$

Although SVMs are inherently binary classifiers, they can also be extended into multiclass problems by combining several SVM classifiers [51]. The most popular multiclass SVM methods are “one vs. all” and “one vs. one”, where separate SVMs are either trained to discriminate one of the classes against the rest, or between every pair of classes, respectively [51].

2.5 Neural networks

This section gives an overview of neural networks. First, Section 2.5.1 provides an introduction and a general description of neural networks. Next, Sections 2.5.2, 2.5.3, and 2.5.4 describe three different commonly-used variants of neural networks, followed by a brief outline of deep learning and a review of some of the common practicalities related to neural networks in Section 2.5.5. Finally, Sections 2.5.6 and 2.5.7 describe two special types of use cases for neural networks.

2.5.1 General description

Neural networks, also known as artificial neural networks, are computational systems which consist of connected units called *artificial neurons*. These artificial neurons are mathematical models which try to mimic the biological neurons of real-life physiological nervous systems of vertebrates [1, 8]. Perhaps the most famous artificial neuron model is the perceptron, which Rosenblatt first presented in 1958 [52].

The perceptron is a binary classifier that takes an arbitrary number of real-valued inputs and provides a single output. Commonly, the output of a neuron is 1 if it is activated,

and 0 or -1 if activation does not occur. The output, also sometimes called the activation, is the weighted sum of the inputs combined with a bias term that does not depend on any input value. This bias term represents a permanent additive offset and is meant to shift the decision boundary of the classifier [8]. The bias term can also be considered as a threshold that the weighted combination of inputs must surpass for the neuron to be activated. To be more specific, the output y of the perceptron is

$$y(x_1, \ldots, x_N) = \begin{cases} 1, & \text{if } b + \sum_{i=1}^{N} w_i x_i > 0 \\ -1, & \text{otherwise} \end{cases}, \tag{2.19}$$

where b is the bias term, x_i are the inputs of the perceptron, w_i are the weights for the inputs, and N is the number of inputs [8]. The weights w_i represent the importances of the connections between neurons. A larger weight between two neurons implies a greater influence, whereas a smaller weight implies a smaller influence [8].

By combining perceptrons side-by-side into layers, and possibly stacking these layers one after another, a network of neurons is formed. If all of the connections in a network feed forward from one layer to the following layer without backward connections, i.e., connections do not form a loop, the network is called a feed-forward neural network (FNN) [8, 53]. An FNN treats every input pattern independently without any memory over time. Both the weights w_i and the bias term b in each neuron of the network are trainable. The network is trained iteratively by feeding training samples to the network and updating the trainable parameters of each neuron according to the output error of the network [8]. Commonly, it is not possible to train a network by inputting all of the training samples to the network at once, for example due to memory limitations. Hence, a neural network is often trained by feeding the network *batches* of data which are smaller than the size of the whole data. When all of the training samples have been fed to the network, one *epoch* has passed. Usually a neural network is trained for a number of epochs ranging from a few epochs up to even many thousands of epochs [8, 9, 53].

A major problem is that Equation 2.19 does not allow for the training of multilayer networks using the *backpropagation algorithm*, which is the most popular training algorithm of neural networks [8, 9]. In the backpropagation algorithm, the neural network is trained iteratively in two passes: a forward pass and a backward pass. In the forward pass, the training samples are fed to the network and the output of the network is computed. In the backward bass, the error of the network is first determined based on the output of the network using some *loss function* [8, 9]. An example of a common loss function is the

mean squared error (MSE) loss function, which is defined as [54]

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2, \tag{2.20}$$

where y_i is the true value, $\hat{y}_i$ is the predicted value, and N is the number of instances. Then, using some optimization algorithm, such as the stochastic gradient descent, the trainable parameters of the network are updated by iterating backward from the last layer to the first [8]. The parameters are modified by taking steps towards the negative of the gradient of the loss function with respect to each parameter. The size of these steps is called the *learning rate* of the network. Here, the chain rule is utilized by first computing the error gradients of one layer and then propagating the computed gradients backwards to the previous layer, one layer at a time [8, 9].

As pointed out by Minsky and Papert in 1969 [55], another major problem with perceptrons utilizing Equation 2.19 is the inability to perform the XOR operation [8, 9]. Thus, instead of using the traditional perceptron model, an alternative model is often used. This model is a neuron whose output is not only a nonlinear mapping, $a(x)$, of its inputs, but whose output is also a differentiable function. Now the output y of the perceptron is continuous-valued and can be written as [8]

$$y(x_1, \ldots, x_N) = a(b + \sum_{i=1}^{N} w_i x_i)\ . \tag{2.21}$$

This nonlinear mapping, $a(x)$, is often called the *activation function*. Common activation functions include the logistic (aka sigmoid) function [8]

$$a(x) = \frac{1}{1 + e^{-x}} \tag{2.22}$$

and the rectifier function

$$a(x) = max(0, x)\ , \tag{2.23}$$

which is also known as the rectified linear unit (ReLU) [56]. Nowadays ReLUs are one of the most popular activation functions used in neural networks because the error gradients propagate efficiently through multiple network layers when applying the backpropagation algorithm. This results from the fact that the derivative with a ReLU is always the same as long as it is positive, regardless of the value of the function [53]. In contrast, the sigmoid function saturates to a high value when the input of the function is very positive, and saturates to a low value when the input is very negative. A widespread saturation of

sigmoid units often makes gradient-based optimization difficult [53].

Variants of ReLUs include the leaky rectified linear unit (LReLU) [57] and the exponential linear unit (ELU) [58] nonlinearities. The LReLU nonlinearity, defined as

$$a(x) = \begin{cases} x, & \text{if } x \geq 0 \\ 0.01x, & \text{if } x < 0 \end{cases}, \tag{2.24}$$

allows for a small, non-zero gradient when the neuron is not active. This property can help a neural network to converge faster [57]. ELU nonlinearities, defined as

$$a(x) = \begin{cases} x, & \text{if } x > 0 \\ e^x - 1, & \text{if } x \leq 0 \end{cases}, \tag{2.25}$$

have proven to lead to faster learning and better performance than ReLUs in deeper networks [58].

2.5.2 Multilayer perceptron

A multilayer perceptron (MLP) is an FNN that consists of an input layer, an output layer, and an arbitrary number of hidden layers in between [8, 9]. Each layer consists of neurons that are, apart from the input layer, activated by an activation function such as 2.22 or 2.23. The name of the hidden layers originates from the fact that their output is only available within the network [8]. An MLP with a single hidden layer is, in theory, capable of modeling any nonlinear function to any desired degree of accuracy, given enough neurons. Hence, an MLP network is a universal function approximator [59]. However, this does not seem to work in practice since complex tasks can theoretically require an infinite number of neurons in a single hidden layer. In addition to the limitations of computational hardware, training neural networks with very large hidden layers has been found to be extremely difficult. Moreover, a more efficient approximation of any nonlinear function, i.e. the same level of accuracy with fewer neurons in the network, can be obtained by using more than one hidden layer [9].

MLP networks are sometimes called ‘vanilla’ neural networks, fully connected networks, or dense networks [8, 60]. The first of these terms originates from MLPs being the traditional neural networks used in literature (as vanilla is the traditional flavor used in ice-cream) [60]. The two other names are derived from the fact that the neurons in MLPs are densely connected in the network, i.e., each neuron is connected to all of the neurons in the following layer [8]. Sometimes the terms MLP and FNN are confused in the literature, although MLPs are a subcategory of FNNs [8, 60]. Another point worth mentioning is that

although the term MLP contains the word 'perceptron', an MLP network does not typically consist of the traditional perceptrons (Equation 2.19), but instead nonlinear neuron models are used. Furthermore, even though the name might suggest that an MLP consists of one or multiple perceptrons that have a multilayer structure, in fact the perceptrons themselves form multiple layers consisting of one-layered perceptrons [8].

2.5.3 Convolutional neural networks

Another type of neural network, convolutional neural networks (CNNs), were first presented by LeCun et al. in 1989 when they used a CNN model for handwritten digit recognition [61]. CNNs are designed to process data with a known grid-like topology, e.g. images (a 2-D grid for grayscale images and a 3-D grid for color images) and time series data (a 1-D grid), using M-dimensional convolutions, where M is the number of dimensions for the input data grid [53]. A typical CNN consists of two parts: an arbitrary number of convolutional layers and a fully connected network. The convolutional layers are meant to produce a high-level representation of the input features by combining several locally learned low-level feature representations, whereas the fully connected network is meant to process these high-level features to perform the actual task of the network, such as classification or regression. It is also possible to have a CNN without any fully connected layers. These are occasionally referred to as fully convolutional neural networks [53].

Typically, the convolutional layers consist of three phases [53]. In the first phase, several convolutions are performed in parallel to produce a set of linear activations called feature maps. This operation can also be considered as filtering the input with a number of convolutional kernels or filters. The trainable parameters of a convolutional layer are the weights and the bias term for each of the convolutional filters. In the second phase, each feature map is passed through a nonlinear activation function, the most common one being the ReLU. This nonlinearity is simply meant to transform the feature maps in a nonlinear manner in order to increase the expression ability of the CNN. In the third phase, subsampling takes place as the input dimensions are shrunk by an integer factor. This is done using a pooling function, such as max pooling [53]. In max pooling, the maximum value within a rectangular block is taken while the other values are discarded. Other common pooling functions are average pooling and weighted average pooling, which either take the average value or a weighted average value, respectively, within a rectangular block. Subsampling not only improves the spatial invariance of the model but also reduces the data size [53].

CNNs have been especially successful in image-processing tasks and have been used extensively in the field of image processing after their breakthrough in the early 2010s [33, 53]. Additionally, CNNs have performed well in other fields such as audio processing, financial data processing, and natural language processing (NLP). For example, in au-

dio processing, the good performance of CNNs in image processing can be exploited by taking into account that multi-dimensional audio feature representations, such as spectrograms, can be treated as image inputs for a CNN. Thus, CNNs have also been used increasingly in modern audio processing pipelines [11, 44].

The *receptive field* is the region of the neuron input that affects its output, typically in terms of spatial location on an image or the number of samples in a time series [53]. In convolutional layers, the receptive field of a neuron is a section of the previous layer, whereas in dense layers it is the whole previous layer. This reduces the number of excessive parameters in CNN models when processing grid-like data. This, in turn, makes the learning process faster and more efficient with large networks [53]. However, on many occasions, the receptive field can also be regarded as the region of the input data that affects a neuron's output [53].

In addition to reducing the number of excessive parameters, one of the main advantages of CNNs over MLPs for certain applications is that CNNs take the spatial structure of the input data into account [53]. MLPs, on the other hand, treat every input value independently. Consequently, the number of trainable parameters in dense networks is excessively large for spatially correlated data, such as images. This leads to MLPs being more prone to learn potentially task-irrelevant characteristics for spatially structured data, i.e., they are prone to overfit the data. For this reason, MLPs have been found particularly inconvenient in image-processing tasks since they do not scale well with large image sizes [53].

Other advantages with CNN models are that the convolution operations are spatially (or temporally) invariant. This is also known as translation invariance. This means that if some pattern is shifted in the input signal, it is also shifted equally in the output signal. For example in time series data, if an event is shifted n time steps later in time in the CNN input, the exact same output representation of this event will appear n time steps later in the CNN output [53]. Additionally, data size is reduced in the subsampling process, which results in CNN models being less prone to overfit the data by decreasing the number of free parameters in the model. However, it should be pointed out that convolution is not e.g. scale and rotation invariant. Thus, data augmentation methods such as transforming the data into different orientations and scales has been found practical when training CNNs [53].

2.5.4 Recurrent neural networks

A third variant of neural networks are recurrent neural networks (RNNs) [62]. What makes RNNs distinct from the aforementioned neural networks is that RNNs are not FNNs, i.e., they contain connections between neurons which form loops. In the same way that CNNs are specialized in processing grid-like data, RNNs are specialized in processing sequen-

tial data [1, 44]. RNN models are appropriate for modeling temporal structures and long-term dependencies in the data. Although CNNs can also process sequential data, e.g. by using 1-D convolutions for time series data, RNNs can process much longer sequences than FNNs. In addition, the majority of RNNs are able to process different-length inputs [44, 53].

Because the connections between RNN neurons form loops, RNNs cannot be trained in the same way as FNNs by using the backpropagation algorithm directly. For this reason, RNNs are commonly trained using a modified version of the algorithm called *backpropagation through time*, in which the network is first unfolded over time before applying the traditional backpropagation algorithm [1, 44].

RNNs contain distinctive properties compared to FNNs. A traditional RNN consists of neurons called RNN cells. In RNN cells, in addition to the traditional neuron input there is another input called the recurrent input. The same applies to the output of an RNN cell: a recurrent output is produced alongside the traditional hidden output. These recurrent inputs and outputs are not meant to be fed to other cells, but instead they are kept for the cells themselves [44, 53]. Each recurrent output of an RNN cell acts as the recurrent input of that same cell when receiving the next input sample. The basic idea behind using these recurrent inputs and outputs within each cell is that the cell retains a memory of its previous internal state. This memory is the basis of RNNs as it is the key factor in forming learned connections between input samples over time [44, 53].

Furthermore, there might be cases where it is beneficial to be able to take the future samples into account. Thus, RNNs can be extended into bidirectional RNNs which, in addition to being able to remember past events, are able to model future temporal context. These bidirectional networks, however, need to have the whole input sequence accessible before processing [1, 44].

In theory, RNNs are able to remember their whole input history. However, in practice it has been found that traditional RNNs are difficult to train to model a temporal context that is longer than a few input samples [53]. This is because either the error gradient starts to shrink (vanishing gradient) or to grow (exploding gradient) exponentially. Variants such as the long short-term memory (LSTM) [63], and later its simpler version, the gated recurrent unit (GRU) [64], have been introduced to overcome these problems. Along with their variants, these two are the most commonplace RNN architectures nowadays [44].

Both LSTMs and GRUs have been successful in various audio processing and NLP tasks, e.g., music analysis, speech modeling, and machine translation [44, 65]. When comparing the two, GRU-based models have fewer parameters and are thus computationally more efficient and might need less data to generalize. GRUs are also simpler to implement. LSTMs, on the other hand, are more complex both computationally and implementation-wise, but may have higher expression ability with more data, which may

lead to better performance. Typically, as with other machine-learning models, finding the best-performing RNN model for some specific task follows the “no free lunch" theorem, which states that one cannot deduce the best model for a task without testing the different possibilities [44, 65].

2.5.5 Deep learning and practicalities

A hot topic in today’s research, *deep learning*, is simply machine learning which utilizes deep neural networks (DNNs) [53]. A DNN, in turn, is a broad term which encompasses all neural networks with more than one hidden layer. The deeper the network, i.e. the more hidden layers there are, the more complex decision functions the networks are able to learn [44, 53]. However, deeper networks require special techniques in order to be trained properly. Probably the most prominent problem that arises when training DNNs is the vanishing gradient problem, in which the error gradient gets exponentially smaller layer by layer when using the backpropagation algorithm. This is because by the chain rule, the derivative of each layer gets multiplied by the derivatives of the previous layers. If the layer-wise derivative of even one layer is smaller than 1, the derivatives of the subsequent layers get smaller layer by layer in the backward pass [53]. The most common solution is to use an activation function in which the gradients flow well on the active paths of neurons, such as the ReLU [53, 56]. Other widely-used solutions include using *residual* or *skip connections*, which utilize shortcuts to bypass some layers, and *batch normalization*, which avoids vanishing gradients by normalizing the layer inputs so that large inputs do not reach the saturated value regions of the activation function [53].

One of the most common practical issues with neural networks is that they are prone to overfitting. This is a side effect of the large number of trainable parameters in the networks, and consequently, the strong modeling capability of the models [53]. Probably the most commonly-used technique for overfit prevention with neural networks is *early stopping*, due to its efficiency and simplicity. In early stopping, a copy of the model parameters is stored after every training epoch in which the validation accuracy was better than the best validation accuracy thus far. If the validation accuracy does not improve for a predefined number of epochs, then the training is stopped and the model with the best validation accuracy is selected [53]. Another popular overfit prevention techniques is *dropout*. In dropout, a predetermined number of randomly selected neurons in a layer are inactive in each training iteration. Since the neurons cannot fully rely on their previous-layer inputs, the network is encouraged to learn alternative activation paths for the same outputs [53].

In addition to early stopping and dropout, *regularization* in machine learning is another technique for preventing overfitting. Regularization is done by imposing the Occam’s razor principle of problem-solving on the proposed solution, which suggests that the simplest

hypothesis that fits the data is the best [8, 53]. The most common regularization technique is *weight regularization*, in which large parameter values are penalized by restricting the possible values of the weight matrix. This, in turn, leads to a more simple function to be fitted to the data [8].

Another practical issue when training neural networks is that neural networks are heavily data-driven models. The training process requires lots of data and the requirement for more data grows even more as the networks get deeper [44, 53]. Especially for smaller data sizes, but also for larger ones, it has been found beneficial to utilize pretrained DNNs when training neural network-based models. Instead of creating a network from scratch and initializing the network parameters at random, one can create ones own network by taking a whole pretrained network or some part of it, and using its parameters as the initial values for the network. Using pretrained weights has been successful in numerous machine-learning tasks, particularly when the task at hand is related to the task of the pretrained network [53].

2.5.6 Autoencoders

An autoencoder is a neural network which tries to reconstruct its input as its output, commonly used for dimensionality reduction or to remove noise [53]. Since learning a perfect reconstruction is not often very useful, autoencoders are restricted so that they are only able to approximately reconstruct their input. This forces autoencoder models to learn which properties of the input data are useful for the reconstruction and which properties are not [53].

An autoencoder typically consists of two parts: an encoder and a decoder. When training an autoencoder, the encoder attempts to produce a latent representation of the input features which has a smaller dimensionality than the input features, whereas the decoder tries to reconstruct the original input from the latent representation produced by the encoder [53]. If an autoencoder is used for dimensionality reduction, the decoder part of the network can simply be discarded after training in order to produce compressed feature representations. For noise removal, the encoder part of the network is trained so that it learns noise-free characteristics of the input data. This, in turn, leads to the decoder network being able to reconstruct noiseless versions of noisy input samples from the latent representation produced by the encoder [53].

2.5.7 Generative adversarial networks

A generative adversarial network (GAN) [66] is a framework in machine learning in which two neural networks, a *generator* and a *discriminator*, compete with each other. The generator aims to produce new data with similar characteristics as its training data, whereas

the discriminator tries to distinguish whether the samples it receives are drawn from the training data or produced by the generator. This is a so-called 'zero-sum game', in which the gain of one network is the loss of the other network [53].

The main motivation behind the GAN setting is that as the discriminator tries to learn to be better at correctly classifying whether its input samples are real or fake, the generator simultaneously tries to learn a representation which is indistinguishable from real data. After a successful GAN training, the discriminator can be discarded and the generator is able to produce realistic fake data. This setting has been successfully used in various settings, such as generating realistic fake images [53].

Training GANs is a very difficult task [53, 67]. One of the main issues when training GANs is the vanishing gradient problem. In theory, an ideal discriminator provides feedback which the generator is able to learn from in order to improve its performance. Yet, in practice, it is common that the discriminator is easier to optimize than the generator. Thus, the discriminator learns significantly faster to discriminate real and fake data than the generator is able to learn to produce realistic outputs. For this reason, the gradient of the generator starts to diminish and the generator learns hardly anything from the output of the discriminator [53, 67].

Goodfellow et al. [66] were aware of the vanishing gradient problem and proposed an alternative cost function to alleviate the problem. However, as Arjovsky and Bottou pointed out [67], this cost function makes the GAN model highly unstable. One way of improving on the stability of GANs is to use Wasserstein generative adversarial networks (WGANs), which were proposed by Arjovsky et al. [68]. In WGANs, the discrepancy measure between the real data distribution and the generated data distribution is the Wasserstein-1 distance, also known as the *earth-mover's distance*. To enforce a Lipschitz constraint, the authors proposed to clip the weights of the discriminator between -0.01 and 0.01 after each gradient update [68].

The main benefit of using WGANs over GANs is that the discriminator is not able to saturate and it converges to a linear function which provides clean gradients everywhere, thus making the generator able to learn even if it does not perform well compared to the discriminator [68]. The discriminator in a WGAN is often called a *critic* since instead of providing the probability of whether the output is real or fake, it provides a scalar score that can be interpreted as how real the output is. Another major benefit of using WGANs is that they provide a meaningful loss metric which correlates well with the quality of the generator's output [68]. In contrast, the quantity of the generator loss function in GANs correlates poorly with the generator's output, which means that practically the only way to find out how the generator is performing is to examine its output during the training process. However, it should be noted that the loss value of the generator in WGANs cannot be used to quantitatively compare different WGAN models since different critics

involve different scaling factors [68].

2.6 Active learning

Active learning (AL) is a subcategory of machine learning in which the machine-learning algorithm can query an information source to give labels for data points chosen by the algorithm [69]. This information source, also known as an *oracle* in AL literature, is usually a human annotator or a group of human annotators, but it can also be e.g. another machine-learning model or a combination of human and machine labelers [69, 70]. The key purpose of AL is to reduce the human annotation effort as much as possible, as well as to produce a well-performing machine-learning model without using more model training and more training data than is required. The need for AL in machine learning is often the case when there is an abundance of unlabeled data but labeling the whole data is too time-consuming or expensive for the given task [69, 71].

In AL, there are three common types of settings in which the AL model, commonly referred to as the learner, will query the oracle for labels [69]. The first setting is *membership query synthesis*, where the learner generates a new instance for labeling from some underlying natural distribution [72]. For example, if an NLP dataset consists of paragraphs in written text with a label assigned for each sentence, the learner could generate a new paragraph-to-be-labeled by taking a few sentences of some already existing paragraph, and then query the oracle for the label of this new paragraph. This approach has been found practical for finite problem domains and for cases when the oracle is not a human being [69, 72]. However, membership query synthesis can be problematic when using human annotators. For example, there are multiple instances of situations where the queries made by the learner are uninterpretable by human annotators [69, 70, 72].

The second setting, *stream-based selective sampling*, is based on an assumption that it is inexpensive to get unlabeled data points [69]. Here, the learner goes through each data point individually and determines which samples are queried and which are discarded. The decision to query a data point can be based on e.g. the *informativeness* of the sample. There are various informativeness measures used in AL, the most common being the *uncertainty* of a given sample (see “Uncertainty sampling" below for a clarification) [69]. The sequential manner of stream-based sampling is, among other things, useful for applications with low memory or computing resources, since data points are processed one at a time [69, 70].

The third setting is *pool-based sampling*, which is similar to stream-based sampling. The fundamental difference between the two sampling-based scenarios is that in pool-based sampling, the learner analyzes a pool of samples in one go instead of going through samples one at a time. This pool can either be a subset of the data or the whole data [70, 71]. In this approach, the basic assumption is that there is a large set of unlabeled

data and a small set of labeled data. After evaluating the informativeness of each sample in a pool, each element in the pool is ranked. Then, one or more of the most informative instances are selected for labeling [70, 71]. Out of all possible query scenarios in AL, the pool-based sampling is by far the most popular [69, 70, 71].

There are multiple strategies for how the learner chooses which data points are queried from the oracle. It should be noted that every AL method involves some kind of informativeness measure to determine these data points. Some of the frequently used query strategies include, but are not limited to [69, 70]:

- *Uncertainty sampling* or *query by uncertainty*: A classifier, typically an SVM or an HMM, is initially trained using the available labeled samples. After training the classifier, the unlabeled samples are fed to the classifier to be examined. The unlabeled instances which the classifier is most uncertain about are then given to the oracle for labeling. This method is the most widely used query strategy in AL, perhaps because it is simple to understand and it does not require much effort to implement.
- *Query by committee*: This query strategy is very similar to uncertainty sampling, with the main difference between the two being that query by committee involves multiple classifiers instead of only one. This committee of classifiers consists of distinct classifiers trained on the available labeled data. Next, these trained classifiers with competing hypotheses vote for the labels of the unlabeled samples. The instances with the largest disagreement between the classifiers are then chosen to be queried. Disagreement can be quantified using metrics such as vote entropy, Kullback-Leibler (KL) divergence, entropy-based disagreement, margin-based disagreement, or uncertainty sampling-based disagreement.
- *Expected model change*: First, an initial classification model is trained using the available labeled samples. Then, unlabeled instances are examined one by one, and the instances that would most change the current model are selected for labeling. Since the true labels for the unlabeled instances are not known, the amount of change is calculated as an expectation over all possible labels. Typically, gradient descend-based classifiers are used in this query strategy, and the amount of change for each label is the expected gradient length of updating the model. The main drawback of this query strategy is that it can be computationally demanding if the features are high-dimensional, if there is a vast number of unlabeled samples, or if there is a large set of possible labels for the data.
- *Expected error reduction*: The concept is very similar to the expected model change, and aims to give labels to the unlabeled instances that are expected to most reduce the generalization error of the classification model. This is achieved by estimating the expected future error of the current model over all possible labels using e.g. $0/1$-

loss or log-loss. This is one of the most computationally expensive query strategies, since iterating through each unlabeled data point requires both retraining the model and estimating the expected future error for all possible labels.

- *Variance reduction*: Rather than trying to minimize the computationally-heavy expected error directly, the generalization error can be indirectly reduced by minimizing the output variance of the classification model. Commonly, this has been found more practical than expected error reduction in situations where there is an abundance of unlabeled data. Even so, computational complexity can be an issue with a large set of unlabeled data and with a classifier containing a substantial number of parameters. Variance reduction has been found problematic in fields such as NLP.
- *Density-weighted methods*: Instead of treating all unlabeled samples equally, the density-weighted methods take the underlying data distribution into account when querying for new labels. This tackles one of the problems encountered in e.g. expected error reduction and variance reduction, which are both prone to querying outliers in the data. One possible solution is to weight the informativeness of each unlabeled instance by its average similarity to other data points in the distribution. Density-weighting can be combined with practically any informativeness measure, and in many tasks density-weighted query strategies have been found to be more successful than their non-weighted counterparts. In addition, computational efficiency can be increased if the densities or the similarities between data points are precomputed and stored into memory before the AL process.

Most AL methods commonly assume that the labels given by the oracle are correct [73]. Nevertheless, even domain experts are prone to make mistakes during the labeling process. There are a number of reasons why labeling errors, also called labeling noise, might occur. The sources of labeling errors include the level of domain expertise of the oracle, the difficulty of the labeling task, and the quality of the sample to be labeled [69, 73]. Furthermore, the quality of the labels are prone to change during the labeling process. This can be the case if the labeling process is time-consuming and tedious, or if the annotator becomes more familiar with the given task over time, for example. It is also possible that the labeling process is arranged in such a way that the labels are slightly biased toward some label, which can lead to accumulated labeling errors over time. Noise in labels not only makes the learner less accurate, but it also makes the query instances less informative [69, 73].

The most popular approach to tackle the problem of noisy labels has been to use Internet-based crowdsourcing techniques, such as Amazon Mechanical Turk[3] or other online annotation platforms [69, 71]. The basic idea behind crowdsourcing techniques is to value quantity over quality by inexpensively acquiring multiple labels from non-experts, and as-

[3] `https://www.mturk.com/`

suming that the correct label is the label that the majority voted for. The main problem with crowdsourcing is that a significant number of annotations is required for each instance to ensure that the given label is correct [69, 73]. Other approaches to handle noisy labels without crowdsourcing techniques include, e.g. [73], where Bouguelia et al. propose a two-stage AL method that handles noisy labels without the need for multiple annotators. Their method significantly improved classification results compared to several other baseline methods by first labeling instances which highly influence the learner, and then eliminating labels which are noisy based on how much they are influenced by the changes in the model. Other practical considerations and active research questions in AL include taking the varying labeling costs between instances into consideration, dealing with the changes of the quality of annotations over time, querying using multiple oracles, labeling instances for multiple tasks simultaneously, reusing the data, and querying for repeated labels from the same annotator [69, 70].

AL has also been used in SER as a solution to reduce annotation effort and to utilize already existing annotations efficiently. Zhang and Schuller [74] propose two iterative AL methods to reduce annotation effort. The first method is based on imbalanced emotional classes in the sense that it selects for labeling instances which it predicts as a sparse class in each iteration. The second method iteratively chooses the instances for which it predicts a medium confidence score to be labeled. The paper demonstrated that both methods efficiently reduced the required number of annotations. Zhao and Ma [75] presented an iterative AL algorithm which utilized conditional random fields to determine the level of uncertainty for each unlabeled sample. The most uncertain samples were then selected for annotation. In most cases, their method performed better than random sampling for data selection. Abdelwahab and Busso [27] examine different AL methods that are based on uncertainty and maximizing the diversity in the training set to simulate limited annotated data in DNN-based classifiers. Their study shows that the tested AL methods outperform random sampling-based methods when selecting samples for labeling.

As already mentioned, uncertainty sampling-based methods are the most common AL methods due to their simplicity [69, 70]. These methods have been proven to efficiently reduce the number of required manually annotated samples in various machine-learning tasks, such as NLP [76] and automatic speech recognition [77]. However, as shown in e.g., [77], a significant number of initial labels are required for the training of the base classifier in order for it to produce reasonable outputs. Simply put, a classifier cannot determine the uncertainty of a sample reliably if it has not been taught with a sufficient amount of labeled data. Furthermore, the more complex a given task is, the more labeled data are required for the initial base classifier to be trained properly [70]. Hence, an alternative AL approach needs to be taken in order to tackle cases with a scarce labeling budget compared to the overall size of the dataset. To this end, Zhao et al. [78] have

proposed an AL method called medoid-based active learning (MAL), for sound event classification. Their method was designed for cases with a limited labeling budget and when the annotations add up to only a small portion of data. Since this is the premise of the annotation process of the present study, MAL serves as the foundation of the AL method used in the present experiments. This modified version of MAL is described in further detail in Section 3.1.

2.7 Domain adaptation

Often the basic starting point for the theoretical and empirical models in machine learning is that the training and testing data are drawn from the same data distribution [79]. Although this might be the case in theory, in practice this assumption does not always hold true. For example, the distribution of the data might change over time, or we might want to deploy machine-learning models trained on one type of data for another type of data. In addition, it is common that the collected data is not a fully unbiased representation of the real-life data [79, 80]. In domain adaptation (DA), it is not assumed that the training and testing data are drawn from the same distribution. Instead, the machine-learning model is trained using data from one or more *source domains* (i.e. source data distributions) and is then applied to a different *target domain* (i.e. target data distribution) [79].

The most common DA methods can be categorized into three main types, the first of which is *instance-based* DA [80, 81]. Here the aim is to minimize the target domain error by using the source domain data while correcting for sample selection bias instead of using all the data as it is. This can be done by performing *importance-weighting*, in which the aim is to weight the cost of errors for some training samples in order to make the errors on these samples more significant than the errors on some other samples [82]. Importance-weighting can be performed either for the data by assuming *covariate shift*, or for the classes by assuming *prior probability shift*. In the well-studied covariate shift setting for DA, the source and target domain data distributions differ but the conditional distributions of the model outputs are the same, i.e. the concept being learned remains stationary [83]. In contrast, the prior probability shift setting in DA considers that the conditional distributions of the model outputs are different [83]. One approach for the covariate shift is to individually reweight the samples in the source domain in order to make them as alike as possible to the samples drawn from the target domain distribution. This is achieved by giving more weight to the source domain instances that are similar to the target domain instances [83]. Frequently occurring alternative names for instance-based methods in DA literature are *reweighting* and *sample-based* methods [80, 81, 83], which both describe the core idea of these methods well.

The second type of DA methods are the *feature-based* methods. These methods try to modify the feature space representations of the data so that both the source and target

domain data represent the same distribution of features and labels. This modification to construct a common feature space can be done either to both the source and target domain data (e.g. [84]) or only to the source domain data (e.g. [85]). Feature-based DA methods try to utilize the source-domain labels in the adaptation process in order to create a classifier that performs well on the target domain data [86]. A common approach is to simultaneously transform the source and target domain feature spaces to be as alike as possible and to train a well-performing classifier for the labeled source-domain data that has been transformed. In the recent years, one of the most popular ways of doing this has been to use GANs or WGANs [53, 68]. The basic framework of GANs and WGANs (presented in Section 2.5.7) can be adopted into DA by slightly altering the mindset of the GANs. In GAN-based DA, the discriminator tries to distinguish whether the samples are drawn from the source or the target distribution, whereas the generator tries to fool the discriminator by modifying the feature representations of the source and target domain data so that they are indistinguishable [53]. In other words, the generator tries to find a common feature representation for the source and target domain data so that their distributions would be similar. This generator is often the feature extractor in GAN-based DA [53].

The third type of common DA methods are what we refer to in this text as *parameter estimation-based* methods. This is probably the most diverse of the three DA method categories, since it involves all DA methods which somehow incorporate the adaptation process into the procedure of estimating classifier model parameters [87]. One of the most widely used parameter estimation-based DA methods are iterative methods which first train a model using labeled source-domain samples, and then give pseudo-labels for some of the unlabeled target-domain data [87]. Then, a new model is iteratively trained by adding the pseudo-labeled samples to the training set and retraining the model in each iteration. Other methods in this category include e.g. Bayesian models where the prior distribution is fit to the source-domain data [88].

One of the most prominent applications of DA is to utilize the adaptation process for situations where labeled target domain data are either scarce (semi-supervised DA) or unavailable (unsupervised DA). For instance, many of the PSP tasks discussed in Section 2.2.2 fall within the scope of settings which involve a scarcity of labeled samples. DA has been successfully used in a plethora of machine-learning application areas to account for missing or scarce labeled target domain data, including e.g. speech processing [84], computer vision [85], NLP [86], and disease diagnosis [89]. Drossos et al. [90] presented an unsupervised DA method for acoustic scene classification (ASC). Their method, Wasserstein distance-based domain adaptation (WDA), improved the previous state-of-the-art DA method for ASC by changing the GAN-based adversarial setting into a WGAN-based setting in the adaptation process. The theoretical foundation of WDA and its training algorithm serve as the basis of the DA method of this thesis, which is a slightly

modified version of WDA. This modified version is described in Section 3.2.

DA can also be utilized in SER as a solution to reduce the need for annotated data. Additionally, DA can be used to address differences between training and testing corpora in SER, including e.g. different languages, speakers, recording conditions, linguistic content, class distributions, and sizes of corpora. Sagha et al. [23] studied a cross-lingual SER setting using four corpora of different languages. They proposed a DA method which attempts to find a common representation space for the source and target languages using principal component analysis (PCA) and kernel canonical correlation analysis (KCCA). First, PCA is used to map the source and target data into their principal components, both with respect to themselves and with respect to each other. Then, KCCA is used to select the top N dimensions that maximized the correlation between the mapped data. Their method provided an improvement in average classification performance compared to the state-of-the-art DA method for SER at the time. Deng et al. [31] conducted a cross-corpus SER study using unsupervised DA. They proposed adding a Universum loss to the reconstruction loss of an autoencoder-based classifier to ensure low reconstruction and classification errors in both domains. Deng et al. [30] extended a popular unsupervised deep denoising autoencoder by combining a supervised learning objective to create a semi-supervised DA method for SER. They presented two variants, with and without skip connections, of which, on average, the former outperformed the latter. Abdelwahab and Busso [29] use an unsupervised neural network-based adversarial DA approach for SER. Their method aims to learn a domain-invariant feature representation between labeled source data and unlabeled target-domain data while maintaining a good performance on the primary SER task.

3. METHODS

The main research goal of the present study is to create well-performing SER model for real-life child-centered audio recordings from a NICU. Since the premise of the present experiments is that there is an absence of labeled data in the target corpus, traditional machine-learning methods utilizing supervised learning cannot be used. Instead, alternative machine learning-based techniques, namely cross-corpus generalization, AL, and DA, are compared in the present study. An overview of the SER system of the present experiments is given in Figure 3.1. Note that in the figure, the source corpus and source labels for AL are a labeled subset of the target corpus.

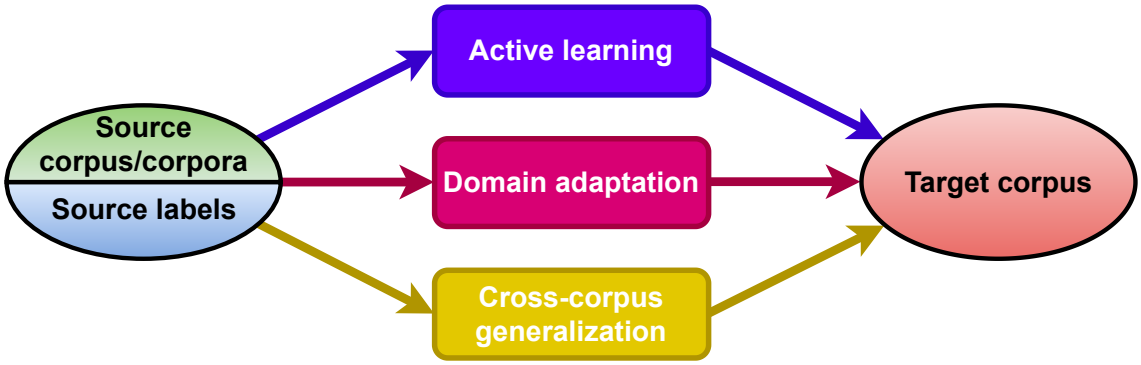


Figure 3.1. *An overview of the SER system of the present experiments. For one or multiple source corpora with labels available, a SER classifier is either trained using AL, DA, or cross-corpus generalization. Then, the trained classifier is tested on a separate target corpus, which has little to no labeled data available.*

In this chapter, the core methods of the present study are described. Section 3.1 depicts the AL method used in this study, followed by a description of the DA method of the present experiments in Section 3.2. Finally, the basic idea of cross-corpus generalization is introduced in Section 3.3.

3.1 Medoid-based active learning

The medoid-based active learning (MAL) algorithm can be divided into three subsequent parts:

1. Obtain a distance matrix that contains the distances between each sample in a dataset.

2. Perform k-medoids clustering using this distance matrix.
3. Starting from the largest cluster, annotate the medoids in a descending size order.

The rest of this section describes these parts in further detail.

3.1.1 Distance matrix in MAL

For MAL, a distance matrix that contains the distances between each pair of samples in the dataset is required [78]. The distance metric should be chosen according to the task MAL is intended to be used for. For example in the context of SER, the purpose of this matrix is to store the pairwise distances between all utterances in a dataset in order to discriminate utterances in an emotion-based feature space as well as possible. In other words, a suitable distance metric for clustering in SER is a metric in which utterances with similar emotional content are close to each other, whereas utterances with a clearly distinct emotional content have a large distance between each other.

The metric used in the present experiments was selected based on pilot experiments with MAL using existing SER datasets. To obtain the distance matrix, a 600-dimensional utterance-level log-mel feature representation (see Section 4.2 for a detailed description) is first used as the initial feature representation of each sample in a dataset . Then, the 600-dimensional log-mel features are compressed into a 32-dimensional latent representation using an DNN-based autoencoder with six layers. After training the autoencoder and compressing the log-mel features using the trained autoencoder, the pairwise dissimilarities between each sample in the dataset are computed from the bottleneck features. The dissimilarity measure used in the present study is the Pearson distance, also known as the Pearson correlation distance or simply the correlation distance. The Pearson distance, d_P, between vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ is defined as

$$d_P(\boldsymbol{a}, \boldsymbol{b}) = \frac{\langle \boldsymbol{a} - \mu_{\boldsymbol{a}}, \boldsymbol{b} - \mu_{\boldsymbol{b}} \rangle}{\|\boldsymbol{a} - \mu_{\boldsymbol{a}}\| \|\boldsymbol{b} - \mu_{\boldsymbol{b}}\|} , \tag{3.1}$$

where $\mu_{\boldsymbol{a}}$ and $\mu_{\boldsymbol{b}}$ are the mean values of the elements of $\boldsymbol{a}$ and $\boldsymbol{b}$, respectively [91]. Pearson distance is a commutative operation (i.e. $d_P(\boldsymbol{a}, \boldsymbol{b}) = d_P(\boldsymbol{b}, \boldsymbol{a})$) and it produces a distance of zero only when $\boldsymbol{a}$ and $\boldsymbol{b}$ are the same. However, it does not fulfill the third condition of a metric, the triangle inequality, and is thus not regarded as a proper distance function. Consequently, although the term ‘distance matrix’ is used in this thesis, the distances described by the matrix are not mathematically exact distances.

3.1.2 k-medoids clustering in MAL

After obtaining the distance matrix that describes the dissimilarities between each sample in the dataset, a single sample is randomly selected as an initial starting point. This sample is also added to a set of selected points, S. Then, data points are added one by one to S until the set reaches a size of k, which is a user-defined positive integer. After the initial starting point, the following data points are chosen based on *farthest-first traversal*. In farthest-first traversal, the farthest sample to the current set S is chosen as the next point to be added to S. Here, the distance from a sample, $\boldsymbol{a}$, to the set S is defined as

$$d_P(\boldsymbol{a}, S) = \min_{\boldsymbol{b} \in S} d_P(\boldsymbol{a}, \boldsymbol{b}) \ , \tag{3.2}$$

where d_P is the Pearson distance from Equation 3.1 [78]. By using farthest-first traversal, the set S can be thought of as being a set of k data points that lie in the distance space D as diversely as possible, where $d_P(\boldsymbol{a}, \boldsymbol{b}) \in D \quad \forall \boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n, \ n \in \mathbb{Z}^+$.

The data points in S are then used as the initial medoids for a k-medoids clustering algorithm (see e.g. [92] for a detailed description), which is similar to the popular k-means clustering algorithm. The main difference between the two algorithms is that the centroids of the clusters are real samples in k-medoids, whereas they are arbitrary points in k-means. Comparing the two, the k-medoids algorithm has proved to be better than k-means in terms of accuracy, especially with larger datasets. One of the main disadvantages of k-means is that it is sensitive to outliers in the data [78, 92]. An example of the two algorithms for randomly generated 2-D data points is depicted in Figure 3.2. In the example, the value for k was set to 2 and the dissimilarity measure between the data points was the squared Euclidean distance.

In the k-medoids clustering algorithm, each data point is linked to its nearest initial medoid. Then, the medoids are iteratively updated in order to minimize the sum of the distances of all data points to their nearest medoids. This process continues until the sum of the distances cannot be reduced by updating the current medoids in each cluster. The medoids can be considered as the best representatives of all elements in each cluster.

3.1.3 Annotation process in MAL

After performing the k-medoids clustering algorithm, the clusters are sorted in a descending order based on the number of elements in each cluster. Then, the cluster medoids are presented to the annotator for labeling. The medoid labels are also assigned as the predicted labels for the rest of the cluster elements in order to artificially increase the number of labels. These predicted labels are referred to as the *cluster labels*. The experiments of

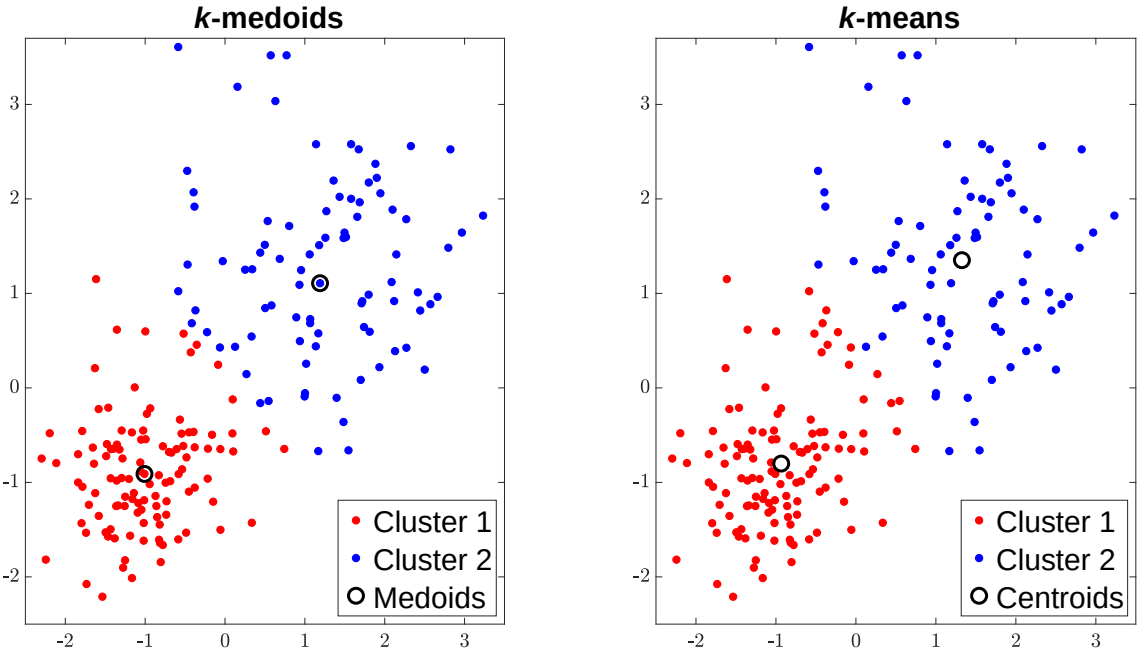


Figure 3.2. *The clusters and the cluster centroids for the k-medoids algorithm (left image) and the k-means algorithm (right image) for randomly generated data.*

the present study go beyond the original MAL paper [78] by including the case where the cluster labels are not utilized.

The selection of k is a trade-off between the average cluster size and the accuracy of the cluster labels. With a large k, the average size of the clusters is smaller and thus the cluster labels are more accurate, but on the other hand, a large k results in less cluster labels. Furthermore, since it is assumed that the annotations add up to only a small portion of the data, a scenario in which all medoids are labeled and a new iteration of the clustering algorithm is possible is not considered.

3.2 Wasserstein distance-based domain adaptation

In Wasserstein distance-based domain adaptation (WDA), a neural network-based classifier is adapted to a target domain by using labeled data from a source domain. This neural network-based classifier consists of two parts, a feature extractor, F, and a label classifier, C_L. The adaptation process of WDA involves two steps, which are demonstrated in Figure 3.3. The first step is training the neural network using the source domain data, D_S, to obtain the source domain feature extractor, F_S. The combination of the trained neural networks F_S and C_L is also known as the *source model*. The source domain data, D_S, includes both the source domain observations, X_S, and their respective labels, Y_S. The second step is adapting F_S for the target domain data, D_T. This adapted feature extractor is denoted as F_T.

In the first step (upper image in Figure 3.3), a neural network consisting of two parts, F

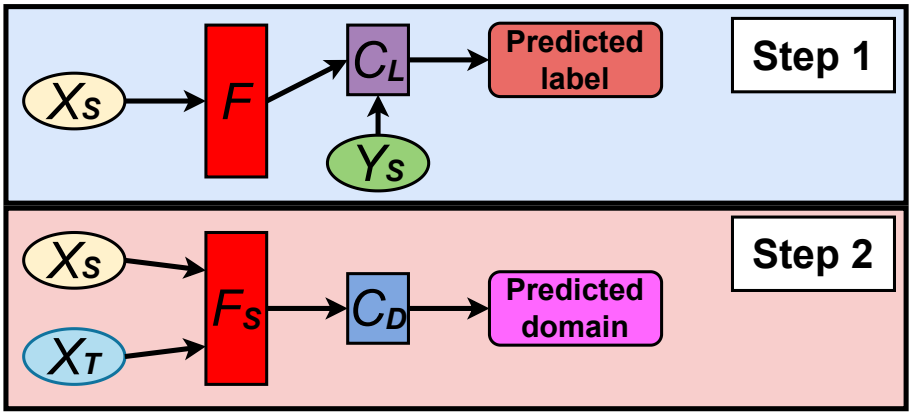


Figure 3.3. *The two steps of the adaptation process of WDA. The first step (upper image) involves training a neural network consisting of a feature extractor (F) and a label classifier (C_L) using the source data to predict the label of the input observation. In the second step (lower image), the network is adapted to the target data using an adversarial training formulation.*

followed by C_L, is trained using D_S to obtain F_S using binary cross-entropy [90]

$$L_L(\boldsymbol{x}, \boldsymbol{y}) = - \sum_{(\boldsymbol{x},\boldsymbol{y}) \in (X_S,Y_S)} \boldsymbol{y}^T log_{10}(C_L(F(\boldsymbol{x}))) \tag{3.3}$$

as the loss function. This is a crucial step in the adaptation process, since it is important to obtain a model which performs well in the source domain. Following the underlying theory of DA in [90], the upper bound of the error in the target domain is affected by three factors. The first factor is the combined error of the ideal joint predictor, i.e., the classifier for both the source and target domain data. This factor is often neglected in DA since it is assumed that it is possible to obtain a model which has a small error in both domains [79]. The second factor is the discrepancy between the distributions of D_S and D_T, which the second step of the adaptation aims to minimize. The third factor is the error of the model in the source domain. The second factor cannot be affected during the first step of the adaptation, and the third factor cannot be affected during the second step of the adaptation. Therefore, it is important to obtain a well-performing source model in the first step of the adaptation process in order for the second step to be able to succeed.

In the second step (lower image in Figure 3.3), F_S is adapted to D_T to obtain F_T by minimizing the Wasserstein-1 distance between the distributions of D_S and D_T using an adversarial training process. Following the WGAN framework (introduced in Section 2.5.7 and specified for DA in Section 2.7), F_S is adapted into F_T by finding a common feature representation for D_S and D_T by iteratively minimizing the losses

$$L_{C_D}(\boldsymbol{x}, \boldsymbol{z}) = \sum_{\boldsymbol{x} \in X_S} C_D(F_S(\boldsymbol{x})) \; - \sum_{\boldsymbol{z} \in X_T} C_D(F_T(\boldsymbol{z})) \tag{3.4}$$

and

$$L_{F_T}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = \sum_{\boldsymbol{z} \in X_T} C_D(F_T(\boldsymbol{z})) + L_L(\boldsymbol{x}, \boldsymbol{y}) \,, \tag{3.5}$$

where C_D is the domain discriminator, X_T are the target domain observations, and L_L is the label loss term from Equation 3.3 [90]. The parameters for C_D and F_T are updated in turns, where Equations 3.4 and 3.5 are the loss functions for updating the parameters of C_D and F_T, respectively. The output features of F_T are the input features for C_D. Additionally, the parameters of F_S serve as the initial parameters of F_T. It should be noted that F_S and C_L are not updated during the second step of the adaptation.

As pointed out in [90], the minimization of L_{C_D} and L_{F_T} (Equations 3.4 and 3.5, respectively) is shown to minimize the Wasserstein-1 distance between the distributions of D_S and D_T. However, since the common feature representation for D_S and D_T is not guaranteed to perform well on C_L, the authors proposed to add an additional loss term to account for the degradation of performance on C_L when adapting the parameters of F_T. This additional term is the second term, L_L, in Equation 3.5, and was not included in the original WGAN paper [68]. For a detailed formulation of the WGAN algorithm in WDA, see Algorithm 1 of [90].

As can be observed from Equations 3.3, 3.4, and 3.5, the adaptation process is unsupervised since the target domain labels, Y_T, are not required. In addition to the unsupervised version of WDA, a semi-supervised variant is also explored. This version utilizes a small subset of Y_T during the adaptation process so that the label classification accuracy of C_L on the subset of Y_T is computed after each training iteration. This information is used as the optimal model selection criterion of the adaptation process of F_T so that the model with the highest accuracy on the subset of Y_T is selected as the final adapted model. In other words, the subset of Y_T is only used for model selection purposes and not in the adaptation loss functions (Equations 3.4 and 3.5).

3.3 Cross-corpus generalization

In cross-corpus generalization, $N \in \mathbb{Z}^+$ labeled source datasets s_i are merged into one training set, S, where $\forall s_i \in S$. Next, this training set is used to train a classifier using supervised learning. Finally, the trained classifier is tested on an unlabeled target dataset T, where $T \notin S$.

4. EXPERIMENTAL SETUP

This chapter describes the experiments conducted in the present study. First, the datasets used in this thesis are depicted in Section 4.1. Then, the experiments conducted for four simulation corpora and the primary audio material of the present study are described in Sections 4.2 and 4.3, respectively. The SVM models were implemented with the Scikit-learn framework [93] and the neural network-based models were implemented with the PyTorch deep learning framework [94], both using the Python programming language.

4.1 Datasets

This section provides an overview of the datasets used in the present experiments. All of the corpora except NICU-A are from now on referred to in this text as *simulation corpora* in reference to them belonging to the simulation setup described in Section 4.2. The corpora described in Sections 4.1.1, 4.1.2 and 4.1.4 are freely available and have been widely used in the SER research community.

4.1.1 EMO-DB

The Berlin Emotional Speech Database[1] (EMO-DB) [42] is well-known and is perhaps the most widely used corpus in SER. EMO-DB consists of 535 spoken utterances with seven emotional labels: anger, boredom, disgust, fear, joy, neutral, and sadness. The data collection consisted of 10 professional actors (five male and five female) speaking 10 predefined, short sentences in German. The recordings were taken in an anechoic chamber with a 48-kHz sampling frequency and they were subsequently downsampled to a 16-kHz sampling frequency. The actors read sentences with predefined emotions, which were also the emotional labels of the sentences. Each utterance in EMO-DB has only one label.

After obtaining the recordings, a data cleaning process was conducted using 20 test subjects in a listening experiment. The test subjects were only allowed to listen to each utterance once before deciding on the emotional state and the naturalness of the utterance. Utterances with an emotion recognition rate of less than 80% and overall naturalness of

[1] Available at: `http://emodb.bilderbar.info/index-1024.html`

less than 60% were left out of the corpus.

4.1.2 eNTERFACE

The eNTERFACE[2] corpus [95] is an audiovisual emotion database. eNTERFACE consists of 1287 video samples with six emotion categories: anger, disgust, fear, joy, sadness, and surprise. The video samples of the final version of the corpus incorporate recordings from 42 test subjects (eight female) from 14 different nationalities. The recordings were conducted in an office environment with predefined spoken content in English, with two human experts judging the quality of the spoken utterance. Only the audio tracks, which were recorded with a 48-kHz sampling frequency, were used in this study.

Each test subject listened to six successive short stories, each of them evoking a particular emotion. There were five different utterances for each short story as reactions to the given situation. After every short story, the subject was able to read, memorize and pronounce one of the proposed utterances, one at a time. If the expressed emotion was agreed on by both the human experts, the utterance was added to the dataset. Otherwise, the subject was asked to try again. If the test subject was not able to perform the correct emotion after a few attempts even with the help of the human experts, the utterance was left out of the dataset and the experiments continued with the next utterance.

4.1.3 FESC

The Finnish Emotional Speech Corpus (FESC) used in this study was originally created as the emotion study material for a study by Airas and Alku [43]. Nine native-speaking Finnish professional actors (five male and four female) portrayed emotions of five different categories: neutral, sadness, joy, anger, and tenderness. The emotions were chosen so that they could be clearly separated in the valence-arousal space. The speech material was a text passage of 83 words of Finnish prose, in which different emotions could be easily expressed. The actors were asked to pronounce the text passage ten times with a given emotion, leading to a total of 50 recitations for each actor and a total of 450 spoken passages altogether. The text passages in a given emotion were in a semi-random order, wherein the same emotion was never repeated twice in a row. The recordings were conducted in an anechoic chamber with a sampling rate of 48 kHz.

Since the audio files of FESC were considerably longer than the audio files of the other corpora used in this study, the spoken passages were split into utterances according to the procedure described in Appendix A. After splitting the audio data of the corpus into separate utterances using this procedure, there were a total of 4254 utterances with an average length of 4.41 seconds. The emotional labels of the utterances were the same

[2] Available at: `http://www.enterface.net/results/`

as the labels of the spoken passages the utterances were taken from.

4.1.4 RAVDESS

The Ryerson Audio-Visual Database of Emotional Speech and Song[3] (RAVDESS) [96] is a multimodal database of emotional speech and song, including a total of 7356 recordings. Only 1440 of these recordings were used in the present study, since the recordings containing anything other than speech were discarded. Altogether 24 professional actors (12 male and 12 female) spoke in a North American accent covering eight different emotional labels: neutral, calm, happy, sad, angry, fearful, surprise, and disgust. All of the actors were native English speakers. The recordings took place in a professional recording studio with a sampling rate of 48 kHz.

All emotional conditions except neutral were vocalized at two levels of emotional intensity, normal and strong. The actors expressed two distinctive neutral statements in a given emotion multiple times, all of which were later reviewed by three investigators. Any clips containing lexical errors or irrelevant gestures in the video recordings were removed. Out of the remaining clips, the best two takes as agreed through the consensus of the investigators were selected for the corpus. After post-processing the selected clips, the corpus was extensively validated for reliability.

4.1.5 NICU-A

The FinEst NICU Audioset (NICU-A) was collected as a part of the APPLE study, and is the primary audio material for which our SER system was aimed to be deployed on. The overall goal of the study is to examine the effect of parental proximity and communication on a child's development for prematurely born children. Only the recordings from Turku University Hospital and only those in which both parents had Finnish as their mother tongue were included the present study. For each family that took part in the research project, the recording process of the preterm babies' sound environment was conducted at a NICU using the LENA[4] recorder. LENA consists of both software and a recording device, and is considered as the standard for measuring vocal interactions with children up to three years in age. In the recording process, the recorder was set next to the child, either in the vest pocket that comes together with the device or without the vest. For the recordings, the single family rooms of the NICU served as the recording environment. In these rooms, the sound environment was moderately quiet since, apart from family members, there were only nurses and doctors occasionally visiting the rooms while carrying out healthcare routines. Parents and nurses were instructed to keep the recorder near the baby in all situations. Data from each child consists of a continuous 16-hour recording.

[3] Available at: `https://zenodo.org/record/1188976`

[4] For further details, see: `https://www.lena.org/`

By utilizing the broad-class speaker diarization output produced by the LENA software [97], the 16-hour recordings were split into utterances. An important sidenote is that the audio segments with tags Male Adult Far (MAF) and Female Adult Far (FAF) were included in the present study, although the majority of studies that involve the use of LENA-produced timestamps discard the audio segments which are tagged as 'far'. Furthermore, Cristia et al. [98] even suggest to exclude the 'far' categories from consideration for most purposes due to their inferior quality compared to the segments which are tagged as 'near'. However, the audio segments with tags MAF and FAF were included in the present experiments based on a validity study for the same data, which verified that the recordings with LENA-produced tags MAF and FAF had a valid quality compared to the recordings with tags Male Adult Near (MAN) and Female Adult Near (FAN) [99]. This is probably due to the highly-controlled recording conditions (one medium-sized room) with limited environmental noise.

The data consisted of 43 families with a total of 688 hours of audio. Following the split of the 16-hour recordings into utterances based on the speaker diarization output produced by the LENA software, audio files shorter than 0.6 s were discarded. After removing the shortest utterances, the total number of utterances was 129,007 with an average length of 1.57 seconds (approximately 56 hours of speech). The recordings had a 16-kHz sampling frequency.

Two professionals familiar with the research project carefully selected 35 families for the training data (average age of approximately 33.3 gestational weeks (GW) with a standard deviation of 0.6) and eight families for the test data (average age of approximately 33.4 GW with a standard deviation of 0.3). The criterion for the selection was to maximize the representativeness of both data sets in terms of covariates such as child health, parental presence etc. After pre-processing the data of NICU-A, both the training set and test set were partially annotated using an annotation platform that the present author specifically developed for this purpose.

4.1.6 NICU-A annotation procedure

For the training data, samples were selected for annotation using MAL. Similar to the AL-based experiments in Section 4.2.3, the log-mel features of the full unlabeled training set (101,813 samples) were compressed into a 32-dimensional feature representation using an autoencoder. The training and validation data for the autoencoder were based on a random split of the unlabeled training set into two sets with a ratio of 80:20 utterances. After compressing the features, the MAL algorithm was performed for each of the 35 training set families separately. After the clustering algorithm, the medoid samples were annotated in a descending order based on the cluster size, independent of the family. Two annotators, a professional familiar with the research project and the author of this

thesis, performed labeling for distinct subsets of the training data, except for the first 200 samples that were annotated by both in order to measure the inter-rater reliability of the annotations.

For the test data, gold standard annotations were derived by having three speech or clinical experts familiar with the research project to perform annotations for a randomly selected subset of the test data (27,194 samples). These samples were the same for each annotator. The final labels were determined by performing majority voting for the three labels from each annotator. The samples for which a majority agreement could not be determined were removed from the test set. The labeled data of the test set will be referred to in this text as the gold standard data.

In the annotation process, the annotators were given two distinct tasks for each utterance. The tasks were to select between two classes for arousal (high/low) and to select between three classes for valence (positive/neutral/negative). These tasks were presented in a random order for each utterance. It was also possible to label the utterance as erroneous if the emotional content of speech could not be inferred by the annotator, such as audio samples corrupted by noise, overlapping speakers, very short utterances, and samples without any speech at all. The utterance was played before each separate task and the user was able to replay utterances without any restrictions. The annotation could be stopped and resumed at any time. Also, the annotator was able to go to the previous utterance at any given time.

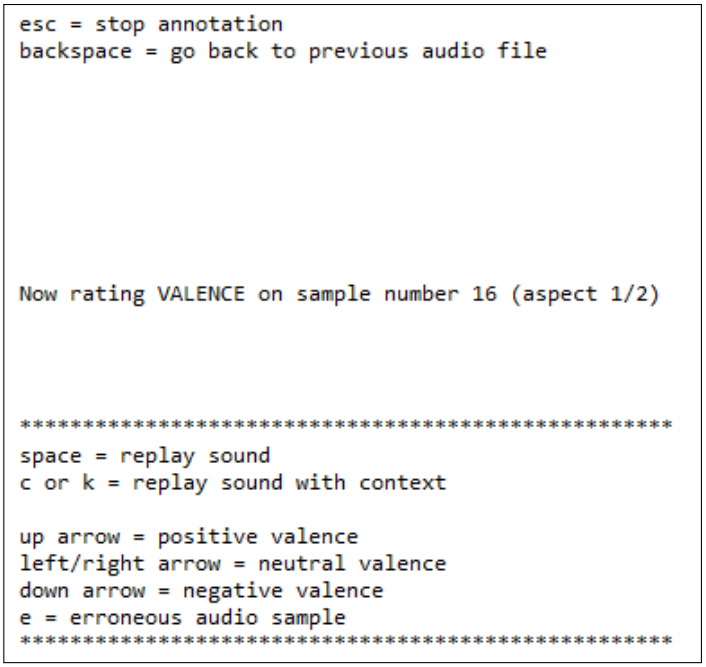


Figure 4.1. *A screenshot of the annotation platform for the test data.*

As an addition for the test data, a 10-second segment of the preceding audio context was played together with each utterance. This context was meant to assist in the annota-

tion by helping the annotators to obtain a contextual understanding of the communicative situation. However, the annotators were instructed to assign labels based on only the utterance following the context. It was possible to replay the utterance with and without the context with no restrictions. Figure 4.1 shows a screenshot of the text-based user interface of the annotation platform for the test data. The annotation process took approximately six working days for the training data and three working days for the test data altogether when the working time of all annotators was combined. For the training data, the annotation process took approximately 19 seconds for each audio sample. For the test data, the annotation lasted for approximately 40 seconds for each audio sample.

After the annotation processes for the training and test data, and after performing majority voting for the test data, all samples which were labeled as erroneous were removed from the data. After removing the erroneous files, the sizes of the labeled training set and the gold standard set were 5198 samples and 345 samples, respectively. When the sizes of these sets are compared to the total number of samples, the labeled training set corresponds to approximately 4% and the gold standard set to approximately 0.3% of all samples in NICU-A. Table 4.1 shows the class distributions of the training set and the gold standard set. By labeling all samples belonging to a cluster based on the cluster's medoid label, the size of the labeled training set was increased to 33,979 samples for the AL-based experiments involving cluster labels.

Table 4.1. *The class distributions of the annotated training set and the gold standard set of NICU-A.*

	Valence			**Arousal**	
	positive	***neutral***	***negative***	***high***	***low***
Training set	1509	3391	298	3165	2033
Gold standard set	120	214	11	89	256

To determine an estimate of the inter-rater reliability, Cohen's kappa was used for the training data (two annotators) and Fleiss' kappa was used for the gold standard data (three annotators). For further details about both kappa scores, see e.g. [100]. For the training data, the first 200 samples were the same for both annotators to determine a kappa score. After removing the files which were labeled as erroneous, the kappa scores for valence and arousal were 0.78 and 0.64, respectively. The kappa score was 0.77 for a binary decision whether a sample was erroneous or not. For the gold standard data, the kappa score was 0.48 for valence and 0.28 for arousal after removing erroneous files. The kappa score was 0.51 for the binary decision for the erroneous files. The kappa scores for the labels of the gold standard data indicate that annotating the given type of real-world audio data even for binary or ternary emotion categories is not an easy task. Based on the inter-rater agreement rates, the annotators agreed more on the valence ratings of the test samples than they did for arousal. Thus, verbal expression of valence can be regarded as more transparent in NICU-A than that of arousal.

The notable difference between the kappa scores of the training data and the gold standard data may be explained by the nature of the MAL algorithm, which tries to group data points together which are similar in the valence-arousal space. Since the medoid samples were annotated in a descending order based on the cluster size, the first 200 samples for which the kappa score of the training data was computed were the medoid samples of the largest clusters. Based on the MAL algorithm, these 200 samples were the most acoustically distinct samples in the training data. Thus, assigning a label for the medoids of these clusters is, in theory, easier than assigning a label for medoids of smaller clusters. This can also be observed by comparing the kappa score of the first 40 samples and the last 40 samples of the 200 mutual samples for arousal after removing the erroneous files. The kappa score for arousal was 0.95 for the first 40 samples and 0.59 for the last 40 samples. By having a larger number of mutual annotated samples for the two annotators of the training data, the kappa scores of the training samples would most likely be closer to the corresponding scores of the gold standard data. The finding also demonstrates the inherent difficulty in annotating a random sample of real-world speech for emotional content.

4.2 Simulation setup

The labels for NICU-A were not available at the beginning of the study. Thus, a simulation setup for experimenting with the four already available SER corpora (EMO-DB, eNTERFACE, FESC, and RAVDESS) was built to simulate different strategies for deploying a SER system on a new unannotated corpus so that, once acquired, the most promising method candidates could be applied to labeled subset of NICU-A. Figure 4.2 depicts a block diagram of the six different simulation setup scenarios in which EMO-DB is used as an example of a simulated test corpus for which prior labels would not be available. These same tests were also conducted for the three other simulation corpora in a similar manner.

Log-mel, GeMAPS, and eGeMAPS features (described in Section 2.3) were used in all experiments except DA-based experiments where only log-mel features were used based on their superior performance in pilot experiments. For the log-mel features, 40 mel filters were used with a Hann window using a 30-ms window size and 10-ms shifts. To get constant-dimensional utterance feature representations, the functionals consisting of the mean, variance, skewness, kurtosis, min, max, and range were taken from the time series of the log-mel features. Additionally, first and second order delta features for the log-mel features were extracted, and the functionals of mean, variance, skewness, and kurtosis were applied to both delta features. This resulted in a 600-dimensional feature vector for the log-mel features. The 62- and 88-dimensional GeMAPS and eGeMAPS features were extracted using the openSMILE toolkit [14].

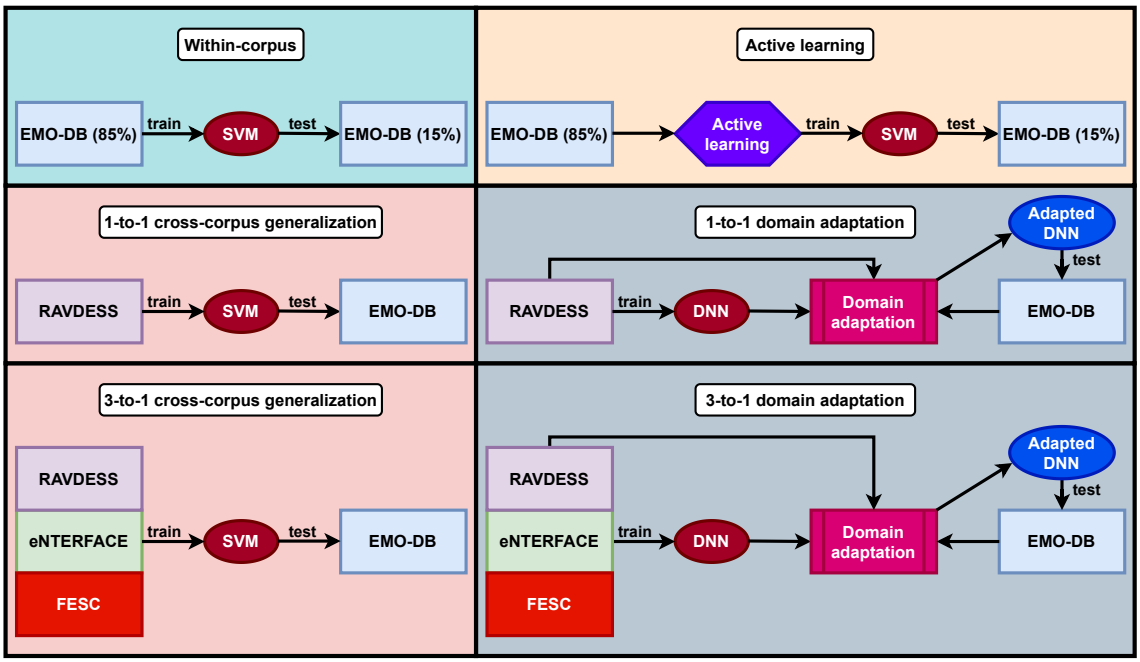


Figure 4.2. *A block diagram of the different simulation setup experiments for EMO-DB as the test corpus.*

The features for each corpus were normalized using z-score normalization in order to have zero mean and unit variance for each of the features at corpus level. As suggested by Schuller et al. [17], normalization at speaker level was also tested. However, initial experiments indicated that normalization in the context of a speaker very rarely worked better than normalization across the whole corpus. Thus, corpus normalization was used in the present study.

Following Schuller et al. [17], the emotional labels were mapped into the quarters of the valence-arousal plane as shown in Figure 4.3. This same emotional mapping has been used in multiple SER studies (e.g. [19], [20], [21], [23], [24], [25]). Although a few SER studies map disgust into high arousal (e.g. [22]), the majority of SER studies map disgust into low arousal. Therefore, disgust was mapped to low arousal in the present study. Furthermore, initial experiments suggested that an SVM classifier is able to discriminate arousal better if disgust was mapped into low arousal in the training data. The mapping of the emotional labels into the quarters of the valence-arousal plane was made to simplify the classification task into two binary classifications (see beginning of Section 2.2.3 for further details).

The primary evaluation measure of valence and arousal classification accuracy used in the present study is unweighted average recall (UAR %), occasionally referred to as unweighted accuracy, which is defined as the mean of the class-specific recalls [1]. UAR is a measure commonly used in PSP, since it ignores the number of occurrences of each class in the dataset, which can be highly imbalanced in PSP tasks. The chance level for UAR is the reciprocal of the total number of classes. For example, in two-class problems

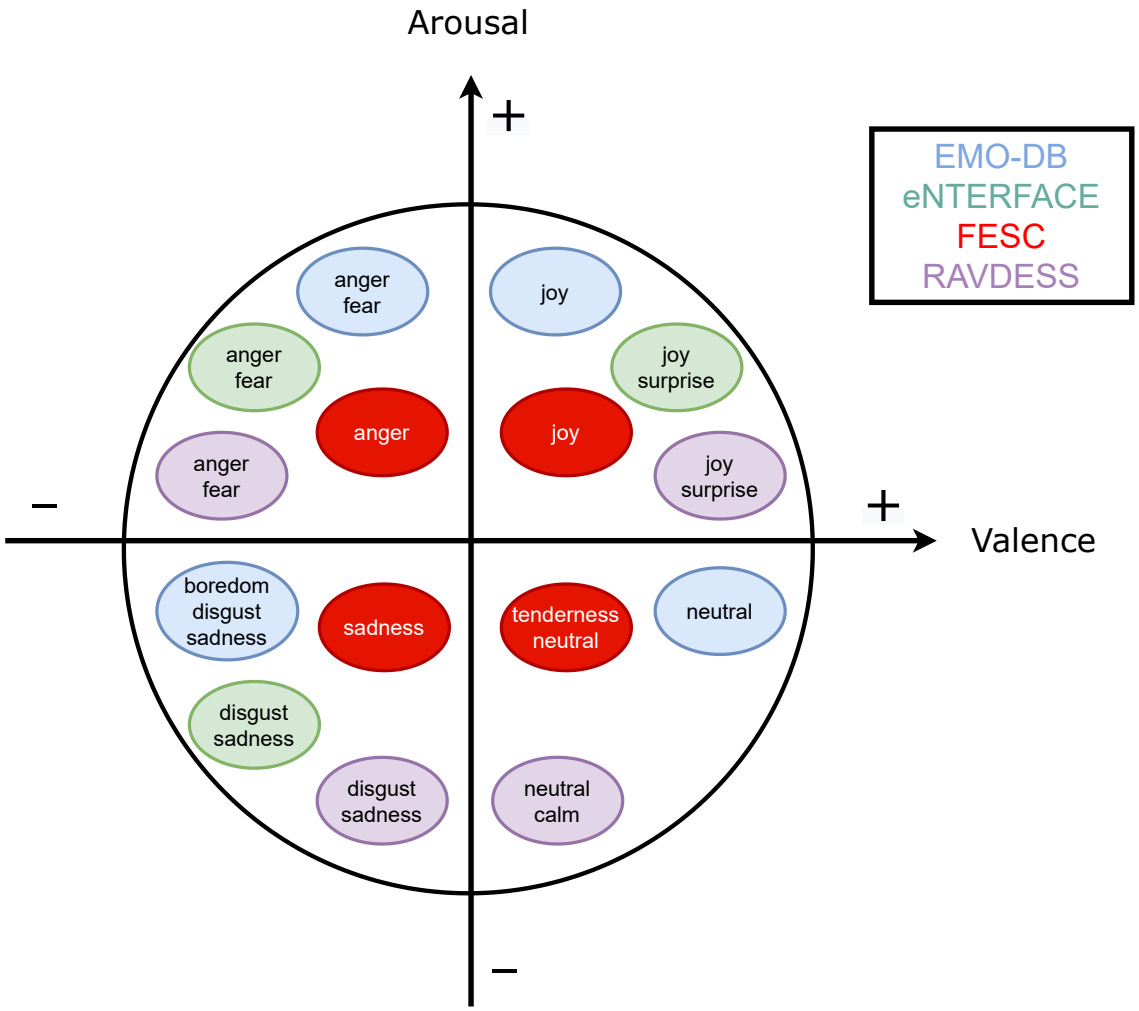


Figure 4.3. *The mapping of emotions into the quarters of the valence-arousal plane in the simulation setup.*

the chance level is 50% UAR, and in three-class problems the chance level is 33.3% UAR [1]. In addition, the standard error of the mean (SEM) [101] is reported for the AL-based experiments. The SEM is defined as

$$SEM = \frac{\sigma}{\sqrt{n}}\,, \tag{4.1}$$

where σ is the standard deviation of the classification accuracies, and n is the number of independent observations.

4.2.1 Within-corpus experiments

In the within-corpus experiments, each of the four simulation corpora were examined individually. The purpose of these experiments was to get an estimate of the accuracy that is achievable if labels for the entire dataset were available for classifier training and evalu-

ation. For the experiments, an SVM classifier with an RBF kernel was used together with log-mel, GeMAPS, and eGeMAPS features. The labeled data of each simulation corpus were randomly split into a training set and a test set in a ratio of 85:15. A binary classification for valence (positive/negative) and arousal (high/low) was performed separately.

Imbalances in the training data class distributions were countered by weighting each sample inversely proportional to its class frequency in the data. Optimal SVM hyperparameters were selected for each feature type and for both classification tasks individually. The SVM hyperparameters that were optimized were the box constraint, C, and the kernel scale parameter, γ. To find a suitable value for C and γ, a grid search was performed by testing all possible combinations of $C \in [0.6, 0.7, \ldots, 10.0] \cup [20.0, 40.0, \ldots, 200.0] \cup [500.0, 10^3, 10^4, 10^5, 10^6]$ and $\gamma \in \frac{1}{dim_f}[0.5, 0.6, \ldots, 4.0]$, where dim_f is the dimensionality of the features. These values were selected based on initial tests on the data. For each combination of C and γ, 5-fold cross-validation was performed on the training set using UAR as the accuracy metric to find the best-performing combination. Then, these optimal values for C and γ were used to train the SVM on the training data. Finally, this trained SVM was used to find the accuracy of the model on the test set.

4.2.2 Cross-corpus generalization

The cross-corpus generalization tests consisted of two separate settings: one-to-one and three-to-one cross-corpus generalization. For the settings, the former number refers to the number of corpora that are used for training, whereas the latter number refers to the number of corpora that are used for testing. For both settings, an SVM classifier with an RBF kernel was trained to discriminate between positive/negative valence and high/low arousal using all three different feature types, similar to the procedure described in Section 4.2.1. Again, class balance weights were used to address the imbalances in the labels. The purpose of these cross-corpus generalization experiments was to investigate the classification performance of a SER system when it is trained on one or several SER corpora and tested on some other SER corpus.

In the one-to-one setting, all of the different scenarios were examined where one SER corpus was used as the training set and another corpus was used as the test set. Now the best-performing hyperparameters were not found using grid search, but instead the optimal values for C and γ from the within-corpus experiments were used. The SVM classifier was trained using these hyperparameters on the training corpus and its performance was tested on the test corpus.

The three-to-one setting covered all scenarios in which three of the four simulation corpora were used as the training set and the fourth corpus was used as the test set. Here the optimal values for C and γ were found using the cross-validated grid search depicted in Section 4.2.1. The SVM classifier was then trained using these hyperparameters on

the three training corpora and tested on the test corpus.

4.2.3 Experiments using active learning

The AL-based experiments using MAL were conducted in a within-corpus manner. In pilot experiments regarding MAL, the aim was to experiment with different dissimilarity measures and to find a suitable value for k (see Section 3.1 for further details) by experimenting with the simulation corpora. Additionally, since the labels for NICU-A the were not available during these preliminary tests, rather than trying to search for methods which work the best for each simulation corpus separately, the core idea was to find and develop methods which perform well in general. Furthermore, the effect of different labeling budgets on the overall classification performance was inspected, bearing in mind that the premise of the labeling process of this thesis is that the annotations amount to only a small part of the data.

In pilot experiments, a suitable value for k was found to be $\frac{N}{3}$, where N is the number of samples in the corpus. This can be interpreted as meaning that the average size of the clusters is three. An important observation was that the clustering algorithm worked better with lower-dimensional features than with higher-dimensional features. Consequently, based on initial experiments the generally best-performing dissimilarity measure was found to be the Pearson distance (Equation 3.1) that was computed for 32-dimensional features which were compressed from the 600-dimensional log-mel representation using an autoencoder. The encoder part of the autoencoder consisted of three dense layers with output dimensionalities of 512, 512, and 32. These three layers were followed by ELU nonlinearities (Equation 2.25). For the first two layers, a dropout of 10% was used. The decoder part is simply a mirrored version of the encoder.

After finding out an optimal dissimilarity measure and an optimal value for k in the preliminary experiments, simulated annotations were performed for the simulation corpora. An autoencoder was trained separately for each simulation corpus using the MSE loss function (Equation 2.20). For the training, the same split into two sets from Section 4.2.1 was performed for the simulation corpora. The larger set was further split into a training set and a validation set so that the overall ratio between the training, validation, and test set was 70:15:15. Early stopping with patience of 300 epochs was used, and the best model according to the validation loss was chosen as the optimal autoencoder. The network was trained using a batch size of 1024 and using the Adam optimization algorithm [102] with a learning rate of 10^{-4}.

For the simulated annotations, labeling budgets of 3%, 6%, and 10% of the total samples in a corpus were tested. The same training samples as in Section 4.2.1 were used for computing the distance matrix and for performing the label predictions. Experiments with and without assigning the label of the medoid as the label of all the cluster members were

carried out. It should be noted that the simulated annotation process used in the present study assumes that the annotator does not give erroneous labels. Labels for both valence and arousal were given for each annotated sample in the simulated annotations.

After the simulated annotation process, the annotated samples were used as the training set for an SVM classifier with an RBF kernel. Similar to the aforementioned experiments, two SVMs were trained separately to distinguish between positive/negative valence and high/low arousal using class balance weights. All of the three feature types defined in Section 4.2 were compared separately and the optimal hyperparameters for the SVM were taken from the within-corpus experiments described in Section 4.2.1. The same test set as in the within-corpus experiments was then used to test the classification performance of the trained classifier.

4.2.4 Experiments using domain adaptation

The DA-based tests involving WDA were conducted in similar one-to-one and three-to-one settings as in the cross-corpus generalization experiments in Section 4.2.2. In the one-to-one adaptation setting, one SER corpus acted as the source data and another corpus acted as the target data. In the three-to-one adaptation setting, three of the four SER corpora acted as the source data and the fourth corpus acted as the target data. All of the different adaptation experiments were conducted for both valence and arousal separately.

In pilot experiments, different network structures for the feature extractor, the label classifier, and the domain classifier were experimented with. In a similar manner as in the AL-based experiments, the purpose was not to find optimal network structures for each setting separately, but instead the goal was to search for generally well-performing network structures for the simulation corpora. Also, different input features were experimented with. The best-performing features were found to be the 600-dimensional log-mel features. These log-mel features were used as the input for the feature extractor, F, which consisted of three dense layers with output dimensionalities of 512, 512, and 256. The first two of these layers were followed by a LReLU nonlinearity (Equation 2.24) and a dropout layer with a dropout of 40%. All of the three layers were followed by batch normalization. Instead of SVM classifiers used in the other simulation setup settings, the label classifier, C_L, was a neural network which consisted of three dense layers with output dimensionalities of 256, 256, and 2. The first two layers were followed by LReLU nonlinearities and a dropout of 30%. The last layer was followed by a softmax nonlinearity to get the class probabilities. The domain classifier, C_D, consisted of four dense layers with output dimensionalities of 512, 512, 256, and 1. The first three layers were followed by ReLU nonlinearities.

After finding well-performing network structures and features in the preliminary tests with

the simulation corpora, experiments were conducted using these network structures and features. Each simulation corpus was split into a training, validation, and test set with a ratio of 70:15:15. The same split was used as in the autoencoder training of Section 4.2.3. For the first stage of the adaptation process, i.e. training F and C_L on the source data to obtain the source model, the labeled training and validation sets of the source data were used for model training. The labeled test set of the source data was then used for model selection purposes. For the second stage of the adaptation process, i.e. adapting F for the target data, the training data was the full unlabeled source data and the unlabeled training set of the target data. To find optimal hyperparameters for the adaptation process, the labeled validation set of the target corpus was used to test the performance of the adaptation. Although it is possible to find optimal hyperparameters for the adaptation without the labeled validation set of the target corpus, this validation set was used to significantly speed up the hyperparameter selection process. The test data for the second stage of the adaptation process was the labeled test set of the target data.

For the first stage of the adaptation process in the one-to-one adaptation setting, the labeled training set and validation set of the source corpus was used to train F and C_L. Early stopping was used with a patience of 100 epochs to obtain the optimal source model. The early stopping criterion was the classification accuracy on the test set of the source corpus. A similar procedure was used with the feature extractor training for the three-to-one adaptation setting, with the only difference being that the training, validation, and test sets for the source data were the combined training, validation, and test sets of the three source corpora. The Adam optimizer was used with a learning rate of 10^{-4}. A batch size of 256 was used in all parts of the DA process. For each possible variant of the source data, a separate source model was trained for both valence and arousal. This model was then used for all of the DA experiments involving the same source data, including both the unsupervised and semi-supervised variant of WDA.

For the second stage of the adaptation process, C_D was trained for four minibatches in a row. For every fifth minibatch, the parameters of the adapted feature extractor, F_T, were updated. The parameters of C_D were updated with the RMSProp optimizer [103] as suggested by Arjovsky et al. in the original WGAN paper [68]. The Adam optimizer was used to update the parameters of F_T. For all experiments, a learning rate of $5 \cdot 10^{-5}$ was used for updating the parameters of C_D. For F_T, a learning rate of $5 \cdot 10^{-5}$ was used in the one-to-one adaptation setting, and a learning rate of $3 \cdot 10^{-5}$ was used in the three-to-one adaptation setting.

To assist C_D at the start of the adaptation process, a predefined number of head start iterations were given for C_D before its parameters and the parameters of F_T were updated in turns. In the one-to-one adaptation setting, the tests involving EMO-DB and eNTERFACE as the source corpus used 100 head start iterations. For the tests involving FESC and RAVDESS as the source corpus, 500 head start iterations were used. In all three-to-one

adaptation settings 1000 head start iterations were used.

Following [90], the unsupervised variant of WDA was trained until the first term in Equation 3.5 was saturated. For the semi-supervised variant of WDA, the labels of a small randomly selected subset of the target corpus training samples (5% of the size of the corpus) were utilized. This labeled subset was used to determine the classification accuracy of the combination of F_T and C_L after each epoch, and the model with the highest classification accuracy on this subset was then selected for testing. The labeled subset of the target corpus was also tested to be utilized in the adaptation loss functions (Equations 3.4 and 3.5) during the adaptation process, but it turned out to systematically worsen the adaptation results. Hence, this approach was left out of the semi-supervised WDA experiments. After optimal adaptation hyperparameters and model parameters were found, the performance of the adapted model was tested on the test set of the target corpus.

4.3 Experiments with NICU-A

Once functioning practices had been established on the simulation setup and the labels for NICU-A had been acquired, similar tests as in Section 4.2 were conducted on the new data. Again, all three feature types (log-mel, GeMAPS, and eGeMAPS) were used for all tests except DA-based experiments where only log-mel features were used based on their superior performance in pilot experiments. Since the labels for valence in NICU-A were categorized into three classes, the three label categories were merged into two for the cases of binary classification for valence. Considering that the researchers in the APPLE study were interested in the proportion of positive valence over other types of valence, the 'neutral' and 'negative' classes for valence were merged into 'neutral' for the two-class experiments. Thus, the binary classification with valence regarding NICU-A was between positive and neutral valence. Consequently, the mapping depicted in Figure 4.3 was modified accordingly for the simulation corpora for the cross-corpus generalization and DA tests involving NICU-A to include the utterances with label 'neutral' in the non-positive valence samples, and treating this class as the 'neutral' class.

4.3.1 Active learning experiments with NICU-A

In the AL-based experiments for NICU-A, optimal values for C and γ of the SVM classifier for each of the three feature types were determined using a grid search and a 5-fold cross validation for the labeled training data, similar to the procedure described in Section 4.2.1. This was done separately for both the labeled training set of 5198 samples and the extended training set of 33,979 samples including the cluster labels from MAL. Then, the optimal hyperparameter values were used to train an SVM model with an RBF kernel. Class balance weights were used in the SVM training process. The trained SVM model

was then used to determine the accuracy of the model on the gold standard data. In addition to the two binary decisions for valence and arousal, a three-class classification setting for valence (positive/neutral/negative) was also tested. For this three-class setting, the SVM was trained using the “one vs. all” multiclass SVM method.

4.3.2 Cross-corpus generalization experiments with NICU-A

For the cross-corpus generalization experiments, similarly to the cross-corpus experiments of Section 4.2.2, two settings were explored with NICU-A: one-to-one and four-to-one cross-corpus generalization. In the one-to-one setting, each of the four simulation corpora were used individually as the as the training set. In the four-to-one setting, all of the four simulation corpora were used as the training set. The gold standard data of NICU-A was used as the test set in both settings. For the two settings, optimal values for C and γ for each of the three feature types were determined using a similar grid search as in the within-corpus experiments. Again, an SVM classifier with an RBF kernel was used, and class balance weights were utilized in the SVM training process.

4.3.3 Domain adaptation experiments with NICU-A

For the DA-based experiments, one-to-one and four-to-one adaptation settings were examined. In each of the one-to-one adaptation settings, one of the four simulation corpora was used as the source corpus to-be-adapted to NICU-A. A similar data split and source model training procedure as described in Section 4.2.4 was used for the simulation corpora for both valence and arousal using the same hyperparameter values. Then, the feature extractor of the source model was adapted and tested in a similar manner as in the experiments of Section 4.2.4. Similarly, in the four-to-one adaptation setting, all of the four simulation corpora were used as the source data and NICU-A was used as the target data. The training, validation, and test set for the four source corpora was the combination of the respective sets for each source corpus.

The training data for the second stage of the adaptation process was the full unlabeled data from the source data together with the 96,615 unlabeled training samples of NICU-A. To find optimal hyperparameters for the adaptation process, the 5,198 labeled training samples were used as the validation data. The unsupervised and semi-supervised variants of WDA were trained according to the procedure described in Section 4.2.4, with the classification accuracy on the 5,198 labeled training samples being used as the model selection criterion for the semi-supervised variant.

For the second stage of the adaptation process in the one-to-one adaptation settings, a learning rate of $5 \cdot 10^{-5}$ was used in all experiments. Exceptions to this rule were when FESC was used as the source corpus for valence and with RAVDESS as the source

corpus for arousal, where a learning rate of $7 \cdot 10^{-5}$ was used for both due to its superior performance with the validation data. The number of head start iterations was 600 in all cases, except with FESC as the source corpus, where 800 head start iterations were used due to better performance with the validation data. For the four-to-one adaptation settings, 1000 head start iterations were used with learning rates of $7 \cdot 10^{-5}$ for valence and $6 \cdot 10^{-5}$ for arousal. All other hyperparameters in the DA-based experiments were the same as described in Section 4.2.4. After the second stage of the adaptation process, the performance of the adapted model was then evaluated on the gold standard data.

5. RESULTS

In this chapter, the results of the experiments described in Chapter 4 are presented and discussed. First, the results for the simulation setup experiments are reported in Section 5.1, after which the results for the experiments with NICU-A are presented in Section 5.2. Finally, the results of Sections 5.1 and 5.2 are summed up and discussed in Section 5.3. All of the given accuracies are in UAR (%).

5.1 Results on the simulation setup

The results of the experiments that were conducted on the simulation corpora are presented in this section. The results of the within-corpus experiments are reported in Section 5.1.1, followed by the results of the cross-corpus generalization experiments in Section 5.1.2. Then, the results of the AL-based experiments are given in Section 5.1.3, followed by the results of the DA-based experiments in 5.1.4. Finally, Section 5.1.5 summarizes the results of the simulation setup.

5.1.1 Results for the within-corpus experiments in the simulation setup

The results for the within-corpus experiments on the simulation corpora are presented in Table 5.1. The reported accuracies are the classification accuracies on the test set of each corpus. Additionally, the mean value for each of the three feature types is reported for both valence and arousal.

In the within-corpus experiments, the log-mel features had the best average performance for both valence and arousal, although GeMAPS and eGeMAPS features came relatively close despite their small dimensionality. For valence, the GeMAPS features achieved 2.0 percentage points UAR lower mean accuracy than the log-mel features, whereas for arousal, the eGeMAPS features achieved only 0.1 percentage points UAR lower mean accuracy than the log-mel features. For the corpus-specific experiments, the classification of arousal with RAVDESS was the only irregularity, since, in that case, the eGeMAPS features outperformed the log-mel features. The distinctly lower classification accuracy for eNTERFACE (approximately 10 percentage points UAR lower accuracy for valence

Table 5.1. *The test set accuracies for the within-corpus experiments of the simulation setup. The highest accuracy feature-wise is highlighted.*

UAR (%)	Valence			Arousal		
	log-mel	***GeMAPS***	***eGeMAPS***	***log-mel***	***GeMAPS***	***eGeMAPS***
EMO-DB	**80.0**	79.6	72.6	**96.2**	94.0	93.9
eNTERFACE	**70.9**	67.6	69.9	**74.3**	74.2	74.2
FESC	**87.5**	85.8	85.9	**91.2**	85.7	90.2
RAVDESS	**84.8**	82.1	84.4	86.5	83.5	**89.2**
mean	*80.8*	*78.8*	*78.2*	*87.0*	*84.4*	*86.9*

and approximately 13 percentage points UAR lower accuracy for arousal when compared to the mean accuracy) may be explained by the different nature of the corpus compared to the other corpora. The speakers in eNTERFACE were not professional actors and the expressed emotions were not predefined, but instead the emotions were evoked from listening to short stories. Moreover, these speakers were from multiple different nationalities. Another observation is that arousal is easier to classify than valence, which has been noted in, e.g., [5] as well.

The results of Table 5.1 are comparable to those of earlier literature. For example, the best result for arousal with EMO-DB reaches the level of 96.2% UAR which is similar to the best results reported in earlier literature for the same corpus (e.g., 97.8% UAR by [32] and approximately 97% UAR by [17]). However, it should be noted that the division of data intro training and test sets is not exactly identical between the studies. As can be observed from the results, SER is a difficult task even when labeled data are available.

5.1.2 Results for the cross-corpus generalization experiments in the simulation setup

Tables 5.2 and 5.3 show the results for the one-to-one and three-to-one cross-corpus generalization experiments (Section 4.2.2), respectively. What can be observed from the one-to-one cross-corpus generalization experiments in Table 5.2 is that there is a significant drop in the classification accuracies compared to the within-corpus results of Table 5.1, which indicates that cross-corpus SER is a difficult task. For example, the best one-to-one cross-corpus generalization classification accuracy for EMO-DB when classifying arousal is approximately 20 percentage points UAR lower than the respective result from the within-corpus experiments. Additionally, what can be seen is that it is difficult to know beforehand that which training corpus and which features work the best for a given corpus. The lower-dimensional GeMAPS and eGeMAPS features worked generally better than log-mel features. The only two exceptions were the classification of arousal with FESC and EMO-DB, where FESC with log-mel features was the best

training corpus for EMO-DB and vice versa. However, the choice between GeMAPS and eGeMAPS features and the optimal training corpus is still a difficult task even if log-mel features were ruled out. Furthermore, there seems to be correlation between better performance and the train and test corpora belonging to Germanic languages. As can be seen from Table 5.2, in most of the tests the best training corpus for a corpus with a Germanic language is also a corpus with a Germanic language.

Table 5.2. *The classification accuracies for the simulation corpora in the one-to-one cross-corpus generalization experiments of the simulation setup. The best classification accuracy for each corpus is highlighted and the mean value for each of the feature types for each simulation corpus is given for both valence and arousal.*

UAR (%)		Valence			Arousal		
Test corpus	**Training corpus**	***log-mel***	***GeMAPS***	***eGeMAPS***	***log-mel***	***GeMAPS***	***eGeMAPS***
EMO-DB	eNTERFACE	56.8	58.6	**61.7**	50.2	62.3	55.0
	FESC	53.8	55.8	56.9	**76.5**	67.5	69.0
	RAVDESS	59.1	54.8	56.6	65.2	74.2	72.4
	mean	*56.6*	*56.4*	*58.4*	*64.0*	*68.0*	*65.4*
eNTERFACE	EMO-DB	52.1	53.1	55.4	52.5	60.1	57.8
	FESC	48.1	43.3	44.4	62.4	61.2	58.6
	RAVDESS	54.0	53.7	**56.1**	60.1	**63.6**	62.9
	mean	*51.4*	*50.0*	*51.9*	*58.4*	*61.6*	*59.8*
FESC	EMO-DB	55.7	**61.9**	60.3	**68.9**	65.2	66.0
	eNTERFACE	48.4	48.1	50.7	65.9	61.0	48.7
	RAVDESS	56.9	60.8	60.1	68.0	63.0	61.5
	mean	*53.7*	*56.9*	*57.0*	*67.6*	*63.1*	*58.7*
RAVDESS	EMO-DB	53.3	53.3	55.5	62.9	**75.7**	73.2
	eNTERFACE	53.4	52.3	53.9	61.5	71.9	64.6
	FESC	54.5	**63.7**	60.8	69.8	70.2	71.2
	mean	*53.8*	*56.4*	*56.7*	*64.7*	*72.6*	*69.7*

The three-to-one cross-corpus generalization results of Table 5.3 further demonstrate the difficulty of cross-corpus SER discussed in the previous paragraph. As can be seen, when using three SER corpora as the training set the best classification accuracies drop even more from those depicted in Table 5.2. For example with FESC, the best result for valence was 61.9% UAR in the one-to-one experiments, whereas it was 58.7 % UAR in the three-to-one experiments. As with the one-to-one experiments, the classification results are on many occasions close to or below chance level, especially when classifying valence. On average, the eGeMAPS features performed best when classifying valence (mean accuracy of 57.7% UAR), and the log-mel features gave the best results when classifying arousal (mean accuracy of 68.3% UAR).

These three-to-one cross-corpus generalization results were also in line with similar earlier studies in the field. For example when classifying valence with EMO-DB as the test corpus using eGeMAPS features, the achieved classification accuracy is 59.8% UAR,

Table 5.3. *The test corpus accuracies for the three-to-one cross-corpus generalization experiments of the simulation setup. The highest accuracy feature-wise is highlighted and the mean value for each of the feature types is given.*

UAR (%)	Valence			Arousal		
Test corpus	***log-mel***	***GeMAPS***	***eGeMAPS***	***log-mel***	***GeMAPS***	***eGeMAPS***
EMO-DB	53.6	59.3	**59.8**	68.1	**69.6**	67.5
eNTERFACE	49.7	50.0	**51.6**	**62.3**	61.7	60.1
FESC	54.8	56.2	**58.7**	**71.4**	63.5	60.8
RAVDESS	57.4	60.2	**60.8**	**71.5**	70.4	70.5
mean	*53.9*	*56.4*	*57.7*	*68.3*	*66.3*	*64.7*

whereas an accuracy of 60.5% UAR was achieved by [24] with EMO-DB using the same features. However, it should be pointed out that the training corpora between the studies were different. Also, for example when classifying arousal with eNTERFACE as the test corpus using log-mel features, the obtained accuracy of 62.3% UAR is close the median accuracy of approximately 64% UAR by [17] and the accuracy of 58.2% UAR by [20], even though the training corpora and the used features were different between the studies.

When comparing the best results of the one-to-one and three-to-one cross-corpus generalization experiments, the only case when an SVM trained with three SER corpora beat the SVMs trained on individual corpora was when classifying arousal with FESC when using the log-mel features. However, when comparing the results of the three-to-one experiments and the mean values of the one-to-one results, the three-to-one setting gives better results overall. This suggests that a training set consisting of multiple SER corpora yields a better classifier in general than a training set consisting of an individual randomly chosen SER corpus. Furthermore, the three-to-one setting is a safer choice than the one-to-one setting since it is difficult to know beforehand that which corpus and which features should be used in the one-to-one setting to obtain better performance than in the three-to-one setting. A further study would be needed to better understand the selection of optimal features and an optimal training corpus or corpora for cross-corpus generalization. What can be deduced from both cross-corpus generalization experiments is that there is clearly a need for other methods such as AL or DA in order to obtain better results, particularly with more complex SER data such as real-life recordings where the aim is to interpret the emotional state of the speaker.

5.1.3 Results for the active learning experiments in the simulation setup

Tables 5.4 and 5.5 present the results for the AL experiments of Section 4.2.3 for valence and arousal, respectively. The given classification accuracies for different labeling

budgets are reported both with and without the cluster labels. Since the medoids are initialized at random which affects the results, the average of the classification accuracies of five consecutive experiments is reported together with the SEM (Equation 4.1) where $n = 5$ in our case.

Table 5.4. *The results for the AL tests for valence in the simulation setup, both with and without the cluster labels. The mean accuracy of five experiments and the SEM are given, where the highest mean accuracy for each labeling budget is highlighted.*

UAR (%), mean accuracy ± SEM				Valence		
Corpus	**Data annotated**	**Annotated samples**	**Cluster labels**	***log-mel***	***GeMAPS***	***eGeMAPS***
EMO-DB	3%	16	No	59.9 ± 1.7	54.2 ± 1.1	54.8 ± 1.1
			Yes	**61.5 ± 1.8**	56.4 ± 2.1	58.3 ± 1.6
	6%	32	No	66.7 ± 2.2	**69.6 ± 1.4**	61.8 ± 0.9
			Yes	67.0 ± 0.9	66.9 ± 1.4	66.3 ± 2.0
	10%	54	No	**70.6 ± 2.2**	69.6 ± 2.0	66.0 ± 2.3
			Yes	67.6 ± 2.0	63.9 ± 0.9	65.3 ± 2.2
eNTERFACE	3%	39	No	50.9 ± 0.2	49.3 ± 0.8	49.3 ± 0.6
			Yes	**54.1 ± 0.7**	50.3 ± 0.3	51.9 ± 0.5
	6%	77	No	54.6 ± 0.9	50.9 ± 0.4	52.8 ± 1.0
			Yes	56.5 ± 0.9	53.8 ± 0.4	**57.1 ± 0.7**
	10%	129	No	55.7 ± 0.9	54.1 ± 0.9	56.0 ± 0.4
			Yes	**60.2 ± 0.6**	52.8 ± 0.3	56.1 ± 0.8
FESC	3%	128	No	64.7 ± 0.6	66.6 ± 0.6	66.1 ± 0.6
			Yes	65.7 ± 0.1	68.7 ± 0.5	**69.3 ± 0.2**
	6%	255	No	70.9 ± 0.4	71.1 ± 0.4	70.5 ± 0.5
			Yes	71.5 ± 0.7	**73.1 ± 0.5**	73.0 ± 0.9
	10%	425	No	74.4 ± 0.2	74.7 ± 0.5	**76.3 ± 0.5**
			Yes	73.6 ± 0.2	74.8 ± 0.5	73.8 ± 0.4
RAVDESS	3%	43	No	**63.3 ± 1.5**	58.3 ± 1.1	54.5 ± 0.9
			Yes	62.7 ± 1.2	60.2 ± 1.8	60.8 ± 2.1
	6%	86	No	63.5 ± 0.7	64.0 ± 0.8	63.4 ± 0.4
			Yes	63.9 ± 0.7	**64.1 ± 0.3**	62.1 ± 0.5
	10%	144	No	**71.5 ± 0.9**	68.3 ± 1.0	66.8 ± 0.8
			Yes	70.6 ± 0.9	66.8 ± 0.8	66.9 ± 0.5

The results of the AL tests in Tables 5.4 and 5.5 show the effect of the increase in the number of annotations. Generally, when comparing labeling budgets of 3% and 6%, there is a significant increase (e.g., approximately 5.2 percentage points UAR on average for valence) in classification accuracy when the number of annotations is doubled. In addition, there is a major increase in classification accuracy when comparing labeling budgets of 6% and 10% (e.g., approximately 2.5 percentage points UAR on average for valence), albeit this increase is not as large as when comparing labeling budgets of 3% and 6%. It is also worth pointing out that the classification accuracies do not increase monotonically when increasing the number of annotations. In some of the tests there was no benefit when increasing the labeling budget from 3% to 6%, but in all cases there was an increase in accuracy when increasing the labeling budget from 3% to 10%. This

suggests that for some corpora and with certain features there is a critical number of labels that needs to be exceeded in order to increase the classification accuracy when using MAL and the chosen dissimilarity metric.

Table 5.5. *The results for the AL tests for arousal in the simulation setup, both with and without the cluster labels. The mean accuracy of five experiments and the SEM are given, where the highest mean accuracy for each labeling budget is highlighted.*

UAR (%), mean accuracy ± SEM				Arousal		
Corpus	**Data annotated**	**Annotated samples**	**Cluster labels**	***log-mel***	***GeMAPS***	***eGeMAPS***
EMO-DB	3%	16	No	83.5 ± 0.8	83.3 ± 1.0	82.0 ± 1.3
			Yes	**88.5 ± 0.2**	79.2 ± 0.9	80.1 ± 0.8
	6%	32	No	90.1 ± 0.5	82.2 ± 0.7	81.6 ± 0.7
			Yes	**93.8 ± 0.6**	87.7 ± 1.1	89.0 ± 0.9
	10%	54	No	**94.1 ± 0.8**	85.6 ± 1.0	84.2 ± 1.3
			Yes	92.3 ± 0.6	85.0 ± 0.3	85.5 ± 0.5
eNTERFACE	3%	39	No	56.6 ± 1.6	61.6 ± 0.2	**65.6 ± 0.8**
			Yes	58.8 ± 1.4	64.8 ± 1.4	63.0 ± 1.6
	6%	77	No	55.3 ± 0.9	**63.8 ± 0.9**	61.6 ± 1.1
			Yes	58.7 ± 0.1	61.5 ± 0.6	61.4 ± 0.5
	10%	129	No	63.5 ± 1.1	68.6 ± 0.5	**69.3 ± 0.3**
			Yes	66.2 ± 0.4	68.9 ± 0.5	66.2 ± 0.8
FESC	3%	128	No	71.8 ± 0.4	65.3 ± 0.7	72.4 ± 1.1
			Yes	73.2 ± 0.1	70.0 ± 0.6	**73.8 ± 0.6**
	6%	255	No	77.5 ± 0.4	73.5 ± 0.5	76.7 ± 0.7
			Yes	**79.6 ± 0.3**	74.3 ± 0.7	78.0 ± 0.5
	10%	425	No	82.0 ± 0.4	74.3 ± 0.3	78.3 ± 0.2
			Yes	**83.4 ± 0.2**	77.9 ± 0.3	80.2 ± 0.2
RAVDESS	3%	43	No	**77.8 ± 0.4**	76.8 ± 0.2	75.7 ± 1.1
			Yes	76.6 ± 0.8	76.1 ± 0.3	75.0 ± 0.6
	6%	86	No	**79.7 ± 0.1**	77.0 ± 0.8	78.1 ± 0.8
			Yes	77.1 ± 0.8	74.9 ± 0.3	73.7 ± 0.8
	10%	144	No	**82.1 ± 0.4**	78.9 ± 0.7	78.6 ± 0.4
			Yes	80.2 ± 0.6	77.1 ± 0.5	76.9 ± 0.9

For the majority of the AL experiments, the use of all medoid cluster samples as labeled data (instead of just using the medoids as samples) led to accuracy gains. This gain was most noticeable and frequent with smaller labeling budgets. For example with valence, the mean classification accuracy increased by approximately 2.3 percentage points UAR when cluster labels were used. However, already when the number of annotations was 10% of the total number of samples in a corpus, more than half of the accuracies were higher without the cluster labels. Furthermore, the gain in accuracy was mostly negligible for the cases when it was beneficial to use the cluster labels when the labeling budget was 10%. This indicates that in many cases the cluster labels are able to provide an increase in accuracy, but most likely in situations when the labeling budget is very small.

An important observation is that the use of cluster labels did not work well with all of the simulation corpora. This can be seen most evidently with the experiments conducted on

RAVDESS where the use of the cluster labels provided inferior results for the majority of classifications compared to not using the cluster labels. One explanation to this may be the choice of the dissimilarity measure that was used with MAL, which was the Pearson distance for the autoencoder-compressed log-mel features. Since one of the principal ideas of the AL experiments was to find a dissimilarity measure that performs well in general with the simulation corpora, it is possible that the chosen dissimilarity measure does not work well with all SER corpora. An optimal solution accuracy-wise would be to find the best-performing dissimilarity measure for all corpora and for both valence and arousal individually.

Overall, when comparing different features in the AL experiments the results varied largely depending on the corpus, the classification task, and the number of annotations. What can be observed is that the AL results did not follow the within-corpus experiments of Table 5.1, where the log-mel features performed the best in almost all experiments for both valence and arousal. This implies that each of the features needs to be tested separately in order to find the best-performing classifier for the AL-annotated data, which can easily lead to problems with overfitting when only a small amount of AL data are available for training and performance evaluation. Also, the comparison between the AL experiments and the within-corpus baseline results reveals that MAL with a labeling budget of 10% gives on average approximately 11 percentage points UAR lower accuracy for valence and 6 percentage points UAR lower accuracy for arousal. On the other hand, the comparison between MAL with a labeling budget of 10% and the cross-corpus generalization experiments gives on average approximately 7 percentage points UAR higher accuracy for valence and approximately 10 percentage points UAR higher accuracy for arousal for MAL.

Furthermore, we tested the classification accuracy for valence and arousal using labeling budgets of $L \in [2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 30, 40]$% of the number of samples in each simulation corpus to better showcase the dependency between the labeling budget and the classification accuracy. This was done for the log-mel features, which performed the best on average for the AL tests. The figures representing these tests are presented in Figures 5.1 and 5.2. In the figures, the mean classification accuracy of five experiments and the SEM is reported to counter the effect of randomness in the medoid initialization. Also, the within-corpus classification accuracy for the log-mel features is included in the figures to give a reference accuracy for cases when the labels for the whole training data are available. The cluster labels of MAL are not used in the experiments regarding the figures.

Again, it can be clearly seen from Figures 5.1 and 5.2 that on average the classification accuracy grows most rapidly when the labeling budget is small, and the growth starts to slow down as the number of annotations increases. This further emphasizes the importance of increasing the number of annotations when the number of annotated samples

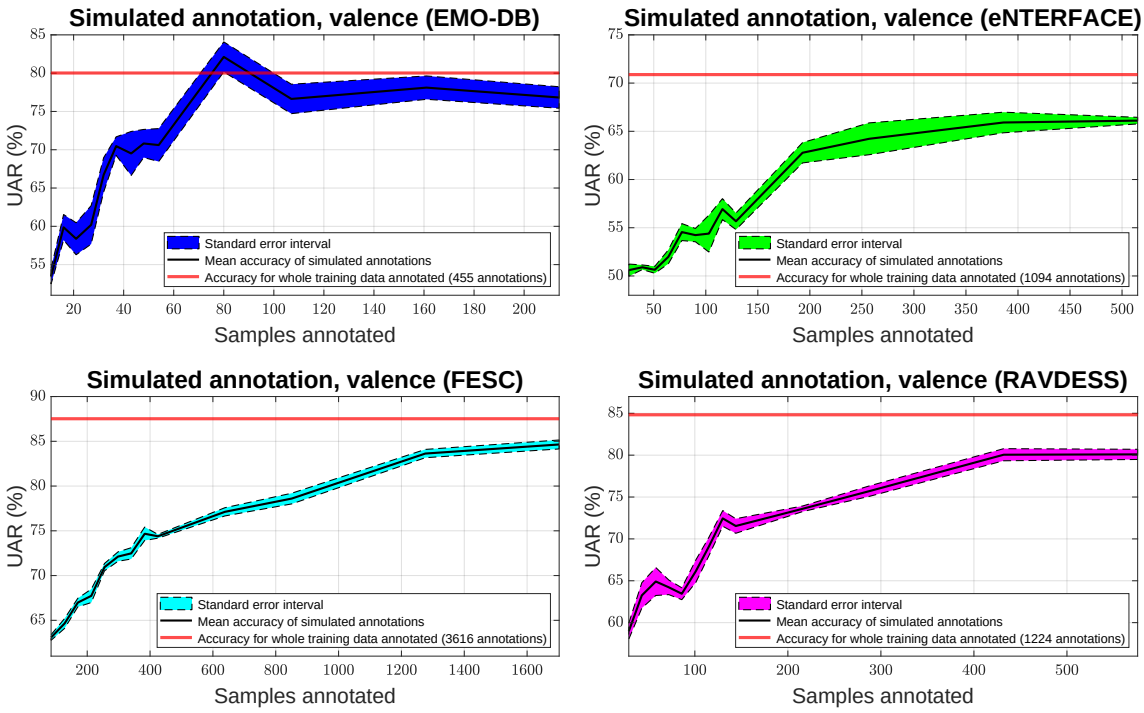


Figure 5.1. *The classification accuracy on the test set for valence with the simulation corpora using different labeling budgets.*

is small in order to obtain a better classifier. When investigating further, for the majority of the simulation corpora a labeling budget of approximately 5% is already sufficient to provide a higher accuracy for both valence and arousal than the best individual results found in the cross-corpus generalization experiments in Section 5.1.2. Overall, a labeling budget of 40% of the number of samples in a corpus leads to classifier performance that is close to the within-corpus baseline results.

An important observation from Figures 5.1 and 5.2 is that, in all of the cases, the accuracy does not grow monotonically when the number of annotated samples increases. Another unintuitive finding is that in many cases the SEM is larger with more annotations than it is with less annotations. Intuitively, the classification accuracy should grow monotonically when the number of annotations is increased and the variance of the accuracies should decrease when more samples are annotated. However, this may be due to the selected dissimilarity measure and the properties of the MAL method. As already stated before, the dissimilarity measure used in the present study was selected because it performed the best on average for all simulation corpora and for both valence and arousal. However, the chosen metric does not perform equally well for all of the simulation corpora. By using the chosen metric together with farthest-first traversal, the initial medoids chosen by the algorithm should correspond to data points which lie in the distance space as diversely as possible. Since the chosen distance function does not perform well with some of the corpora, sometimes the medoids chosen by the algorithm are not well representative of the test data sample distribution after the algorithm has chosen the most obviously dis-

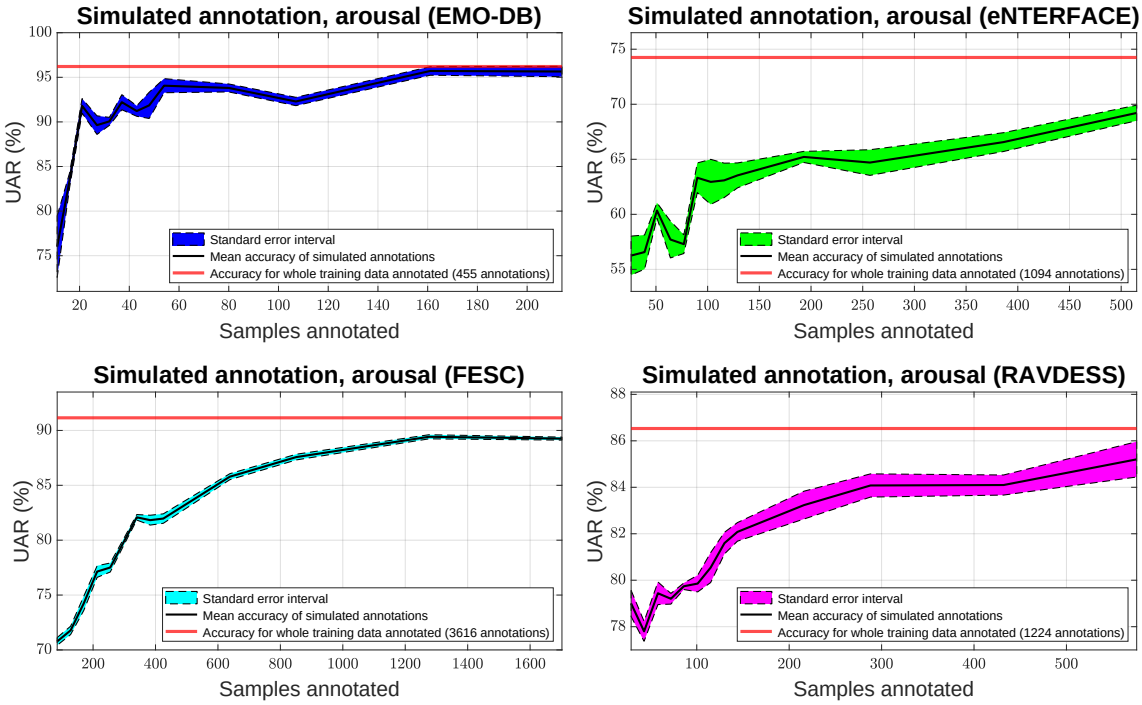


Figure 5.2. *The classification accuracy on the test set for arousal with the simulation corpora using different labeling budgets.*

tinct data points according to the distance measure. In other words, MAL systematically selects poor medoid candidates after a number of samples. This can perhaps be seen most evidently with the classification of valence with EMO-DB in Figure 5.1, where MAL is able to select an optimal subset of the training data which even outperforms the baseline result, after which the accuracy starts to decrease as the number of annotations grows.

At other times, the first medoids chosen by the algorithm are not well representative of the test data, but then after some number of samples the chosen medoids start to be representative of the test data. An example of this can be seen with the classification of arousal with eNTERFACE in Figure 5.2, where, at first, the accuracy plot does not grow monotonically and the variance of the accuracies varies considerably. However, after approximately 100 labeled samples the plot starts to grow almost monotonically and the variance of the accuracies does not change significantly as the number on annotated samples increases. Out of the simulation corpora, the selected dissimilarity measure performed the best with FESC. This can be seen both from the behavior of the mean value (almost monotonically increasing for both valence and arousal) and the relatively small values for the SEM throughout the plots regarding FESC in Figures 5.1 and 5.2.

5.1.4 Results for the domain adaptation experiments in the simulation setup

The results for the one-to-one and three-to-one DA experiments are shown in Tables 5.6 and 5.7, respectively. As can be seen from both tables, the semi-supervised variant of WDA consistently beat the unsupervised variant in all DA experiments. This implies that the criterion to select the optimal adapted model is better in the semi-supervised variant. In the unsupervised variant, there is no specific method to determine when to stop the adaptation process other than the saturation of the first term in Equation 3.5. Although the saturation of this term is one way of describing the state of the adaptation process, it does not guarantee an optimal point at which to stop the process since the optimal state of the adaptation can also be before or after the saturation of the term. Moreover, the adaptation process is not guaranteed to succeed even if the term saturates. However, by carefully selecting the hyperparameters for the adaptation process, it is possible to obtain a successful adaptation for the vast majority of adaptation iterations. In the semi-supervised variant, the classification accuracy on the subset of the target corpus data is clearly a better criterion to select an optimal adapted model during the adaptation process.

Table 5.6. *The DA test results for the one-to-one adaptation setting in the simulation setup, both for the unsupervised and the semi-supervised variant of WDA. The best classification accuracy for each target corpus is highlighted for valence and arousal.*

UAR (%)		Valence		Arousal	
Target corpus	**Source corpus**	***unsupervised***	***semi-supervised***	***unsupervised***	***semi-supervised***
EMO-DB	eNTERFACE	65.9	**67.1**	62.1	64.2
	FESC	58.9	60.3	78.1	78.9
	RAVDESS	65.8	66.3	79.7	**82.6**
eNTERFACE	EMO-DB	49.0	51.8	59.0	61.1
	FESC	43.6	45.4	59.8	60.3
	RAVDESS	50.1	**52.7**	64.0	**64.4**
FESC	EMO-DB	59.6	60.4	70.7	73.4
	eNTERFACE	46.1	49.4	69.3	70.1
	RAVDESS	62.5	**63.6**	72.2	**74.2**
RAVDESS	EMO-DB	52.1	52.6	73.5	75.8
	eNTERFACE	56.2	57.9	74.3	76.6
	FESC	61.8	**62.0**	75.5	**77.1**

Comparing the DA experiments and the cross-corpus generalization experiments, the one-to-one adaptation results for the unsupervised variant of WDA surpassed the results of the one-to-one cross-corpus generalization tests in almost all the scenarios. Additionally, the three-to-one adaptation results for the unsupervised variant were better than the three-to-one cross-corpus generalization results in the majority of different cross-corpus generalization settings. This suggests that DA can be a practical tool to assist in the

utilization of some SER corpora to provide classification models for other SER corpora when labels are scarce or there are no labels available.

Table 5.7. *The DA test results for the three-to-one adaptation setting in the simulation setup, both for the unsupervised and the semi-supervised variant of WDA. The best classification accuracy for each target corpus is highlighted for valence and arousal.*

UAR (%)	Valence		Arousal	
Target corpus	*unsupervised*	*semi-supervised*	*unsupervised*	*semi-supervised*
EMO-DB	55.4	**56.3**	69.5	**70.3**
eNTERFACE	51.9	**52.3**	61.7	**62.2**
FESC	62.4	**63.5**	72.8	**74.1**
RAVDESS	62.0	**62.4**	77.7	**78.0**

However, when comparing the DA results and the AL results, AL with a labeling budget of 3% was almost always better than the semi-supervised variant of WDA which in turn was consistently better than the unsupervised variant. An important sidenote is that the semi-supervised variant of WDA utilized a labeling budget of 5%. This suggests that it is often more beneficial to annotate data using MAL and use these annotations directly for training a model instead of annotating data for DA purposes.

It should be noted that the DA results presented here are not the most optimal results corpus-wise. The aim was not to search for optimal network structures for each adaptation setting individually. Instead, the goal was to test the performance of DA using network structures which perform well in all settings. Also, it should be noted that the inclusion of eNTERFACE in the three-to-one adaptation setting worsened the results slightly in all of the three-to-one DA experiments where eNTERFACE was included in the source corpora. However, the corpus was decided not to be left out of the three-to-one DA experiments to avoid cherry-picking of the most suitable corpora, since the optimal source corpora for any new corpus cannot be known beforehand.

5.1.5 Summary of the results of the simulation setup

Figure 5.3 summarizes the results of the simulation setup for the log-mel features. These features were selected for the summary since they were used in all of the tests, and, in addition, they were the best-performing features on average. For simplicity, the classification accuracies for valence and arousal have been merged and averaged for each distinct method. The results involving MAL are shown as a function of annotated data samples, whereas the other results are shown as a reference.

The one-to-one scenarios for the cross-corpus generalization and DA tests have been left out of the figure, since their three-to-one counterparts gave better results on average. An interesting observation is that, although the majority of AL tests with a labeling budget of

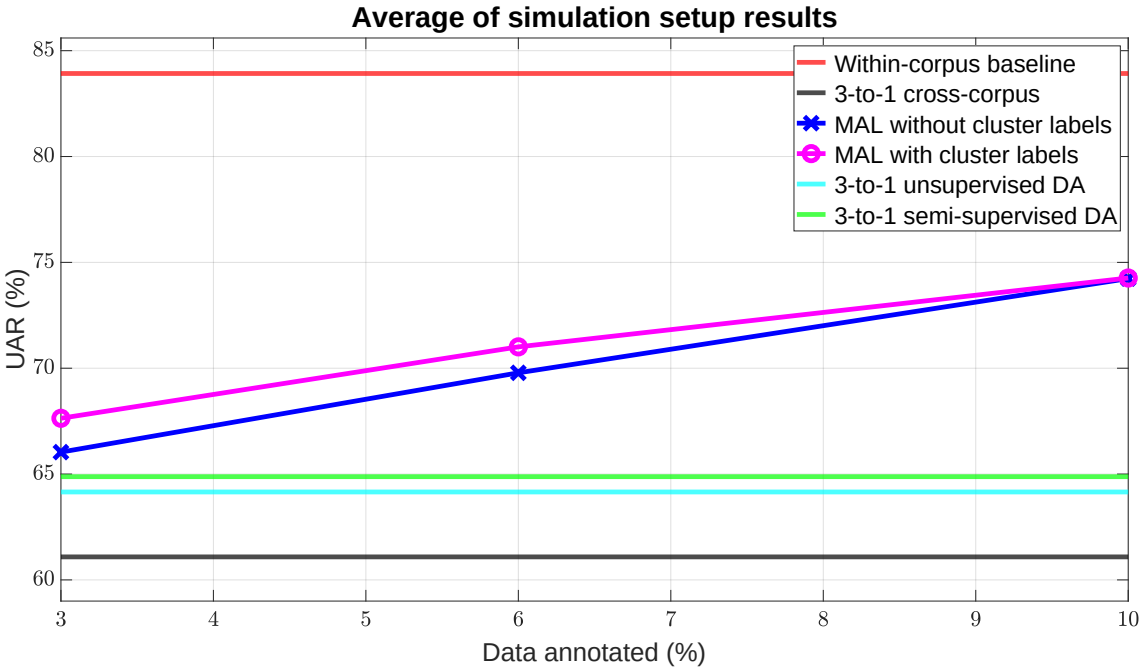


Figure 5.3. *The averaged simulation setup results for valence and arousal for the log-mel features. The results involving MAL are shown as a function of the percentage of annotated data samples, other results are shown as a reference.*

10% led to better performance without the cluster labels, the average accuracy of the AL tests with a labeling budget of 10% is higher with the cluster labels than without them. Based on the results of the simulation setup altogether, MAL is superior compared to cross-corpus generalization and WDA when there are annotations available and number of annotations is scarce compared to the overall size of the dataset. When comparing WDA and cross-corpus generalization, WDA provided better results on average. Furthermore, the unsupervised variant of WDA worked better than cross-corpus generalization when there were no labels available.

5.2 Results on NICU-A

The results for the AL-based experiments for NICU-A are shown in Table 5.8. The given accuracies are the classification accuracies on the gold standard data for the three feature types in three different classification scenarios.

In the AL-based experiments, contrary to the simulation setup results of Tables 5.4 and 5.5, the binary classification accuracy for valence was (on average) higher than it was for arousal. This may be explained by the fact that the kappa score for arousal was lower than the kappa score of valence in both the training data and the gold standard data. For both valence and arousal, the GeMAPS and eGeMAPS features achieved a higher mean accuracy than the log-mel features. Moreover, it can be observed that the use of cluster labels in MAL increased the classification accuracy in binary classifications for

Table 5.8. *The classification accuracies for the AL-based experiments for NICU-A. The highest accuracy in each scenario is highlighted.*

UAR (%)		Valence (2 classes)	Valence (3 classes)	Arousal
Cluster labels	Features			
No	log-mel	70.9	44.2	68.5
	GeMAPS	71.0	48.8	**69.3**
	eGeMAPS	71.9	**53.9**	65.8
Yes	log-mel	68.2	48.1	67.0
	GeMAPS	**73.4**	45.5	68.9
	eGeMAPS	72.9	45.0	67.6

the GeMAPS and eGeMAPS features with an average of approximately 1.2 percentage points UAR. Furthermore, the three-class classification accuracy for valence was higher than chance level (mean 47.6% UAR) but considerably lower than that of the two-class classification for valence (mean 71.4% UAR). In addition to the classification task being more difficult in itself, the low number of negative training samples in the three-class setting was not only highly imbalanced compared to the number of neutral and positive training samples (see Table 4.1), but also insufficient for the SVM classifier to learn a proper classification bound between the three classes.

Table 5.9. *The classification accuracies for the cross-corpus generalization experiments for NICU-A as the test corpus. The highest accuracy for valence and arousal is highlighted.*

UAR (%)	Valence			Arousal		
Training corpus	*log-mel*	*GeMAPS*	*eGeMAPS*	*log-mel*	*GeMAPS*	*eGeMAPS*
EMO-DB	48.5	53.8	53.4	64.1	63.7	62.7
eNTERFACE	56.8	52.7	50.2	63.1	64.3	64.1
FESC	45.3	**57.3**	54.9	56.3	68.3	**70.8**
RAVDESS	50.4	53.8	53.3	64.3	62.0	58.7
All simulation corpora	42.9	54.9	56.8	61.3	64.4	65.5

Table 5.9 presents the one-to-one and four-to-one cross-corpus generalization results for all three feature types. The cross-corpus generalization results show that there is a severe mismatch between the statistical properties of the data of the simulation corpora and NICU-A when classifying valence. For example, this can be observed when comparing the best result for valence in the cross-corpus generalization experiments (57.3% UAR) and the best result for valence in the AL-based experiments (73.4% UAR). For valence, the lower-dimensional GeMAPS and eGeMAPS features clearly outperformed the log-mel features but the results were distinctly lower than in the AL-based experiments. For arousal, there was plenty of variation in the best-performing features for the different training corpora. Surprisingly, the classification accuracy for arousal with FESC as the

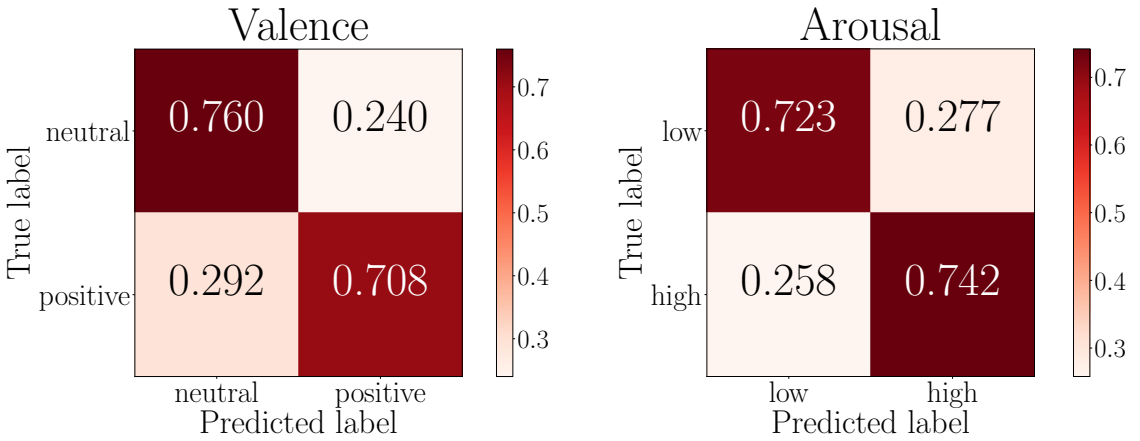
training corpus together with the eGeMAPS features (70.8% UAR) was higher than the best accuracy from the AL-based experiments (69.3% UAR). This implies that the statistical properties of FESC together with the eGeMAPS features had a smaller mismatch to the gold standard data compared to the labeled training data of NICU-A using the eGeMAPS features. However, it is likely that this might have partially happened due to pure coincidence since the respective accuracies for GeMAPS and log-mel features are higher in the results of Table 5.8 than they are in the results of FESC in Table 5.9.

Table 5.10. *The classification accuracies for the DA-based experiments for NICU-A as the target corpus. The highest accuracy for valence and arousal is highlighted.*

UAR (%)	Valence		Arousal	
Source corpus	***unsupervised***	***semi-supervised***	***unsupervised***	***semi-supervised***
EMO-DB	49.7	51.3	71.0	**73.2**
eNTERFACE	57.0	**58.0**	67.2	68.6
FESC	46.9	47.4	61.6	63.1
RAVDESS	57.1	57.7	66.5	68.4
All simulation corpora	53.2	53.5	71.0	71.3

Table 5.10 shows the results of the one-to-one and four-to-one DA experiments for NICU-A. The DA-based classification results are better than the results of the cross-corpus generalization experiments for the most part, but, e.g., the classification model for valence with EMO-DB and FESC was clearly not able to adapt properly to NICU-A data. However, considering that the input features to the DA model were the log-mel features, both the unsupervised and the semi-supervised variant of WDA were consistently able to outperform the results of the cross-corpus generalization experiments regarding the log-mel features in Table 5.9. Similar to the results of the DA-based experiments in the simulation setup, the semi-supervised variant of WDA was consistently better than the unsupervised variant with an average gain of approximately 1.1 percentage points UAR. For valence, the DA method did not provide a major improvement over the classification accuracies of the cross-corpus generalization experiments, but for arousal some of the model adaptations yielded a significant improvement over the AL and cross-corpus generalization results of Tables 5.8 and 5.9, respectively. Unexpectedly, the best adapted model for arousal outperformed all other methods tested with NICU-A with an accuracy of 73.2% UAR.

For valence, the best-performing classification model for the gold standard data was an SVM that was trained with GeMAPS features using the cluster labels from MAL. For arousal, the best-performing model was a neural network that was adapted to NICU-A data using semi-supervised WDA with EMO-DB as the source corpus. These models achieved classification accuracies of 73.4% UAR and 73.2% UAR for valence and arousal, respectively. To further inspect the reliability of these models, the confusion matrices for the models are presented in Figure 5.4. As can be seen from the figure, there are no



(a) Confusion matrix for valence.

(b) Confusion matrix for arousal.

Figure 5.4. *The normalized confusion matrices for valence and arousal using the best-performing classification models for NICU-A. For valence, the best model was an SVM trained with GeMAPS features using the cluster labels from MAL. For arousal, the best model was a neural network that was adapted using semi-supervised WDA with EMO-DB as the source corpus.*

major differences between the accuracies on both the diagonal and the off-diagonal. This indicates that these models do not systematically favor one label over the other when performing predictions.

5.3 Discussion of the results

Cross-corpus generalization, AL, and DA were compared for SER in the simulation setup using four already existing SER corpora in cases with little to no labeled data in the test corpus. The simplest of these methods, cross-corpus generalization, did not work very well in the experiments. As can be seen from the summary of the simulation setup results of Figure 5.3, training a classification model with one or multiple SER corpora and using the model on some new corpus is not generally the best solution to address the absence of labeled data in SER. This was also the case in the extensive cross-corpus SER study conducted by Schuller et al. [17], where cross-corpus generalization led to mediocre SER performance. What was observed in the present experiments was that for cross-corpus generalization, it is better (on average) to combine multiple corpora than to use one corpus as the training set when applying the model to new data from a new domain.

By using alternative methods such as AL and DA, the performance of SER models in the simulation setup was increased in cases with little to no labeled data compared to the results of cross-corpus generalization. For situations when there are no labeled data available, the unsupervised variant of WDA was the best alternative out of the tested methods. On the other hand, when a small number of labeled samples were available, MAL turned out to be the best method. By looking at the summary of the simulation setup

results of Figure 5.3, MAL outperformed both the unsupervised variant of WDA and its better-performing semi-supervised variant already with a labeling budget of 3%. What can also be observed from Figure 5.3 is that the performance of MAL increases as the number of annotated samples increases.

When comparing the results of the simulation setup and the results on NICU-A, there were notable differences in the results of the two setups. Although the results for valence were better in the AL-based tests than in the cross-corpus generalization and DA tests, the same trend did not follow for arousal. In the tests regarding arousal for NICU-A, the best-performing models in cross-corpus generalization and DA provided better classification performance than the best-performing models in the AL-based tests.

For the best-performing model in cross-corpus generalization outperforming the best model in the AL-based tests with NICU-A, it is probable that this might have partially happened due to pure coincidence based on the results of Table 5.9. However, there might be several reasons why the best-performing DA model outperformed the best AL-based model, contrary to the results of the simulation setup. One explaining factor for this might be that the MAL algorithm benefits from consistent labels. Annotating similar clusters differently during the annotation process of MAL might lead to strong confusion between the annotated classes, which was also observed in [78]. Therefore, it might be that the relatively low inter-rater reliability score with NICU-A (Section 4.1.6) lowered the classification performance with MAL more than it did with WDA. Another explaining factor for the differences between the findings of the simulation experiments and the experiments with NICU-A could be that the expressed emotions are realistic in NICU-A, whereas the expressed emotions were acted in the majority of the simulation corpora. This might potentially degrade the performance of MAL more compared to WDA because realistic emotions might be more difficult to be grouped together into similar clusters than acted emotions. This is because emotional categories in acted emotions are more evidently distinguishable from each other than they are with realistic emotions [3, 16].

Regarding the best-performing classification models on NICU-A, the best model for valence was an SVM that was trained with GeMAPS features using the cluster labels from MAL. For arousal, the best model was a neural network classifier that was adapted to NICU-A data using semi-supervised WDA with EMO-DB as the source corpus. Based on the results of the best-performing models on NICU-A in Figure 5.4, it seems that these models are suitable for analyzing the emotional content of speech for the child-centered audio recordings recorded in a NICU at Turku University Hospital.

As for the most suitable features, it was observed that the log-mel features achieved the best accuracy on average. The performances of the low-dimensional GeMAPS and eGeMAPS features were comparable to, and, in many cases, even better than the higher-dimensional log-mel features. This was also in line with the results of Eyben et al. [32],

who found that the low-dimensional GeMAPS and eGeMAPS features surpassed or came close to the performance of many high-dimensional feature sets in SER.

When comparing the time that it takes to fine-tune neural network architectures and hyperparameters of WDA and the time that it takes to annotate samples for MAL, it is considerably faster to annotate enough samples (approximately 3% of the size of the corpus) for any of the simulation corpora for MAL to outperform WDA than it is to fine-tune WDA to work well with any given target corpus. However, with very large datasets such as NICU-A with 129,007 unlabeled samples, it takes approximately twice the time to annotate 3% of the data than it takes to fine-tune neural network architectures and hyperparameters for WDA. This states that if there is not enough time and resources to annotate a significant number of samples, WDA is the best alternative of the tested methods. On the other hand, already with a very moderate human annotation effort, MAL leads to similar or better performance than cross-corpus generalization and WDA. What should be noted is that although a labeled subset of each target corpus was used in the present experiments to validate the success of the adaptation for WDA, it is possible to find optimal hyperparameters for WDA even without labeled data from the target corpus, albeit it takes a somewhat longer time to fine-tune the hyperparameters without any labeled data from the target corpus.

Finally, it is important to point out that although many unsupervised and semi-supervised AL and DA methods are considered to have little to no time expenses (e.g., [72], [78], [88], [104], [105], [106], [107]), what is almost always left out of consideration in AL and DA literature is that the development and implementation of these models can be very time-consuming. Experiences from the present study suggest that better results could be achieved if the time spent on developing and fine-tuning these models would be used to simply annotate data of the target corpus for use in traditional supervised learning methods. However, testing and developing unsupervised and semi-supervised AL and DA methods is essential for these methods to be further developed.

6. GENERAL DISCUSSION

In the present work, a system which is able to perform large-scale analysis of the emotional content of real-life child-centered audio recordings from a NICU was created. Cross-corpus generalization, AL, and DA were examined as mechanisms for deploying a SER system for a novel dataset that has no a priori labels for emotion classes of interest. For situations when there are no labeled data available for the target corpus, the WGAN-based DA method used in the present experiments outperformed the cross-corpus generalization approach. When classifying valence, with a moderate number of samples labeled, the k-medoids clustering-based AL method used in the present study was superior compared to both cross-corpus generalization and DA. This applied both to simulations conducted on already existing SER corpora and to experiments conducted on the primary audio material of the study, NICU-A. However, the same trend followed for the classification of arousal only in the simulations but not for the experiments on NICU-A, where DA resulted in slightly better results compared to AL. Furthermore, the results showcase that the earlier proposed MAL [78] for AL and WDA [90] for DA can be successfully applied to real-life SER scenarios.

In order to further improve the cross-corpus generalization and DA performance of the present experiments, a separate study should be conducted on how to select optimal training or source corpora for the specific methods used in the present experiments. Additionally, the cross-corpus generalization experiments revealed that it is in general better to combine multiple SER corpora than it is to use only one corpus as training set. Therefore, a larger number of SER corpora as the training set than what was used in the present experiments should also be tested. In addition, utilizing pretrained SER models should be further investigated.

Other future improvements are to test alternative AL and DA methods. For example, CNN-based image processing DA methods could be applied directly for multidimensional audio feature representations such as log-mel spectrograms. Also, other feature representations than the features used in the present experiments should be studied. Furthermore, what should be tested in the simulation setup is that different levels of noise could be added to the simulated annotations in the AL method, since the current setup assumed that the annotator is always correct. Thereby e.g. the dissimilarity metric in the AL method could be further developed to be more robust to labeling noise.

One observation with NICU-A was that the annotations had a relatively low inter-rater agreement rate. Although the annotation task for the emotional content of real-life audio recordings is in itself difficult, there is room for improvement in the annotation process of the present study. To obtain more consistent labels between different annotators in related future studies, more thorough and unequivocal instructions on how to determine that which label should be given for each utterance should be given for each annotator. In addition, the annotators should discuss together about samples that have turned out to be difficult to annotate in order to find a common policy on how to annotate these kinds of samples. A comprehensive set of audio samples and their respective ground truth labels that would be given as self-study material for each annotator before performing annotations could also help in establishing common ground for the annotation process between each annotator.

For future SER research, larger annotated datasets are required in order to better utilize recent state-of-the-art models which are highly data-driven [11]. Even though creating large SER datasets manually is difficult and expensive, improving existing machine-learning methods and developing new reliable automated tools for audio analysis can be a solution to obtain large-scale annotated SER corpora [4, 11]. Also, realistic large-scale data should be used more often when developing new SER methods since results on small-scale acted data from clean recording environments can be highly optimistic compared to real-life performance [3, 17, 40]. Besides, some models that are created using acted speech might not even work with real-life data since, in addition to different recording conditions, there is a severe mismatch between the way emotions are portrayed in acted speech and realistic speech [41, 42, 43].

REFERENCES

[1] A. Batliner and B. Schuller, *Computational Paralinguistics: Emotion, Affect and Personality in Speech and Language Processing*. New York: John Wiley & Sons, Incorporated, 2013.

[2] L. Rabiner and R. Schafer, *Introduction to Digital Speech Processing*, ser. Foundations and Trends in Technology. Now, 2007, vol. 1, pp. 1–194.

[3] A. Batliner, B. Schuller, D. Seppi, S. Steidl, L. Devillers, L. Vidrascu, T. Vogt, V. Aharonson, and N. Amir, "The Automatic Recognition of Emotions in Speech", in *Emotion-Oriented Systems*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 71–99.

[4] 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.

[5] B. Schuller, F. Weninger, Y. Zhang, F. Ringeval, A. Batliner, S. Steidl, F. Eyben, E. Marchi, A. Vinciarelli, K. Scherer, M. Chetouani, and M. Mortillaro, "Affective and Behavioural Computing: Lessons Learnt from the First Computational Paralinguistics Challenge", *Computer Speech & Language*, vol. 53, pp. 156–180, 2019.

[6] E. Ståhlberg-Forsen, A. Aija, B. Kaasik, R. Latva, S. Ahlqvist-Björkroth, L. Toome, L. Lehtonen, and S. Stolt, "The validity of the Language Environment Analysis system in two neonatal intensive care units", *Acta Paediatrica*, 2021.

[7] J. E. Swain, "Stress-Sensitive Parental Brain Systems Regulate Emotion Response and Motivate Sensitive Child Care", in *Early Vocal Contact and Preterm Infant Brain Development: Bridging the Gaps Between Research and Practice*, M. Filippa, P. Kuhn, and B. Westrup, Eds. Springer International Publishing, 2017, pp. 241–269.

[8] T. M. Mitchell, *Machine Learning*, 1st ed. USA: McGraw-Hill, Inc., 1997.

[9] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, 4th ed. Academic Press, 2008.

[10] J. A. Goetz, *IEEE standard dictionary of electrical and electronics terms*, 3. ed. New York, NY: Institute of Electrical and Electronics Engineers, 1984.

[11] B. Schuller, Y. Zhang, and F. Weninger, "Three recent trends in Paralinguistics on the way to omniscient machine intelligence", *Journal on Multimodal User Interfaces*, vol. 12, pp. 273–283, 2018.

[12] Y. Bengio, A. Courville, and P. Vincent, “Representation Learning: A Review and New Perspectives”, *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, pp. 1798–1828, Aug. 2013.

[13] A. Batliner, S. Steidl, B. Schuller, D. Seppi, T. Vogt, J. Wagner, L. Devillers, L. Vidrascu, V. Aharonson, L. Kessous, and N. Amir, “Whodunnit – Searching for the most important feature types signalling emotion-related user states in speech”, *Computer Speech & Language*, vol. 25, no. 1, pp. 4–28, 2011.

[14] F. Eyben, F. Weninger, F. Gross, and B. Schuller, “Recent developments in openSMILE, the Munich open-source multimedia feature extractor”, in *Proceedings of the 2013 ACM Multimedia Conference*, ACM, Oct. 2013, pp. 835–838.

[15] J. Pohjalainen, O. Räsänen, and S. Kadioglu, “Feature selection methods and their combinations in high-dimensional classification of speaker likability, intelligibility and personality traits”, *Computer speech & language*, vol. 29, no. 1, pp. 145–171, 2015.

[16] 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.

[17] B. Schuller, B. Vlasenko, F. Eyben, M. Wöllmer, A. Stuhlsatz, A. Wendemuth, and G. Rigoll, “Cross-Corpus Acoustic Emotion Recognition: Variances and Strategies”, *IEEE Transactions on Affective Computing*, vol. 1, no. 2, pp. 119–131, 2010.

[18] J. A. Russell, “A Circumplex Model of Affect”, *Journal of Personality and Social Psychology*, vol. 39, no. 6, pp. 1161–1178, Dec. 1980.

[19] B. Schuller, Z. Zhang, F. Weninger, and G. Rigoll, “Using Multiple Databases for Training in Emotion Recognition: To Unite or to Vote?”, in *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, Jan. 2011, pp. 1553–1556.

[20] Z. Zhang, F. Weninger, M. Wöllmer, and B. Schuller, “Unsupervised learning in cross-corpus acoustic emotion recognition”, in *2011 IEEE Workshop on Automatic Speech Recognition Understanding*, 2011, pp. 523–528.

[21] 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 *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 4818–4822.

[22] A. Tursunov, S. Kwon, and H.-S. Pang, “Discriminating Emotions in the Valence Dimension from Speech Using Timbre Features”, *Applied sciences*, vol. 9, no. 12, Article 2470, 2019.

[23] H. Sagha, J. Deng, M. Gavryukova, J. Han, and B. Schuller, “Cross lingual speech emotion recognition using canonical correlation analysis on principal component subspace”, in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 5800–5804.

[24] S. Latif, J. Qadir, and M. Bilal, “Unsupervised Adversarial Domain Adaptation for Cross-Lingual Speech Emotion Recognition”, in *2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII)*, Sep. 2019, pp. 732–737.

[25] Q. Mao, G. Xu, W. Xue, J. Gou, and Y. Zhan, “Learning emotion-discriminative and domain-invariant features for domain adaptation in speech emotion recognition”, *Speech Communication*, vol. 93, pp. 1–10, 2017.

[26] G. Trigeorgis, F. Ringeval, R. Brueckner, E. Marchi, M. A. Nicolaou, B. Schuller, and S. Zafeiriou, “Adieu features? End-to-end speech emotion recognition using a deep convolutional recurrent network”, in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 5200–5204.

[27] M. Abdelwahab and C. Busso, “Active Learning for Speech Emotion Recognition Using Deep Neural Network”, in *2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII)*, 2019, pp. 1–7.

[28] P. Song and W. Zheng, “Feature Selection Based Transfer Subspace Learning for Speech Emotion Recognition”, *IEEE Transactions on Affective Computing*, vol. 11, no. 3, pp. 373–382, 2020.

[29] M. Abdelwahab and C. Busso, “Domain Adversarial for Acoustic Emotion Recognition”, *IEEE/ACM Trans. Audio, Speech and Language Processing*, vol. 26, no. 12, pp. 2423–2435, Dec. 2018.

[30] J. Deng, X. Xu, Z. Zhang, S. Frühholz, and B. Schuller, “Semisupervised Autoencoders for Speech Emotion Recognition”, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 1, pp. 31–43, 2018.

[31] ——, “Universum Autoencoder-Based Domain Adaptation for Speech Emotion Recognition”, *IEEE Signal Processing Letters*, vol. 24, no. 4, pp. 500–504, 2017.

[32] F. Eyben, K. R. Scherer, B. W. Schuller, J. Sundberg, E. André, C. Busso, L. Y. Devillers, J. Epps, P. Laukka, S. S. Narayanan, and K. P. Truong, “The Geneva Minimalistic Acoustic Parameter Set (GeMAPS) for Voice Research and Affective Computing”, *IEEE Transactions on Affective Computing*, vol. 7, no. 2, pp. 190–202, 2016.

[33] N. Cummins, S. Amiriparian, G. Hagerer, A. Batliner, S. Steidl, and B. W. Schuller, “An Image-Based Deep Spectrum Feature Representation for the Recognition of Emotional Speech”, in *Proceedings of the 25th ACM international conference on Multimedia*, Mountain View, California, USA: Association for Computing Machinery, 2017, pp. 478–484.

[34] M. Chen, X. He, J. Yang, and H. Zhang, “3-D Convolutional Recurrent Neural Networks With Attention Model for Speech Emotion Recognition”, *IEEE Signal Processing Letters*, vol. 25, no. 10, pp. 1440–1444, 2018.

[35] C. Etienne, G. Fidanza, A. Petrovskii, and B. Schmauch, “CNN+LSTM Architecture for Speech Emotion Recognition with Data Augmentation”, in *Workshop on Speech, Music and Mind*, Sep. 2018, pp. 21–25.

[36] Z. Zhao, Y. Zheng, Z. Zhang, H. Wang, Y. Zhao, and C. Li, “Exploring Spatio-Temporal Representations by Integrating Attention-based Bidirectional-LSTM-RNNs and FCNs for Speech Emotion Recognition”, in *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2018, pp. 272–276.

[37] B. Zhang, Y. Kong, G. Essl, and E. M. Provost, “f-Similarity Preservation Loss for Soft Labels: A Demonstration on Cross-Corpus Speech Emotion Recognition”, *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 5725–5732, Jul. 2019.

[38] M. Abdelwahab and C. Busso, “Study of Dense Network Approaches for Speech Emotion Recognition”, in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5084–5088.

[39] J. Jia, S. Zhou, Y. Yin, B. Wu, W. Chen, F. Meng, and Y. Wang, “Inferring Emotions From Large-Scale Internet Voice Data”, *IEEE Transactions on Multimedia*, vol. 21, no. 7, pp. 1853–1866, 2019.

[40] W. Fan, X. Xu, X. Xing, W. Chen, and D. Huang, “LSSED: a large-scale dataset and benchmark for speech emotion recognition”, *arXiv preprint arXiv: 2102.01754*, 2021.

[41] J. Wilting, E. Krahmer, and M. Swerts, “Real vs. Acted Emotional Speech”, in *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2006, pp. 805–808.

[42] F. Burkhardt, A. Paeschke, M. Rolfes, W. Sendlmeier, and B. Weiss, “A database of German emotional speech”, in *9th European Conference on Speech Communication and Technology*, vol. 5, Jan. 2005, pp. 1517–1520.

[43] M. Airas and P. Alku, “Emotions in Vowel Segments of Continuous Speech: Analysis of the Glottal Flow Using the Normalised Amplitude Quotient”, *Phonetica*, vol. 63, pp. 26–46, Feb. 2006.

[44] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational Analysis of Sound Scenes and Events*, 1st ed. Cham: Springer Verlag, 2018.

[45] B. Boser, I. Guyon, and V. Vapnik, “A Training Algorithm for Optimal Margin Classifiers”, in *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, ACM Press, 1992, pp. 144–152.

[46] V. Vapnik and A. Chervonenkis, “On a class of perceptrons”, *Avtomatika i Telemekhanika*, vol. 25, Jan. 1964.

[47] B. Schölkopf and A. Smola, *Learning with Kernels - Support Vector Machines, Regularization, Optimization and Beyond*. MIT Press, Cambridge, MA, Jan. 2001, vol. 98.

[48] I. Steinwart, “Consistency of support vector machines and other regularized kernel classifiers”, *IEEE Transactions on Information Theory*, vol. 51, no. 1, pp. 128–142, 2005.

[49] H. Drucker, C. J. C. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, “Support Vector Regression Machines”, in *Advances in Neural Information Processing Systems*, MIT Press, 1997, pp. 155–161.

[50] V. Vapnik, *The nature of statistical learning theory*, 2nd ed., ser. Statistics for engineering and information science. New York (N.Y.): Springer.

[51] C.-W. Hsu and C.-J. Lin, “A comparison of methods for multiclass support vector machines”, *IEEE transactions on neural networks*, vol. 13, no. 2, pp. 415–425, 2002.

[52] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain”, *Psychological review*, vol. 65, no. 6, pp. 386–408, 1958.

[53] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, `http://www.deeplearningbook.org`.

[54] C. Sammut and G. Webb, Eds., *Encyclopedia of Machine Learning*, 1st ed. 2010. New York, NY: Springer US.

[55] M. Minsky and S. Papert, *Perceptrons: An Introduction to Computational Geometry*. Cambridge, MA, USA: MIT Press, 1969.

[56] X. Glorot, A. Bordes, and Y. Bengio, “Deep Sparse Rectifier Neural Networks”, in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, vol. 15, Apr. 2011, pp. 315–323.

[57] A. L. Maas, A. Y. Hannun, and A. Y. Ng, “Rectifier Nonlinearities Improve Neural Network Acoustic Models”, in *Proceedings of the 30th International Conference on Machine Learning (ICML)*, Atlanta, Georgia, USA, Jun. 2013.

[58] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, “Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)”, in *4th International Conference on Learning Representations (ICLR)*, San Juan, Puerto Rico, May 2016.

[59] K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators”, *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[60] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2. ed. New York: Springer, 2009.

[61] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, “Backpropagation Applied to Handwritten Zip Code Recognition”, *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[62] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors”, *Nature (London)*, vol. 323, no. 6088, pp. 533–536, 1986.

[63] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory”, *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[64] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation”, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734.

[65] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling”, *Neural Information Processing Systems (NIPS) 2014 Workshop on Deep Learning, December*, 2014.

[66] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative Adversarial Nets”, in *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS)*, Montreal, Canada: MIT Press, 2014, pp. 2672–2680.

[67] M. Arjovsky and L. Bottou, “Towards principled methods for training generative adversarial networks”, in *5th International Conference on Learning Representations (ICLR)*, Palais des Congrés Neptune, Toulon, France, Apr. 2017.

[68] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein Generative Adversarial Networks”, in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, D. Precup and Y. W. Teh, Eds., vol. 70, International Convention Centre, Sydney, Australia: PMLR, Aug. 2017, pp. 214–223.

[69] B. Settles, “Active Learning Literature Survey”, University of Wisconsin–Madison, Computer Sciences Technical Report 1648, 2009.

[70] F. Olsson, “A literature survey of active machine learning in the context of natural language processing”, Swedish Institute of Computer Science, Technical Report T2009:06, 2009.

[71] B. Settles, “From Theories to Queries: Active Learning in Practice”, in *Proceedings of Machine Learning Research*, vol. 16, Sardinia, Italy, May 2011, pp. 1–18.

[72] R. Schumann and I. Rehbein, “Active Learning via Membership Query Synthesis for Semi-Supervised Sentence Classification”, in *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 472–481.

[73] M.-R. Bouguelia, S. Nowaczyk, K. C. Santosh, and A. Verikas, “Agreeing to disagree: active learning with noisy labels without crowdsourcing”, *International journal of machine learning and cybernetics*, vol. 9, no. 8, pp. 1307–1319, 2017.

[74] Z. Zhang and B. Schuller, “Active Learning by Sparse Instance Tracking and Classifier Confidence in Acoustic Emotion Recognition”, in *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2012, pp. 362–365.

[75] Z. Zhao and X. Ma, “Active Learning for Speech Emotion Recognition Using Conditional Random Fields”, in *2013 14th ACIS International Conference on Software*

Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, 2013, pp. 127–131.

[76] J. Zhu, H. Wang, T. Yao, and B. K. Tsou, “Active Learning with Sampling by Uncertainty and Density for Word Sense Disambiguation and Text Classification”, in *Proceedings of the 22nd International Conference on Computational Linguistics*, vol. 1, Manchester, United Kingdom: Association for Computational Linguistics, 2008, pp. 1137–1144.

[77] G. Riccardi and D. Hakkani-Tür, “Active learning: Theory and Applications to Automatic Speech Recognition”, *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 4, pp. 504–511, Jul. 2005.

[78] Z. Shuyang, T. Heittola, and T. Virtanen, “Active learning for sound event classification by clustering unlabeled data”, in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 751–755.

[79] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, “A theory of learning from different domains”, *Machine learning*, vol. 79, no. 1-2, pp. 151–175, 2009.

[80] S. Ben-David and R. Urner, “On the Hardness of Domain Adaptation and the Utility of Unlabeled Target Samples”, in *Algorithmic Learning Theory*, vol. 7568, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 139–153.

[81] J. G. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N. V. Chawla, and F. Herrera, “A unifying view on dataset shift in classification”, *Pattern recognition*, vol. 45, no. 1, pp. 521–530, 2012.

[82] C. Cortes, Y. Mansour, and M. Mohri, “Learning Bounds for Importance Weighting”, in *Advances in Neural Information Processing Systems*, vol. 23, Curran Associates, Inc., 2010, pp. 442–450.

[83] M. Sugiyama and M. Kawanabe, *Machine Learning in Non-Stationary Environments: Introduction to Covariate Shift Adaptation*. Cambridge: MIT Press, 2012.

[84] S. Sun, B. Zhang, L. Xie, and Y. Zhang, “An unsupervised deep domain adaptation approach for robust speech recognition”, *Neurocomputing (Amsterdam)*, vol. 257, pp. 79–87, 2017.

[85] N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy, “Optimal Transport for Domain Adaptation”, *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 9, pp. 1853–1865, 2017.

[86] S. Ben-david, J. Blitzer, K. Crammer, and O. Pereira, “Analysis of representations for domain adaptation”, in *Proceedings of the 20th International Conference on Neural Information Processing Systems (NIPS)*, vol. 19, MIT Press, 2007, pp. 137–144.

[87] W. M. Kouw and M. Loog, “A Review of Domain Adaptation without Target Labels”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 3, pp. 766–785, 2021.

[88] J. R. Finkel and C. D. Manning, “Hierarchical Bayesian Domain Adaptation”, in *Proceedings of the 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Boulder, Colorado: Association for Computational Linguistics, Jun. 2009, pp. 602–610.

[89] Ó. Pérez and M. Sánchez-Montañés, “A New Learning Strategy for Classification Problems with Different Training and Test Distributions”, in *Computational and Ambient Intelligence*, vol. 4507, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 178–185.

[90] K. Drossos, P. Magron, and T. Virtanen, “Unsupervised Adversarial Domain Adaptation Based on The Wasserstein Distance For Acoustic Scene Classification”, in *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2019, pp. 259–263.

[91] K. A. S. Immink and J. H. Weber, “Minimum Pearson Distance Detection for Multilevel Channels With Gain and/or Offset Mismatch”, *IEEE Transactions on Information Theory*, vol. 60, no. 10, pp. 5966–5974, 2014.

[92] H.-S. Park and C.-H. Jun, “A Simple and Fast Algorithm for K-Medoids Clustering”, *Expert Systems with Applications*, vol. 36, no. 2, pp. 3336–3341, Mar. 2009.

[93] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine Learning in Python”, *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[94] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “PyTorch: An Imperative Style, High-Performance Deep Learning Library”, in *Advances in Neural Information Processing Systems*, vol. 32, Curran Associates, Inc., 2019, pp. 8026–8037.

[95] O. Martin, I. Kotsia, B. Macq, and I. Pitas, “The eNTERFACE’ 05 Audio-Visual Emotion Database”, in *22nd International Conference on Data Engineering Workshops (ICDEW’06)*, 2006, pp. 1–8.

[96] S. R. Livingstone and F. A. Russo, “The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English”, *PLoS ONE*, vol. 13, no. 5, pp. 1–35, May 2018.

[97] D. Xu, U. Yapanel, S. Gray, J. Gilkerson, J. Richards, and J. Hansen, “Signal processing for young child speech language development”, in *Proceedings of the 1st Workshop on Child, Computer, and Interaction (WOCCI-2008)*, 2008, pp. 20–25.

[98] A. Cristia, M. Lavechin, C. Scaff, M. Soderstrom, C. Rowland, O. Räsänen, J. Bunce, and E. Bergelson, “A thorough evaluation of the Language Environment Analysis (LENA) system”, *Behavior Research Methods*, Jul. 2020.

[99] K. Siirilä, “Language Environment Analysis (LENA) -menetelmän validiteetti keskosvauvojen ääniympäristön arvioinnissa”, Master’s thesis, University of Turku, 2019.

[100] M. L. McHugh, “Interrater reliability: the kappa statistic”, *Biochemia medica*, vol. 22, no. 3, pp. 276–282, 2012.

[101] D. G. Altman and J. M. Bland, “Standard deviations and standard errors”, *BMJ*, vol. 331, no. 7521, p. 903, 2005.

[102] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization”, in *3rd International Conference on Learning Representations (ICLR)*, San Diego, California, USA, May 2015.

[103] T. Tieleman and G. Hinton, *Lecture 6.5—RMSProp: Divide the gradient by a running average of its recent magnitude*, COURSERA: Neural Networks for Machine Learning, 2012.

[104] S. Zhao, T. Heittola, and T. Virtanen, “Active Learning for Sound Event Detection”, *IEEE Transactions on Audio Speech and Language Processing*, vol. 28, pp. 2895–2905, Nov. 2020.

[105] S. Gharib, K. Drossos, E. Cakir, D. Serdyuk, and T. Virtanen, “Unsupervised Adversarial Domain Adaptation for Acoustic Scene Classification”, in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, Tampere University of Technology, 2018.

[106] S. Sun, C.-F. Yeh, M.-Y. Hwang, M. Ostendorf, and L. Xie, “Domain Adversarial Training for Accented Speech Recognition”, in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2018, pp. 4854–4858.

[107] X. Glorot, A. Bordes, and Y. Bengio, “Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach”, in *Proceedings of the 28th International Conference on Machine Learning (ICML)*, Bellevue, Washington, USA: Omnipress, 2011, pp. 513–520.

[108] F. Afroz and S. G. Koolagudi, “Recognition and Classification of Pauses in Stuttered Speech Using Acoustic Features”, in *2019 6th International Conference on Signal Processing and Integrated Networks (SPIN)*, 2019, pp. 921–926.

APPENDIX A: THE PROCEDURE FOR SPLITTING AUDIO SAMPLES OF FESC INTO UTTERANCES

First, the audio signal was highpass-filtered with 50 Hz as the passband cut-off frequency to remove noise. Second, after filtering, the short-time energy (STE) of the signal was calculated using a 250-ms sliding Hamming window. The STE of a time domain signal is defined as

$$STE = \sum_{n=0}^{N-1} \left[x(n)w(m-n)\right]^2, \tag{A.1}$$

where $x(n)$ is the time domain signal, $w(n)$ is the windowing function, m is the time index of the block of samples in the windowing function, and N is the total number of samples in the time domain signal [2]. Third, after calculating the STE of the signal, median filtering was performed with a window length of 1.0 seconds to smooth out the STE. Fourth, a threshold of $\frac{1}{40}$ of the median value of the median filtered STE was chosen as an indication of speech. The time instants with an STE greater or equal to this threshold were considered as time instants with speech. Because speech does not typically contain pauses that are longer than 250 ms [108], a silence threshold of 250 ms was chosen. If speech was not present for more than 250 ms, it was considered as a pause in speech. Finally, the audio clips were split into separate utterances at the midpoints of the pauses in speech. To avoid having audio clips that don't contain speech, e.g. a person inhaling or exhaling, any audio clips that were shorter than 1.0 seconds were discarded.