Universität Zürich[UZH]

Masterarbeit
zur Erlangung des akademischen Grades
Master of Arts
der Philosophischen Fakultät der Universität Zürich

Acoustic modelling for Swiss German ASR

Verfasserin/Verfasser: Iuliia Nigmatulina
Matrikel-Nr: 16-736-597

Referentin/Referent: PD Dr. Tanja Samardzic

Institut für Computerlinguistik

Abgabedatum: 01.06.2020

Abstract

This thesis is about automatic speech recognition (ASR) for Swiss German dialects. Swiss German is a spoken variety of Standard German and does not have orthographic writing. At the same time, it is represented by a number of regional variations, some of which are strongly different. These two features are the main challenges for multi-dialectal automatic speech recognition (ASR). Strong regional variability causes many difficulties for modelling on all the linguistic levels: from the phonological level (modelling minimal speech units) to the syntactic level (modelling sequences of words). It becomes even more complicated without a standardised spelling since the transcription of speech can be hardly consistent. Nevertheless, the prestige of Swiss German in the country and its dominant usage in spoken communication (including formal situations) make the development of a high quality ASR system for Swiss German dialects very important.

I propose a complete recipe for building Swiss German multi-dialectal ASR systems using the Kaldi toolkit. For the first time, the system is trained with a single acoustic model on the data from many dialects spoken in different German-speaking parts of the country (14 regional varieties). The data is taken from the ArchiMob corpus of spoken Swiss German where it is manually transcribed with an approximate phonemic transcription. This 'dialectal' writing provides close correspondence between grapheme labels and the acoustic signal but is extremely noisy due to the lack of consistency: for example, the phrase 'viele Leute' ('many people') can be transcribed as *viil lüüt, viil liit, vill lüüt, vill liit*. To better assess word-level performance for dialectal transcriptions, a flexible WER measure was used.

The advanced speech recognition techniques and architectures have been tested with the best results achieved by the time-delay neural network architecture with data augmentation and the iVector features: 42.38% of WER and 21.53% of flexible WER. This result is a 12.01% absolute and a 28.34% relative WER improvement over the baseline. The work, therefore, offers the new strong baseline for Swiss German ASR and provides the discussion for the possible future improvements.

Zusammenfassung

Diese Arbeit befasst sich mit der automatischen Spracherkennung (ASR aus dem Englischen für Automatic Speech Recognition) für Schweizer Dialekte. Schweizerdeutsch ist eine Variation der gesprochenen deutschen Sprache und besitzt keine spezifische Orthografie. Die Sprache ist ausserdem durch verschiedene zum Teil stark variierende Unterschiede zwischen geografischen Regionen gekennzeichnet. Diese zwei Eigenschaften repräsentieren die Hauptherausforderungen für multidialekt ASR. Dialektvariabilität verursacht diverse Schwierigkeiten für die Sprachmodellierung auf allen linguistischen Ebenen: von der phonologischen (Modellierung von minimalen Spracheinheiten) bis zur syntaktischen Ebene (Modellierung von Wortsequenzen). Eine fehlende standardisierte Rechtschreibung erschwert die Aufgabe zusätzlich da die Transkription von Texten kaum konsistent ist. Der dominante Gebrauch der Dialektsprache in der gesprochenen Kommunikation (inklusive formale Situationen) unterzeichnet die Wichtigkeit der Entwicklung eines hochqualitativen ASR-Systems für die schweizerdeutsche Dialektsprache.

Ich präsentiere ein Rezept für die Entwicklung von multidialekt ASR-Systemen für Schweizerdeutsch basierend auf dem Kaldi toolkit. Das System wird zum erstem Mal mit einem einzigen akustischen Modell basierend auf Daten von vierzehn verschiedenen Dialekten trainiert. Die Daten stammen vom ArchiMob Korpus, welcher manuell phonetisch transkribiert wurde. Diese Dialektschrift zeigt nahe Übereinstimmung mit Grapheme Labels und dem akustischen Signal aber ist extrem verzerrt wegen der fehlenden Konsistenz: Zum Beispiel, der Ausdruck 'Viele Leute' kann als *viil lüüt, viil liit, vill lüüt* or *vill liit* transkribiert werden. Ein flexibler WER Massstab wurde benutzt, um die Wort-Level Performance für Dialekttranskriptionen besser zu evaluieren.

Tests mit fortgeschrittenen ASR-Techniken und Architekturen zeigen die besten Resultate mit der Time-Delay Neural Network Architektur mit Data Augmentation und dem iVector Feature: 42.38% WER und 21.53% flexible WER. Dieses Resultat widerspiegelt eine absolute WER Verbesserung von 12.01% und eine relative WER Verbesserung von 28.34% gegenüber der Baseline. Diese Arbeit dient somit als neue Baseline für ASR in Schweizerdeutsch und öffnet die Diskussion für weitere zukünftige Verbesserungen.

Acknowledgement

First of all, I would like to thank a lot my supervisor Tanja Samardžić who has supported me from the beginning and helped to realise my old dream to work with automatic speech recognition. I appreciate all the discussions we had together, her motivating guidance and the detailed feedback that Tanja gave me during the work on the thesis. With Tanja's supervision I always felt supported and confident which allowed me being better focused on the subject.

I would like to thank Francisco Campillo for his very useful recommendations, motivation, and help at the beginning.

Many thanks to Martin Volk and Volker Dellwo for their invaluable support, involvement, and encouragement. I very much appreciate an incredibly inspiring and motivating atmosphere in the Institute of Computational Linguistics, and I am very grateful to all the lecturers, colleagues and other students with whom I worked during my studies. Especially, I would like to thank Thayabaran Kathiresan for all the discussions we had about speech recognition and for the enormous general support from his side.

I would also like to thank Tannon Kew who worked on his master thesis on the related topic almost at the same time as I did. Tannon helped me a lot with many questions, from general topics to small but not less important details. It is great when there is someone to share with and who can very well understand your very special concerns.

I thank all my friends and flatmates for staying with me all these months. Special thanks to Denis S. who supported me a lot during my studies, and to Lucas T. for translating my abstract.

Finally, I would like to say thanks to my family that was always close in spite of the distance; to my parents who do not trust any speech technologies and to my brother who is my greatest inspiration ever.

Contents

Abstract i

Acknowledgement iii

Contents iv

List of Figures vii

List of Tables viii

List of Acronyms ix

1 Introduction 1

1.1 Objectives . . . 2
1.2 Motivation . . . 2
1.3 Structure . . . 3

2 Automatic speech recognition 5

2.1 ASR pipeline . . . 6
2.2 Evaluation . . . 9
2.3 Acoustic features . . . 10
2.4 ASR toolkits: why Kaldi . . . 14
2.4.1 Kaldi . . . 17
2.5 Summary . . . 18

3 Acoustic modelling 19

3.1 Hidden Markov Models . . . 19
3.2 Gaussian Mixture Models . . . 22
3.3 Deep Neural Networks acoustic models . . . 25
3.4 Decoding and training . . . 26
3.4.1 Decoding . . . 26
3.4.2 Training . . . 28
3.5 Adaptation techniques . . . 31
3.5.1 GMM specific adaptation . . . 31

3.5.2 DNN specific adaptation . . . 32
3.5.3 Adaptation with iVector . . . 32
3.5.4 Subspace Gaussian Mixture Models . . . 33
3.6 Summary . . . 34

4 ASR of Swiss German 35
4.1 Challenges of Swiss German ASR . . . 35
4.1.1 Inter- and intra-dialect variability . . . 35
4.1.2 Absence of orthography . . . 37
4.1.3 Quality estimation of the non-standardised languages ASR . . . 40
4.2 Swiss German speech resources . . . 40
4.2.1 Swiss German speech data sets . . . 41
4.2.2 ArchiMob . . . 43
4.3 Related work . . . 45
4.3.1 Studies on multi-dialect ASR . . . 46
4.3.2 Studies on ASR for Swiss German . . . 47
4.4 Summary . . . 48

5 Replicating the baseline model 49
5.1 Replication with the initial data set . . . 49
5.1.1 Data preparation . . . 50
5.1.2 Feature extraction . . . 51
5.1.3 Training acoustic models . . . 52
5.1.4 Decoding . . . 53
5.1.5 Evaluation . . . 55
5.1.6 Summary . . . 57
5.2 Replication with increased in-domain data set . . . 58
5.2.1 Adding data from the second ArchiMob release . . . 58
5.2.1.1 Preparing CSV transcription file . . . 59
5.2.1.2 Synchronisation between csv and audio files . . . 60
5.2.2 Training/development/evaluation data split . . . 61
5.3 Evaluating systems trained on the first VS the second releases of the ArchiMob data . . . 62
5.3.1 Flexible WER evaluation . . . 63
5.3.2 Evaluation results for GMM and NN acoustic models . . . 64
5.4 Summary . . . 65

6 Improving the acoustic model 66
6.1 GMM level improvements . . . 67
6.1.1 Discriminative criteria . . . 67

6.1.2 Parameters tuning: number of senones and Gaussians 68
6.1.3 Improving generalisation 70
6.1.4 Results 71
6.2 Improvements for NN 73
6.2.1 Modelling long-term temporal dependencies 74
6.2.2 iVector adaptation 76
6.2.3 Simulated data augmentation 77
6.2.4 Training TDNN 77
6.2.5 Results 78
6.2.6 Summary 79

7 Performance analysis **80**
7.1 In-domain and out-of-domain data evaluation 80
7.2 Analysis of recognition results 82
7.2.1 Phoneme recognition 82
7.2.2 Per dialect evaluation 84
7.3 Discussion 85

8 Conclusion **87**

References **89**

List of Figures

1 ASR pipeline (from https://medium.com/@jonathan_hui/speech-recognition-gmm-hmm-8bb5eff8b196 (20.02.20)). . 8
2 An example of HMM for the word /s ih k s/ ('six') (from http://www.inf.ed.ac.uk/teaching/courses/asr/2018-19/asr04-cdhmm-handout.pdf (26.05.20)). 20
3 Gaussian Mixture Models. 23
4 Weighted finite-state transducer example illustrating a relationship between two levels of representation: pronunciation lexicon and language model $L \circ G$ (The input label i, the output label o, and weight w of a transition are marked on the corresponding directed arc by $i : o/w$). 27
5 Distribution of dialects for different linguistic levels: (from left to right) lexicon, phonology, morphology, syntax [Scherrer and Stoeckle, 2016]. 36
6 Transducer components of the decoding graph in Kaldi (from [Ravanelli, 2017]). 54
7 Examples from the EXB (left side) and XML (right side) formats. . . 58
8 Distribution of insertions, deletions and substitutions for WER and FlexWER scoring. 64
9 Models with different numbers of senones and Gaussians. 71
10 TDNN architecture with sub-sampling (red) and without sub-sampling (blue+red)[Peddinti et al., 2015]. 75
11 Distribution of errors for vowels (two upper plots) and consonants (two lower plots) for the NNET-DISC-4k40k model evaluated on the in-domain data. 83
12 WER for different dialects with in-domain evaluation. 84
13 Zürich German vowel system [Fleischer and Schmid, 2006]. 103
14 Distribution of errors for vowels (two upper plots) and consonants (two lower plots) for the TDNN-iVector model evaluated on the in-domain data. 103

List of Tables

1 The results of comparison of the open-source ASR frameworks in accuracy and speed [Belenko and Balakshin, 2017] 16
2 Swiss German spoken corpora (SSG stands for Swiss Standard German) 42
3 Number of interviews per canton 43
4 Normalised and Dialectal (Dieth) transcriptions 44
5 Transducers used in the ASR decoding 53
6 Framework of the ASR system for ArchiMob prepared by Spith AG; from [Campillo, 2018] 55
7 Scores for the system trained and evaluated on different dialects . . . 56
8 Number of utterances in train/dev/test sets 62
9 Evaluation of the baseline setup trained on the first release and on the second release of the ArchiMob corpus ('ins' — insertions, 'del' — deletions, 'sub' — substitutions) 63
10 Evaluation of the baseline acoustic models trained on the ArchiMob_r2 65
11 Evaluation of systems with different GMM models (compared to the DNN-HMM systems; development set) 73
12 Evaluation of the system with the TDNN acoustic model in comparison with the baseline model (development set) 78
13 Statistics for the in-domian and out-of-domain test sets 81
14 Evaluation of the models on the in-domian and out-of-domain test sets 81
15 Files created by the end of data preparation and feature extraction steps. 102

List of Acronyms

NLP	Natural Language Processing
ASR	Automatic Speech Recognition
SR	Speech Recognition
STT	Speech to Text
AM	Acoustic Model
LM	Language Model
SLM	Statistical Language Model
NLM	Neural Networks Language Model
GMM	Gaussian Mixture Model
DNN	Deep Neural Network
HMM	Hidden Markov Models
WFST	Weighted Finite State Transducers
CD	Context-Dependent
CI	Context-Independent
WER	Word Error Rate
CER	Character Error Rate
MFCC	Mel-Frequency Cepstral Coefficients
FBANK	Mel-frequency filterbank log energies
PLP	Perceptual Linear Prediction
DFT	Discrete Fourier Transform
iDFT	Inverse Discrete Fourier Transform
LDA	Linear Discriminant Analysis
CMN	Cepstral Mean Normalization
CMVN	Cepstral Mean and Variance Normalization
VTLN	Vocal Tract Length Normalization
ML	Maximum Likelihood
EM	Expectation-Maximization
MLLT	Maximum Likelihood Linear Transformation

MLLR	Maximum Likelihood Linear Regression
fMLLR	feature-domain Maximum Likelihood Linear Regression
MAP-LR	Maximum a Posteriori Linear Regression
CE	Cross-Entropy
CD-DNN-HMM	Context-Dependent Deep Neural Network — Hidden Markov Model
MMI	Maximum Mutual Information
BMMI	Boosted Maximum Mutual Information
MPE	Minimum Phone Error
MBR	Minimum Bayes Risk
sMBR	state Minimum Bayes Risk
UBM	Universal Background Model
CTC	Connectionist Temporal Classification
NN	Neural Networks
RNN	Recurrent Neural Networks
LSTM	Long Short-Term memory Networks
TDNN	Time-Delay Neural Networks
CNN	Convolutional Neural Networks
DBN	Deep Believe Networks
LSTMP	Long Short-Term Memory Projection
SGD	Stochastic Descent Method
OOV	Out-Of-Vocabulary words

1 Introduction

In the few last decades, automatic speech recognition (ASR) has made a considerable breakout thanks to enhanced computational resources, new neural networks algorithms (NN), and creation of massive data sets. Since natural language communication with automatic systems (i.e. chatbots) becomes more in demand nowadays, it is going beyond the limited number of languages spoken world widely, such as English or Spanish, bringing new challenges to the development of ASR[1].

A special case of ASR is a multi-dialect speech recognition, which instead of dealing with homogeneous language data includes highly variable data from different language varieties. Multi-dialect ASR is getting more popular because many languages are presented with a number of dialects and training separate recognition systems for each variety often is not possible. It is especially important for those communities where the status of dialects is high in their everyday communication (as for Arabic or Swiss German dialects) and/or when the amount of training data for each variety in separation does not allows building high quality ASR. This type of speech recognition leaves much space for improvement, as its recognition accuracy is still considerably lower comparing to the standard speech recognition [Elfeky et al., 2018].

This thesis is a study on ASR for Swiss German including dialectal varieties spoken in 14 different cantons. So far, there has been no ASR system with an acoustic model trained on that many Swiss German dialects. Therefore, training such a multi-dialect speech recognition system, as well as investigating the better options for its improvement, became the initial motivation for this work.

[1]For the terminological simplicity, under ASR here and further I mean large vocabulary continuous speech recognition (LVCSR).

1.1 Objectives

The **goal** of the thesis is to train and evaluate different configurations of ASR system for Swiss German on the ArchiMob corpus[2] data with the focus on the acoustic model. The research questions investigated here are:

RQ1: Which techniques and architectures are useful in order to improve the acoustic model in the multi-dialect scenario and with limited data resources?

RQ2: What problems remain after adapting the standard approaches?

1.2 Motivation

Some dialects, taking into account the importance and the scale of their usage, should be in a focus of natural language processing (NLP) research in the same degree as other official languages. From a linguistic perspective, it is not always possible to set the threshold, above which a dialect should become a language. In the functional use, from the sociolinguistics point of view, a dialect is defined as a regional variant of the standardised language [Haugen, 1966]. Thus, a dialect can receive the status of a language after its standardisation on the political ground, e.g. when a unique national language needs to be established (like Finnish, Norwegian). The perception of dialects can be very different and is often also politically-determined. Some dialects, however, have a higher social status comparing to other dialects or even to a standardised language, because of their general prestige (e.g. Swiss German dialects [Clematide et al., 2016]), or because of being a regional variety on the state level, like Arabic or Spanish world varieties [Elfeky et al., 2018], etc.

The sociolinguistic situation in Switzerland can be described as *diglossic*. According to Ferguson [1959], *diglossia* defines a special type of bilinguism in a speech community when two speech varieties coexist and are used depending on the communicative context: for example, formal or written communication vs informal and everyday communication. In Switzerland, these two varieties are Standard German and Swiss German, where Swiss German is an umbrella term referred to Alemanic dialects spoken on the territory of Switzerland. Swiss German can be also understood as a Swiss accented standard German. To avoid ambiguity, further in the text, I will refer to Swiss accented standard German as Swiss Standard German (SSG).

[2] https://www.spur.uzh.ch/en/departments/research/textgroup/ArchiMob.html.

Developing a high quality ASR system for Swiss German would correspond the needs of Swiss population, about 64% of which are Swiss German-speakers, and is, therefore, an important research and engineer task. The linguistic situation in German-speaking part of Switzerland is not a typical case of diglossia. The perception of two languages does not satisfy the Fergusonian criterion of prestige when a high variety (here Standard German) is supposed to have a higher social status [Hogg et al., 1984; Stępkowska et al., 2012]. Despite playing the role of 'informal' language, the social status of Swiss German dialects can be regarded as a higher one comparing to typical dialects of other languages, as in Switzerland, in everyday communication, dialects dominate over standard German[3]. Moreover, standard German is almost inferior in its prestige to Swiss German [Hogg et al., 1984].

Another motivation to build and improve a multi-dialect speech-to-text system for Swiss German is a research one, in order to define and understand better those challenges of dialectal ASR, which still stay unsolved. First, Swiss German dialects differ considerably between the regions on the phonological, morphological, syntactic, and lexical levels. Second, there is no standardised written tradition, grammar or vocabulary for Swiss German and it is mostly a spoken variety. Finally, there is only little annotated data available for Swiss German dialects. In the previous studies on multi-dialect speech recognition, various approaches have been tested. For example, a) training a single model with the data from different dialects together, b) using a language specific decoding with a generally trained model, c) training separate models for each of the dialects [Elfeky et al., 2018]. The results are usually language-dependent, as well as sensitive to the amount of data available per dialect.

1.3 Structure

Chapter 2 and Chapter 3 provide the technical background for the ASR: with Chapter 2 being focused on the general ASR pipeline and speech recognition evaluation and Chapter 3 going more into detail of acoustic modelling component. The rest of the chapters describe the approaches to the research questions and constitute the main contribution of the thesis. Chapter 4 is an overview of ASR for Swiss German including main challenges for multi-dialect speech recognition and available spoken corpora resources for Swiss German. In Chapter 5, the baseline is introduced with

[3]"Dialect use is common even in formal situations. For instance, members of the national government are free to use their local dialect in political discussions on national TV" [Siebenhaar, 2006]: 482.

a detailed technical description of its pipeline; I present the results of the baseline replication, as well as the new results of the baseline framework trained on more in-domain data; the training/development/test split is also introduced. In Chapter 6, I describe the acoustic modelling experiments aiming to improve the baseline system and discuss the results achieved on the development set. Chapter 7 presents the results of in-domain and out-of-domain evaluation, as well as the analysis of the recognition output (including the recognition of phonemes and per dialect recognition). Chapter 8 is a general conclusion to the whole work. All the code can be found on GitHub[4].

[4] `https://github.com/yunigma/Kaldi-for-ASR-of-Swiss-German`.

2 Automatic speech recognition

Automatic speech recognition (ASR) aims to find the most likely sequence of words given the acoustic input of continuous speech[5]. ASR system nowadays can be either a pipeline system consisting of separate components or an end-to-end one when a system takes features as input and outputs the end solution directly without intermediate steps. End-to-end systems started appearing just recently and among their main advantages are 1) more simple training of the whole system in a single step, 2) avoiding error propagation from acoustic modelling to the final decoding, and 3) avoiding conditional independent assumptions of HMM training. Such systems do not need any additional modules (like text normalisation, or clustering sounds in a bigger groups), training separate models and/or using extra information (like a lexicon or time alignments[6]) [Chiu et al., 2018].

The end-to-end systems, however, cannot yet fully outperform conventional pipeline hybrid models, which consist of separately trained models, where the acoustic modelling is usually a hybrid HMM-Deep Neural Network (DNN) model (see Chapter 3). One of the strong sides of the conventional pipeline models is a more transparent optimisation of an objective function ([Kingsbury, 2009]; see more in 3.4.2). Moreover, training separate models affords to use different training data for models. For example, language model chooses the best sequence of words given several assumptions; it is trained on the collection of texts in the target language and, therefore, does not demand any audio data (see 2.1). For training a language model, large non-transcribed data can be used to improve the output of a system in general that is cheap and easy to augment. Most of commercial systems are still HMM-based, especially when no big training data is available: "Even today, the best speech recognition performance still comes from HMM-based model (in combination with deep learning techniques). Most industrially deployed systems are based on HMM" [Chiu

[5]ASR includes also recognition of isolated words but in this work we will speak about the recognition of continuous speech only.

[6]One of the significant problems for both Hidden Markov Model (HMM)-based and end-to-end models is a data alignment problem, i.e. defining how a label from the label sequence should be aligned to the speech data. The end-to-end models deal with this task using soft alignment when each audio frame corresponds to all possible states but with a certain probability distribution, therefore, it does not need a forced correspondence [Wang et al., 2019].

et al., 2018; Wang et al., 2019, p. 2].

Because of these reasons and due to the limited corpus resources for Swiss German, in the present project, I work with the conventional ASR system with separated models using HMM-DNN acoustic models and an alignment built with the HMM-GMM pipeline. In this chapter, I discuss the technical background for ASR pipeline systems including its major components, evaluation, and feature extraction. At the end of the chapter, I give an overview of the leading ASR toolkits.

2.1 ASR pipeline

ASR task is to map an acoustic sequence of length T to a text of N words length. The ASR pipeline consists of data preparation stage, training three probabilistic models, and decoding stage where three models are combined (see Fig. 1). Three models represent different aspects of language: acoustic model, pronunciation model and language model. Acoustic model serves for mapping the acoustic signal (i.e. acoustic feature vectors) to the phonemes, distinct units of sound in a language, sound 'labels'. Pronunciation model, or dictionary, finds the best correspondence between a sequence of phonemes and graphemes creating *pronunciation lexicon*; it is usually built by professional linguists, or trained separately if some training data is available. Finally, language model generates the most probable sequences of words in a language using the output from the acoustic and the pronunciation models.

Formally speaking, ASR aims to establish such a model that would find the best probability candidate output sequence of words from a set of all possible word combinations (or sentences) in a language given a noisy acoustic observation-sequence. It means calculating the conditional probability $p(W|X)$, where W is a sequence of words ($W = \{w_1, ..., w_n\}$), X is a sequence of input feature vectors representing the observations ($X = \{x_1, ..., x_t\}$) and $\mathcal{V}$ is a vocabulary (adapted from [Jurafsky and Martin, 2009, p. 291] and [Wang et al., 2019]):

$$\hat{W} = \underset{W \in \mathcal{V}}{\operatorname{argmax}}\, p(W|X) \tag{2.1}$$

Modelling $p(W|X)$ directly is complicated and will be biased against the training data. Bayes's theorem is then applied to reformulate the Eq. 2.1:

$$\hat{W} = \underset{W \in \mathcal{V}}{\operatorname{argmax}}\, \frac{p(X|W)p(W)}{p(X)} \tag{2.2}$$

The observation sequence $p(X)$ for a particular phrase is a constant and can be ignored to simplify Eq. 2.2:

$$\hat{W} = \underset{W \in \mathcal{V}}{\operatorname{argmax}}\, p(X|W)p(W) \tag{2.3}$$

In Eq. 2.3, $p(W)$ is a language model and $p(X|W)$ can be regarded as a combination of an acoustic $p(X|Q)$ and a pronunciation $p(Q|W)$ models: $p(X|W) = \sum p(X|Q)p(Q|W)$, where Q is a sequence of phonemes (Fig. 1). Then, we can rewrite the Eq. 2.1 as:

$$\hat{W} = \underset{W \in \mathcal{V}}{\operatorname{argmax}} \sum_{Q} p(W, Q, X) \tag{2.4}$$

$$= \underset{W \in \mathcal{V}}{\operatorname{argmax}} \sum_{Q} p(X|Q)p(Q|W)p(W) \tag{2.5}$$

Some authors call the probability of acoustic features given a word $p(X|W)$ an acoustic model considering pronunciation model $p(Q|W)$ as a part of it [Jurafsky and Martin, 2009]; I will follow this approach.

The first step of the pipeline is converting values of the acoustic signal into a sequence of numbers, or feature vectors. The speech signal is usually highly variable in time. To represent better its variability, it is typically turned into as a sequence of discrete observations where each observation is an audio frame of 25 milliseconds and the step between frames is 10 milliseconds (Fig. 1: Input). This is a sliding window whose length contains enough information to define a phone and yet remaining stationary enough to avoid further ambiguity while mapping feature vectors to phonemes. Each frame is then encoded as a feature vector, and an acoustic input is represented as a sequence of feature vectors.

The next step is acoustic modelling and, to find the correspondence between the acoustic information and a set of (sub)-phones, we classify frames into the classes of (sub)-phones. A GMM and DNN-based classifier is usually used for this purpose to compute the likelihood of an input feature vector given each HMM state q_t representing a phoneme: $p(x_t|q_t)$.

Hidden Markov Model (HMM) is a statistical stochastic model that predicts a probability distribution over a set of 'hidden', i.e. unobserved, states based on the observation of 'visible' objects (Fig. 1: Acoustic model and Lexicon; see more in 3.1). For an HMM with a set of phonemes $\{1, \ldots, J\}$, or 'hidden' states, HMM-based model uses Bayes's theory to introduce the HMM phoneme sequence, as

$Q = \{q_t \in \{1, \ldots, J\} | t = 1, \ldots, T\}$. Then, for a feature vector at a time point t, a phoneme with the highest probability is chosen [Wang et al., 2019].

At the decoding step, we combine this sequence of acoustic likelihoods that represents the acoustic model with HMM structures of word pronunciations, and language model, to get the most likely sequence of words as an output (Fig. 1).

Based on the input data from the acoustic model, the language model selects the best sequence of words for a given language. The probability distribution over sequences of words allows us to predict the next word in a sequence. There are many options for training a language model, but most language models can be referred to as either a statistical or a neural networks one. While in statistical modelling (SLM), the probability distribution is calculated on the n-gram level, neural networks language models (NLM) learn the distributed representation for words, which allows modelling more complex dependencies, better generalisation and better scaling with the large vocabulary size [Bengio et al., 2003]. Although now NLMs become more and more popular, SLMs are still widely used due to their lower computational complexity and better performance on small training data if no pre-trained models are used (e.g. in hybrid systems [Xiong et al., 2018]).

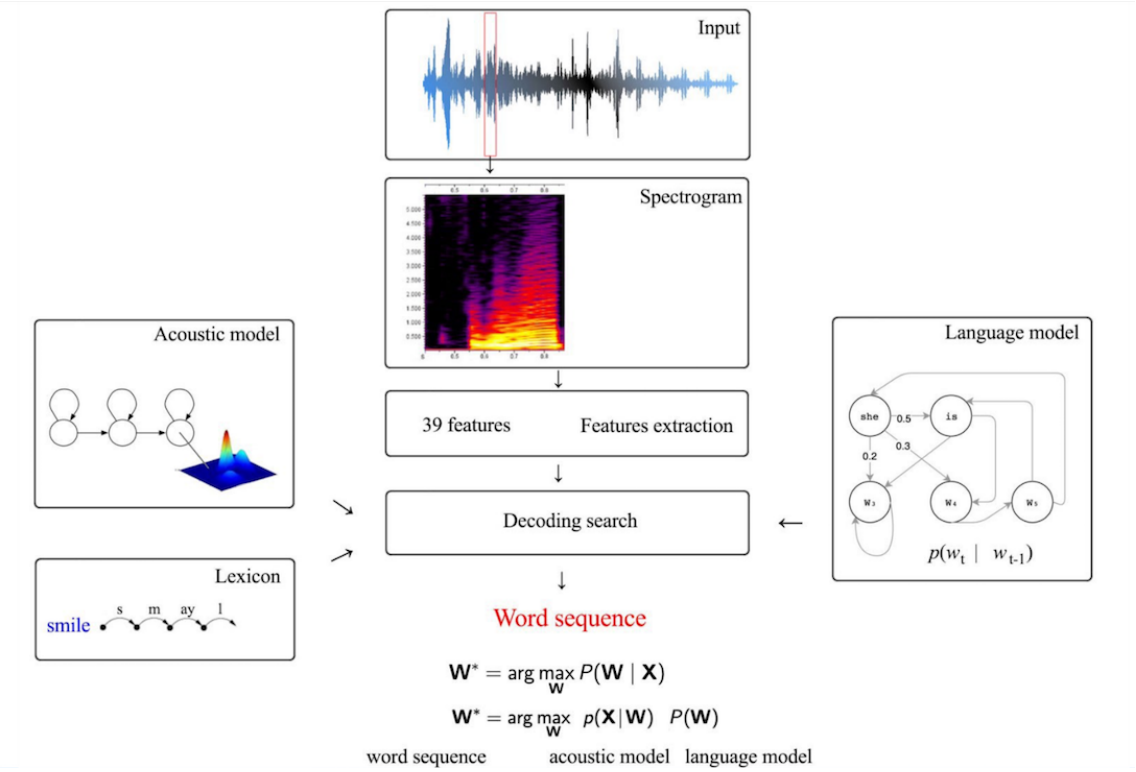

Figure 1: ASR pipeline (from `https://medium.com/@jonathan_hui/speech-recognition-gmm-hmm-8bb5eff8b196` (20.02.20)).

Depending on the data and/or research goals, one can balance the contributions from

the acoustic and language models tuning the lambda parameter (λ). Additionally, to control the length and the number of words in the output, one can use word insertion penalty ($\omega_{penalty}$): if penalty is large (smaller LM probability), decoder will prefer fewer longer words, on the other hand, if penalty is small (larger LM probability), decoder will prefer more shorter words[7]. This kind of weighting helps to avoid splitting long words into multiple shorter ones (Eq. 2.6; [Medennikov, 2015]).

$$\hat{w} = \underset{w}{\operatorname{argmax}}[\ln(p(x|w)) + \lambda * \ln(p(w)) - \omega_{penalty} * n(w)] \tag{2.6}$$

where $n(w)$ is a number of words in a sequence w.

The result of the decoding stage can be a list of N-best hypotheses, or a word lattice. A word lattice is a directed acyclic graph with one starting point, each edge of which contains a word, as well as its acoustic, linguistic and final probability. In the case of word lattice, the best hypothesis is a path in the graph with the highest total probability (see more about decoding during training in 3.4.1).

2.2 Evaluation

The most widely used method of evaluation in continuous speech recognition is word error rate (WER). WER calculates the minimum number of mistakes produced by a system at the word level, relative to the number of words in the reference transcription:

$$WER = \frac{S + D + I}{N} \cdot 100\% = \frac{S + D + I}{S + D + C} \cdot 100\% \tag{2.7}$$

where S, D and I is the minimum number of substitution, deletion and insertion operations required to transform the system prediction to the reference text, where N is the number of words in the reference text and C — the number of words correctly recognised by a system. It is important to mention that WER value can be more than 100%, which depends on the number of insertions. WER, thus, is proportional to the correction cost.

A variation of WER is character error rate (CER). Its formula is the same as for the WER (Eq. 2.7) but CER measures error rate at the character level. CER is more flexible to catch the difference in recognition within a word. A minor error in the

[7] `https://web.stanford.edu/class/cs224s/lectures/224s.17.lec4.pdf` (26.01.20).

ending would be less punished than a completely different word; both cases would be an equal error for the WER measure. At the same time, CER is less successful in evaluation of syntax and semantics errors that are more noticeable on the word level.

There is one more score derived from WER and defining word accuracy:

$$WAcc = 100\% - WER = \frac{C - I}{N} \cdot 100\% \tag{2.8}$$

Speaking about the quality of speech recognition, I will mean the value of WER. When comparing the evaluated system with the baseline system or between each other, the absolute and relative improvement in the quality of recognition means the absolute and relative decrease in WER, determined by two formulas correspondingly:

$$\Delta WER = WER_1 - WER_2 \tag{2.9}$$

$$WERR = \frac{WER_1 - WER_2}{WER_1} \cdot 100\% \tag{2.10}$$

where WER_1 and WER_2 are WER scores of two compared systems.

2.3 Acoustic features

The speech signal is a complex nearly periodic waveform consisting of many sine waves [Ladefoged and Johnson, 2014]. The sound produced by the vocal cords passes through the vocal tract of a speaker, the shape of which in combination with the articulation defines the quality of speech sounds. The aim of acoustic feature extraction is to catch the most relevant information from a continuous speech that would be used to differentiate between different sounds.

For each time frame taken from the speech signal with a certain step (2.1), we extract a feature vector, a numerical representation of the acoustic properties. These features are used as input for the entire ASR system and are responsible for accurate transmission of acoustic information. The most widely used acoustic features are Mel-Frequency Cepstral Coefficients (MFCC) [Davis and Mermelstein, 1980] that are extracted in the following steps [Medennikov, 2015]):

1. Pre-emphasis. One of the characteristics of the acoustic signal is sound energy that is defined by the sound pressure and differs within the same acoustic

fragment. Before splitting a signal into the frames, the signal is typically pre-emphasised. Such preprocessing is needed because voiced segments (like vowels) have more sound energy at the lower frequencies than at the higher frequencies. This leads to the loss of information from the higher frequencies. Pre-emphasis filters a signal to boost the amount of energy at the higher frequencies, which makes information from the high frequencies easier to extract and be used by the acoustic model, and, thus, improves phone detection accuracy.

2. Windowing. Acoustic signal is analogue and needs to be discretised for digital processing, this is achieved by segmenting the signal into frames. When getting a frame of a signal, window function is usually used to define a shape, mostly edges, of a framed signal. The typical window functions are Rectangular, Hanning, and Hamming windows. The last two ones allow smoothing the signal at the boundaries in order to avoid discontinuity; the discontinuity can be problematic for the Fourier analysis described in the next step. The Hamming window is:

$$\omega_t = 0.54 - 0.46 \cos\left(\frac{2\pi t}{T-1}\right), t = 0, 1, ..., T \tag{2.11}$$

where 0.54 and 0.46 are the most typical values of the Hamming window parameters that make edges of the function describing the signal within a frame smoother.

3. Discrete Fourier Transform (DFT) function defined by Eq. 2.12 converts a sequence of equally-spaced samples of the acoustic signal from a time to a frequency domain:

$$Y_k = \sum_{t=0}^{T-1} \omega_t y_t \exp\left(\frac{-2\pi i}{T} kt\right), k = 0, 1, ..., T/2 \tag{2.12}$$

where T is a number of samples, ω is a window function, y is a signal after pre-emphasis, and k is a frequency index[8].

4. Mel filterbank. The dependence between the frequency and the mel scale (loudness) of a signal is not linear, i.e. human ear perception of loudness is different with different frequencies, especially above 1kHz. In order to simulate the frequency resolution according to the properties of the human perception, we apply triangular band-pass filters that transform the DFT power spectrum

[8] i is a unit imaginary number.

to the Mel-scale power spectrum:

$$mel(f) = 1127 \ln \left(1 + \frac{f}{700}\right) \tag{2.13}$$

where f is a signal frequency.

5. Energy is different in different sounds and, therefore, is also an informative feature. Log energy is calculated as a logarithm of the sum of the power of samples in a frame:

$$E_m = \ln \left(\sum_{k=0}^{T/2} |Y_k|^2 H_m(k) \right), m = 0, 1, ..., M-1 \tag{2.14}$$

where Y_k is a signal after DFT, M is a number of triangle filters, and H_m is a Mel-frequency filterbank.

6. Cepstrum: Inverse Discrete Fourier Transform (iDFT). To discriminate better between different sounds, acoustic features containing the information from the vocal tract (filter), where the 'formation' of sounds happens, is more relevant than the more general information from the source of the signal (vocal cords; e.g. F0). Cepstrum analysis helps to separate the information from the source and the filter: while the lower cepstral values are more important for sound detection, the higher cepstral values are useful in detecting pitch (intonation). iDFT, thus, discards some coefficients that, at the same time, increases the decorrelation of the filter bank coefficients:

$$c_n = \sum_{m=0}^{M-1} E_m \cos \left(\frac{\pi(m + 1/2)n}{M} \right), n = 0, 1, ..., M-1 \tag{2.15}$$

Sometimes the cepstrum analysis step is omitted and, then, FBANK features, which are the output from a Mel-frequency filterbank, are used instead. The MFCC features are derived from the filterbank outputs and are not less informative then the FBANK features. The reason that MFCC are often more preferable is that they are less correlated between each other than the filterbank outputs (because of the last iDFT transform). This property is beneficial for machine learning algorithms, e.g. when fitting a Gaussian probability density function to the data without a full covariance matrix. The MFCC components are independent and they are easier to model using a mixture of diagonal covariance Gaussians. Yet, for DNN modelling, the independence of coefficients is not that important and some studies showed bet-

ter results returned with the FBANK features, i.e. [Mohamed et al., 2011; Yoshioka et al., 2014].

The alternative to the MFCC features are Perceptual Linear Prediction (PLP) features [Hermansky, 1990]. The process of their extraction is similar to the one for the MFCC features in terms of using DFT at the beginning and iDFT at the end. The difference is that instead of the log compression, PLP applies equal loudness pre-emphasis and cube-root compression to model the logarithmic nature of speech loudness perception. It is hard to say which approach for feature extraction is a better one. Generally, PLP are slightly more robust against the noise comparing to MFCC; nevertheless, MFCC are more widespread in ASR and considered a safer choice [Psutka et al., 2001].

There are 13 features derived from the cepstral analysis including one energy feature. We can also use some additional techniques to improve and supplement the MFCC features. For example, the MFCC features are not able to catch the dynamic characteristics of an acoustic signal. To include the dynamic information about how a signal is changing in time, vectors of first (delta; Eq. 2.16) and second (delta-delta, acceleration; Eq. 2.17) derivatives are typically added to the vector of 13 cepstral features. Delta features measure the feature changes from the previous frame to the next frame, and delta-delta features measure the acceleration of these changes:

$$D_n = \frac{\sum_{l=1}^{L} l(c_{n+l} - c_{n-l})}{2\sum_{l=1}^{L} l^2} \tag{2.16}$$

$$A_n = \frac{\sum_{l=1}^{L} l(D_{n+l} - D_{n-l})}{2\sum_{l=1}^{L} l^2} \tag{2.17}$$

where c_n is a cepstrum defined by Eq. 2.15. Therefore, 13 cepstral features together with 13 delta and 13 delta-delta features compose a typical vector of 39 acoustic features used in speech processing.

Other techniques for increasing the effectiveness of features that make them less correlated and more resistant to noise can be of two types:

1. Speaker independent

    a) Linear Discriminant Analysis (LDA) is matrix transformation that tends to maximise the distinction between classes (e.g. classes of phones) [Haeb-Umbach and Ney, 1992].

b) Feature-space Maximum Likelihood Linear Transformation (fMLLT) [Gopinath, 1998] or Semi-Tied Covariance (STC) [Gales, 1999] is a feature orthogonalising transformation that makes the features more accurately modelled by diagonal-covariance Gaussians maximising the average likelihood and, thus, reduce the mismatch between an initial model set and the adaptation data.

2. Speaker specific

a) Cepstral Mean Normalization (CMN) and Cepstral Mean and Variance Normalization (CMVN) serve to decrease the signal variability over all the frames in a single utterance. The normalisation helps to adjust values to countermeasure the variants in each recording [Liu et al., 1993; Acero and Huang, 1995].

b) Vocal Tract Length Normalization (VTLN) is usually applied to smooth the interspeaker variability [Eide and Gish, 1996].

c) Feature-space Maximum Likelihood Linear Regression (fMLLR) transforms acoustic features to speaker adapted features by a multiplication operation with a transformation matrix [Gales and Woodland, 1996; Li et al., 2002]. ("maximum likelihood linear regression (MLLR) aims to linearly transform the means of the gaussian models in order to maximise the likelihood of the adaptation data given the correct hypothesis (supervised MLLR) or the decoded hypothesis (unsupervised MLLR). (Padmanabhan, 2000)")

d) Maximum a Posteriori Linear Regression (MAP-LR) is another type of linear regression used for the adaptation purposes [Gauvain and Lee, 1994].

2.4 ASR toolkits: why Kaldi

Nowadays there are many ASR systems offered by different companies and universities for commercial or research purposes. Among the open-source frameworks providing the ASR functionality, the most famous are Lingvo, Kaldi ASR[9], HTK

[9] `http://kaldi-asr.org/doc/index.html`

Toolkit[10], CMU Sphinx[11], Julius[12], RWTH ASR[13]. The HTK is a Hidden Markov Model Toolkit written in C to build and manipulate Hidden Markov Models. An actively maintained GitHub repo is, however, absent for HTK, its latest version was released in 2015 only and it provides only classical speech recognition algorithms and data structures. RWTH ASR was developed in Rhine-Westphalian Technical University of Aachen and has good recognition accuracy but low computational speed [Belenko and Balakshin, 2017]. Julius was originally created for Japanese speech recognition that makes it less flexible in regard to other languages. CMU Sphinx is written in Java and like Kaldi has a module structure good for research purposes.

Kaldi is a framework written mostly in C++ and one of its advantages is that it is under the Apache License that allows its use in the commercial projects too. As well as CMU Sphinx, Kaldi is not a user application, but rather a toolkit that can be used in order to develop applications for end users with a powerful capability for speech recognition [Matarneh et al., 2017]. Another important advantage of Kaldi over some other speech recognition software is its extensibility and modularity. It is still well updated according to the newest ASR techniques and many contributors keep providing plenty of 3rd-party modules that can be used in various speech processing tasks. At the current moment, Kaldi perhaps provides the most modern and the best updated ASR techniques among the existing open-source frameworks in speech recognition [Gaida et al., 2014]. The evaluation of the most famous open-source ASR frameworks with the models trained on the WSJ1 data set[14] (Wall Street Journal1: 160 hours of training data and 10 hours of test data) showed that the best results, both in accuracy and speed, were achieved with a big gap by Kaldi system (see Table 1).

These characteristics partly predetermined the choice of Kaldi as the ASR tool for the current research. Another more practical reason is that the University of Zürich already has Kaldi installed on its server that allowed me directly working with it from the very beginning.

[10] http://htk.eng.cam.ac.uk/

[11] https://cmusphinx.github.io/wiki/

[12] https://github.com/julius-speech

[13] http://www-i6.informatik.rwth-aachen.de/rwth-asr/

[14] https://catalog.ldc.upenn.edu/LDC94S13A.

Table 1: The results of comparison of the open-source ASR frameworks in accuracy and speed [Belenko and Balakshin, 2017]

SYSTEM	WER, %	WRR, %	SF (Speed Factor)
HTK	19.8	80.2	1.4
CMU Sphinx	21.4/22.7	78.6/77.3	0.5/1
Kaldi	6.5	93.5	0.6
Julius	23.1	76.9	1.3
RWTH ASR	15.5	84.5	3.8

Among the best commercial systems that report state-of-the-art results are Dragon Speech Recognition by Nuance, Google Speech API, Microsoft Speech API (Cortana), Yandex SpeechKit, Amazon Alexa API, Siri by Apple, etc. (see [Matarneh et al., 2017] for an overview). Big companies usually have in their disposition a large amount of training data that, along with the advanced techniques, still stays the most important advantage on the way of achieving the best results. The big annotated data is expensive and demands powerful computational resources, therefore, only big companies can afford this. The code of the commercial ASR tools is inaccessible and evaluation experiments in order to compare different models can be conducted only with the already trained models. The difference in training data makes a fair comparison of the models' architectures impossible. Nevertheless, there are some attempts to show a huge gap between the systems from big companies and the systems trained on much less data: tested on the same properly balanced test data, The Sphinx-4, Google Speech API and Microsoft Speech API got the 37%, 9% and 18% WER respectively [Këpuska and Bohouta, 2017].

Evaluation of several commercial and open-source systems on different English data sets proved that, despite the significantly lower performance comparing to the commercial models, Kaldi WER is one of the best among the open-source frameworks [Germann et al., 2019]. In the present case, we have to deal with the low-resource language, when only limited data is available and the use of larger pre-trained models is not possible either due to the computational complexity. Yet, the goal of the thesis is not to outperform the best commercial results but to investigate the main obstacles of acoustic modelling in the multi-dialect and low-resource situation.

2.4.1 Kaldi

Kaldi ASR provides many recipes (or pipelines) for various speech processing tasks: e.g. *sre*, *voxceleb* — for speaker identification; *wsj*, *swbd* — for speech recognition; *lre* — for language identification, etc. Some recipes intended for the same tasks are usually adapted for different data sets, including data sets on different languages or multilingual sources. The motivation behind having separate recipes is that, in most cases, a particular preprocessing is expected for each data set. All recipes are located in the folder /egs (examples); each of them presents a separate folder with README file, the core pipeline file `run.sh`, auxiliary files `path.sh` and `cmd.sh`, and all other necessary pipeline scripts (mostly written in bash) organising the whole flow and calling Kaldi functional scripts (mostly written in C++, and sometimes in Python). There can be more than one version of the same recipe and then all versions are kept in the subfolders in the current recipe direction: e.g. the *swbd* recipe has three different versions *s5*, *s5b*, *s5c*.

To build a pipeline for a particular task, a user usually needs to choose a corresponding Kaldi recipe and whether to use it directly or to adapt it to the given data and goals. All necessary scripts are located in two folders — /utils and /steps — while /utils has scripts for more general operations (shuffling, validation, conversion, etc.), in /steps, there are task specific scripts, i.e. for extracting features, aligning, training models, decoding, etc.

Kaldi functionality covers a broad range of ASR techniques including Linear Discriminant Analysis with Maximum Likelihood Linear Transformation (LDA+MLLT), speaker adaptive training (SAT), maximum likelihood linear regression (MLLR), feature-space MLLR (fMLLR) and maximum mutual information (MMI, fMMI). Kaldi supports Gaussian Mixture Models (GMM) and Subspace GMM (SGMM), as well as the training of Deep Neural Networks (DNN) on top of GMM models with layer-wise pre-training based on Restricted Boltzmann Machines, per-frame cross entropy training, and sequence-discriminative training, using lattice framework and optimising different training criteria [Gaida et al., 2014] (see more in Chapter 3).

Initially, Kaldi pipelines offered recipes for English only; later, recipes for some other languages have been added but there is no particular recipe that would be used for a German data set yet. There are more than 10 recipes for English, including such corpora as Wall Street Journal (81 hours), the Switchboard corpus (317 hours), HUB4 (1996 & 1997) English Broadcast News (75 + 72 hours) , Fisher (1761 hours), and Librispeech (960 hours). The Wall Street Journal recipe (`egs/wsj/s5`) has been originally chosen to train the first ASR system on the ArchiMob data [Scherrer

et al., 2019b]; see more in Chapter 4), because this recipe is the most basic one for speech recognition (all other recipes link to its steps and utils) and the amount of training data of this recipe is getting close to the data available from ArchiMob (around 60 hours). For my baseline and for the further experiments, I will use the same pipeline originally adapted for the ArchiMob data by Francisco Campillo ([Campillo, 2018]; see more in Chapter 5).

2.5 Summary

To understand the general principles of ASR, in this chapter, I have presented a standard ASR pipeline, its core components, feature extraction and evaluation metrics. The last section is an overview of the main ASR toolkits nowadays; it shows the advantages of the speech recognition framework Kaldi, which was used for training the recognition systems in the current master project (see more in Chapters 5-6). Since in my experiments, I am mostly focused on acoustic modelling, the following chapter provides more insight into this ASR component, including its basic setup and some advanced techniques.

3 Acoustic modelling

Acoustic model aims to classify the acoustic feature vectors to phoneme classes, and to map sequences of phonemes to the sequences of graphemes. A classifier computes the likelihood of an input feature vector given each possible phoneme and, then, the best probability phoneme is chosen. The acoustic model, with its general idea and objective, is briefly discussed in 2.1. In this chapter, I address acoustic modelling in more detail.

Since speech has a sequential nature, it can be represented as a time-varying sequence of 'stationary' spectral vectors. To combine the dynamic and stationary aspects of speech, we can represent an acoustic model as consisting of two parts (section 2.1). The first part is responsible for estimating the likelihood of having an observable stationary signal fragment (usually taken in a frame) given a phoneme and is called the emission probability. The second part is the transition probability, which serves to estimate the probability of a transition from one phoneme to another. The second component is used to capture the variability of a speech signal over time. For this purpose, Hidden Markov Models (HMM) are widely used in speech recognition, which are built with both the transition and the emission probabilities.

I will start the chapter introducing HMM that enable the dynamic aspect of acoustic modelling. Later, I will present two approaches of acoustic signal classification: Gaussian Mixture Models (GMM) and Deep Neural Networks (DNN)-based. Besides these core components of acoustic modelling, the chapter covers training and decoding approaches and introduces enhancement techniques, such as adaptation.

3.1 Hidden Markov Models

Hidden Markov Model (HMM) is a sequence model that predicts unobserved ('hidden') states (phonemes in our case) given 'observed' events (feature vectors) assuming that behaviour of 'observed' events depends on the 'hidden' ones. An HMM makes two other assumptions that allows efficient modelling of a speech signal. First, the probability of a state is conditionally dependent only of the previous state; sec-

ond, the observation generated at a particular state depends only on the state that generated it. At each time step, an HMM makes a transition from its current state to one of its connected states and generates a new observation [Gales and Young, 2007] (Fig. 2).

HMM is defined with:

1. a set of N states (phonemes) $Q = \{q_1, \ldots, q_N\}$. When a number of speech frames is T, a phoneme of a model for the frame t is q_t.

2. a set of observations $O = \{o_1, \ldots, o_T\}$ where an acoustic observation o_t for the frame t is a feature vector extracted from this frame (e.g. 39 MFCC; see 2.3). A set of likelihoods of these observations, or emission probability $b_i(o_t) = p(o_t|Q_i)$ where Q_i is a phoneme at a model state i.

3. a transition probability matrix A that contains probabilities for each phoneme of staying at the same state or going to the next phoneme: $A = \{a_{ij}\}$:

$$a_{ij} = p(q_t = Q_j | q_{t-1} = Q_i), i, j = 1, 2, ..., N \tag{3.1}$$

4. an initial state distribution $p = \{p_1, \ldots, p_N\}$ where $p = P(q_1 = Q_i)$

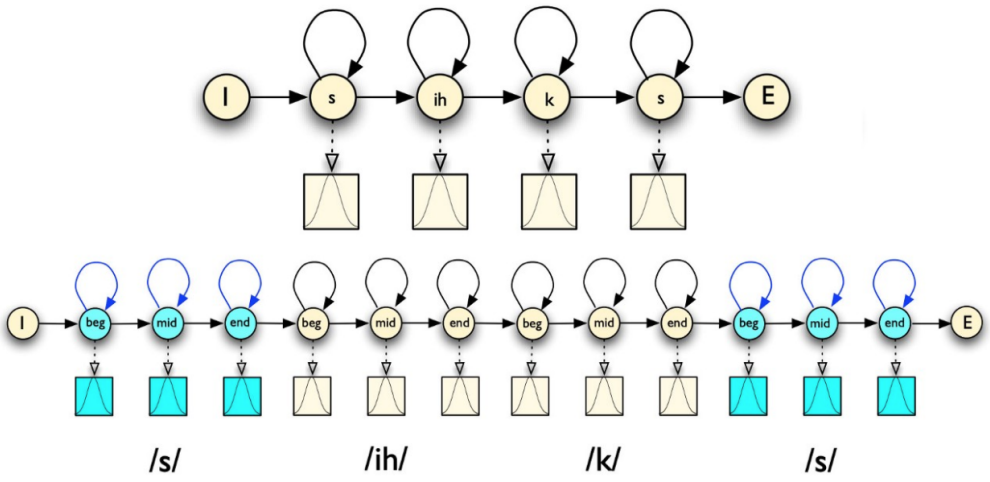

Figure 2: An example of HMM for the word /s ih k s/ ('six') (from `http://www.inf.ed.ac.uk/teaching/courses/asr/2018-19/asr04-cdhmm-handout.pdf` (26.05.20)).

Inter-influence between neighbouring phonemes leads to the coarticulation effect. Because of the coarticulation, a phoneme can be acoustically very different at its beginning (coarticulating with the previous phoneme), end (coarticulating with the following phoneme) and in the middle that is a stationary part. The informa-

tion about coarticulation can help to predict better the target phoneme, as well as the phonemes around it. Hence, instead of isolated phonemes, context-dependent phonemes are typically used.

Context-dependent (CD) phoneme is usually a triphone that includes the left and right context of a target phoneme: equals phoneme_B, phoneme_I and phoneme_E that are called *states* (correspond to 'beg', 'mid', 'end' in Fig. 2). In the case where a word consists of only a single phoneme and occurs outside of any context, e.g. between pauses, a state phoneme_S is used[15]. CD states supply more detail information per frame that makes this approach advantageous over the monophone based models [Dahl et al., 2011]. At the same time, the number of all possible triphones is very high, squared number of all phonemes, which makes the computation much more complex. Since some of the triphones are similar and others can not appear in a language, instead of directly using all the triphones, tied triphone HMM states are used. Tied triphone HMM states are also called *senones* and are, in fact, clusters of original triphone states (for example, based on a decision tree), which share the similar set of parameters. A number of senones, as well as a number of Gaussians, are hyperparameters that can be tuned during the training of an acoustic model.

The HMM structure for a word is a concatenation of phoneme HMMs, where each phoneme is typically represented by three CD states (an inside state and two transition states): illustrated in Fig. 2.

When we have got a feature vector extracted from a frame of the acoustic signal, the next step at both decoding and training stages is to define the likelihood of this feature vector given an HMM state. While the observation likelihood is $p(o_t|q_t)$, where o_t is an observation and q_t is an individual state, the probability of a sequence of states is the product of transitional probabilities between the states [Medennikov, 2015]:

$$P(q_1^T) \approx \pi_{q_1} \prod_{t=1}^{T-1} \alpha_{q_t q_{t+1}} \tag{3.2}$$

Then, the joint probability of a sequence of observations and a sequence of states is the product of the $p(q)$ and $p(o|q)$ and the full probability of a sequence of observations is a sum of such joint probabilities:

$$P(o_1^T) = \sum P(o_1^T, q_1^T) = \sum P(o_1^T|q_1^T)P(q_1^T) \tag{3.3}$$

[15] a_Beginning, a_End, a_Inside, a_Single.

This probability can be computed with the forward-backward algorithm [Baum and Petrie, 1966]. The forward algorithm is used to calculate the likelihood of observations so far and the backward algorithm is used to predict the probability of seeing all upcoming observations, where training is initiated with the initial state distribution.

Therefore, the HMM model describes the dynamic process of speech, encoding its transmissions based on the observation likelihoods and the probabilities derived from the pronunciation lexicon. How are the observation likelihoods estimated? In decoding, for example, we have got an observation o_t and the probability $p(o_t|q_t)$ for each possible HMM state given this observation should be computed in order to reveal the most likely sequence of states.

Speech is not a categorical process and its feature vectors are real-valued numbers that enable a better description of the wide variability of the speech signal. To get the probability of continuous input feature vectors given a limited number of state-'categories', we need to choose some parameters to distinguish similar feature vectors and such an approach that would compute $p(o_t|q_t)$ on real-valued observations. The two possible ways to get posterior probability estimates for the HMM states are acoustic models based on either GMM or DNN.

3.2 Gaussian Mixture Models

The more traditional approach of estimating the observation likelihoods on the real-valued, continuous feature vector is computing a probability density function (pdf) with the method of Gaussian Mixture Models. The Gaussian distribution, or the normal distribution, is a type of continuous probability distribution, which is described by the parameters of a mean (μ) and a variance (σ^2):

$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(x-\mu)^2}{2\sigma^2}) \tag{3.4}$$

We can use this formula of a univariate Gaussian pdf to estimate the likelihood that a particular HMM state produces a value of a feature vector[16]. Thus, we define the observation likelihood function for one value of an acoustic vector as a Gaussian (Fig. 3). Since an acoustic observation is a vector of multiple values, to take into account all values of a feature vector, a multivariate Gaussian is used. A Gaussian is parameterised by a mean vector $\overrightarrow{\mu}$ and a diagonal covariance matrix Σ[17] that

[16] Assuming that the values of observation feature vector o_t are normally distributed.

[17] *The covariance matrix* is a matrix containing the covariance of each pair of elements of a given

contains the variance of each value (dimension) as well as the covariance between any two values (dimensions):

$$f(\overrightarrow{x}|\overrightarrow{\mu},\Sigma) = \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma|^{\frac{1}{2}}}\exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$
$$= \prod_{d=1}^{D}\frac{1}{\sqrt{2\pi\sigma_{jd}^2}}\exp\left(-\frac{1}{2}[\frac{(x_{td}-\mu_{jd})^2}{2\sigma_{jd}^2}]\right) \quad (3.5)$$

where D is a dimension of a feature vector given HMM state j. Building Gaussian distribution, we make an assumption of a normal distribution of the data. A cepstral feature, however, can be far from being normally distributed. Therefore, the observation likelihood is usually modelled with a weighted mixture of multivariate Gaussians, called a Gaussian Mixture Model, instead of a single multivariate Gaussian (Fig. 3). Finally, for computational convenience and to increase the computational speed, in speech recognition, log probabilities are usually used that helps to avoid multiplication and exponentiating.

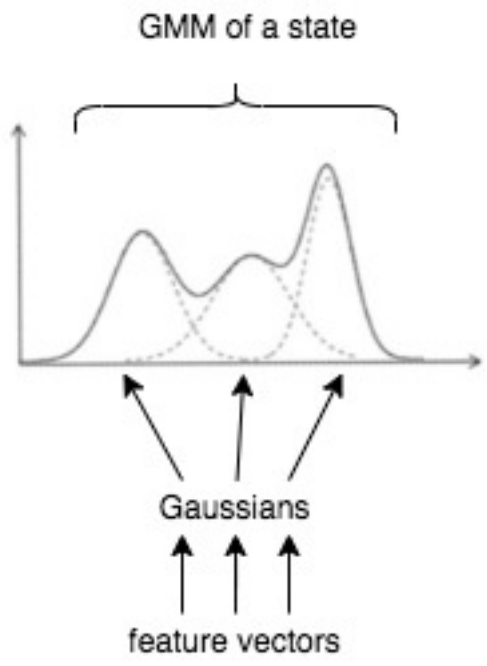

Figure 3: Gaussian Mixture Models.

At the decoding stage, the likelihood of an observation feature vector given a GMM of a state j with its mean, mixture weight, and covariance matrix is estimated.

random vector. *Covariance* provides a measure of the correlation between two sets of random variates. In the diagonal of the matrix, there are the covariances of each element with itself (variances) [Weisstein, 2003].

Training the acoustic model, however, we do not know apriori which mixture corresponds to which part of each distribution. The Baum-Welch algorithm is usually used to learn on the labelled data the probability of a certain mixture accounting for the observation with the initial mean and variance set identically for each Gaussian, to the global mean and variance for the entire training data. The algorithm iteratively updates the counts until getting the best probability, $\xi(t_i)$, the probability of being in state i at time t [Baum and Petrie, 1966]. In the basic way, the GMM-HMM acoustic model is trained according to Maximum Likelihood (ML) objective function:

$$F_{MLE}(\lambda) = \sum_{t=1}^{T} \log p_\lambda(O_t|M_{s_t}) \tag{3.6}$$

where λ represents the acoustic model parameters, O_t are observations (or training utterances), M_s is an HMM model corresponding to a sequence of states s, and s_t is the correct transcription for the t^{th} utterance. The ML function is optimised via the Expectation Maximisation (EM) algorithm [Moon, 1996]:

$$\theta_1 = \underset{\theta_1}{\operatorname{argmax}} \sum_{o \in D} E_{\theta_2} \log p(O, \theta_2; \theta_1) \tag{3.7}$$

where O are observations, θ_1 are parameters (a, b) for transition and emission probabilities and θ_2 are parameters (γ, ξ) for internal state distribution (μ, σ^2) and transition. At the Expectation-step, θ_1 parameters are optimised by the forward-backward algorithm given the current θ_2 and O. At the Maximisation-step, the probability distribution of θ_2 is established given θ_1 and O.

GMM have been used for speech recognition for a long time, yet, they have some limitations:

1. GMM are not flexible enough for statistically efficient modelling data that lie in the nonlinear data space. Modelling the data points, which are close to the surface of a sphere requires a very large number of diagonal Gaussians or a large number of full-covariance Gaussians that is statistically very ineffective [Hinton et al., 2012; Yu and Deng, 2016].

2. GMM can not efficiently deal with multiple simultaneous events, as it assumes that each datapoint is produced by one component of the mixture only.

3.3 Deep Neural Networks acoustic models

Another way to get the acoustic observation likelihoods is to train a DNN-based model. Neural networks are potentially much better in learning nonlinear relationships of data. Moreover, using different subsets of their hidden units, DNN are more successful in modelling multiple simultaneous events within one frame or window comparing to GMM, which are highly localised and, therefore, often result in only performing local generalisation [Dahl et al., 2011]. For the last few years the computational capacity of the hardware has been considerably improved, and DNN with several hidden layers became competitive outperforming GMM-HMM models [Hinton et al., 2012].

There are two common configurations of DNN-based acoustic models called *hybrid* and *tandem* [Yoshioka et al., 2014; Parthasarathi et al., 2015]. In the hybrid configuration, the DNN aim to predict CD HMM states (emission likelihoods) that are then directly used for the decoding. In the tandem approach, the DNN perform nonlinear discriminative feature transformation. The discriminatively transformed 'features', or vectors of probabilistic features instead of raw posterior probabilities, are used after as input to a generative acoustic model based on GMM-HMM or another DNN-HMM hybrid model. The tandem approach was popular in the early studies of DNN in acoustic modelling and was a first attempt of combining discriminative and generative training that led at that moment to a considerable improvement [Hermansky et al., 2000]. Another similar method of getting features with DNN is using bottle-neck systems with 5-layer MLP architecture where 'bottle-neck' in the middle layer helps to reduce the size of the system [Grézl et al., 2007].

A speech signal is a time series, that is why those DNN architectures that require an input signal of a fixed length (e.g. feed-forward ones) cannot be directly used to model the dynamic nature of speech. Then, in such cases, the HMM component stays unchanged in the training setup. As a GMM-based model, a DNN model works as a sequence classifier to predict the best phoneme label for an acoustic input. The output distribution is defined at the final layer of the DNN with the softmax nonlinearity function (see more about DNN in [Goldberg and Hirst, 2017]). Cross entropy (CE) loss function is typically used as an objective function:

$$J_{CE}(W, b; o, y) = -\sum_{i=1}^{C} \hat{y}_i \log y_i \tag{3.8}$$

where $\hat{y}_i$ is the observed probability that o belongs to the class i and y_i is the model prediction. Optimisation of the objective function is usually done with stochastic

gradient descent method (SGD). The alignment of CD states to frames derives from the GMM baseline systems and is fixed during the training.

There are different possible DNN architectures that can be used for acoustic modelling, among which are deep believe networks (DBN) [Hinton et al., 2006], connectionist temporal classification (CTC) [Sak et al., 2015], recurrent neural networks (RNN) [Graves et al., 2013], time-delay neural network (TDNN; see more in section 6.2.1) [Waibel et al., 1989], and convolutional neural networks (CNN) [LeCun et al., 1995][18].

3.4 Decoding and training

3.4.1 Decoding

When vectors of observation likelihoods (or emission probabilities) are obtained from the acoustic model (either GMM or DNN) for each frame, the goal at the decoding step is to define the most probable string of words. First, at each time point, the joint probability of observation and state is computed for each state; then, the product of this joint probability and the prior probability from the language model (see eq. 2.1-2.6) is computed in order to find the most likely sequences of words. More than just the best scoring hypothesis are usually generated. An efficient way to construct and to encode multiple sentence hypotheses is using a word lattice (or word graph). During training, a number of lattices with the state and word probabilities and state-level alignments is usually used for further model optimisation.

A convenient representation of word lattice is as a weighted finite state transducer (WFST) [Mohri et al., 2008; Povey et al., 2012]. The WFST decoding graph combines the main components of speech recognition with the algorithms of composition ($\circ$), minimisation (min) and determinisation (det)[19] and is defined as:

$$HCLG = \min(\det(H \circ \min(\det(C \circ \min(\det(L \circ G)))))) \tag{3.9}$$

where H is the HMM structure that keeps transition probabilities, C is a phonetic

[18] End-to-end models are usually trained with the CTC criterion or attention-based sequence-to-sequence models, where the attractive property of CTC is that it enables end-to-end optimisation of acoustic models [Maas et al., 2017].

[19] "In a *deterministic automaton*, each state has at most one transition with any given input label and there are no input ϵ-labels $<\ldots>$ The key advantage of a deterministic automaton over equivalent nondeterministic ones is its irredundancy"; "Given a deterministic automaton, we can reduce its size by minimization, which can save both space and time" [Mohri et al., 2008].

context-dependency that maps states of triphones to phonemes, L is a lexicon that keeps pronunciation probabilities and G is a grammar, or language model, that keeps n-gram probabilities.

WFST works as a finite state automaton (machine) that has a set of states, including one start state and final states, and a set of arcs between the states that have weights, or probabilities, of the transitions between these states. Its state transitions are labelled with input and output symbols and a path through the transducer maps from an input symbol string to an output one [Partee et al., 2012].

Transducer allows the combination of different levels of representation, for example, between phones and words, or between HMMs and CD phones. Thus, representation of several components of decoding at the same time is possible (see Fig. 4). This property is implemented with the operation of *composition* that combines Transducers together. When two Transducers are composed ($T_{new} = T_1 \circ T_2$), a new Transducer has exactly one path mapping an input sequence to an output sequence $u : w$ for each pair of paths[20], the first in $T1$ mapping u to some string v and the second in $T2$ mapping v to w. The path weight in T is calculated from the weights of two corresponding paths in $T1$ and $T2$ [Mohri et al., 2008, p. 563].

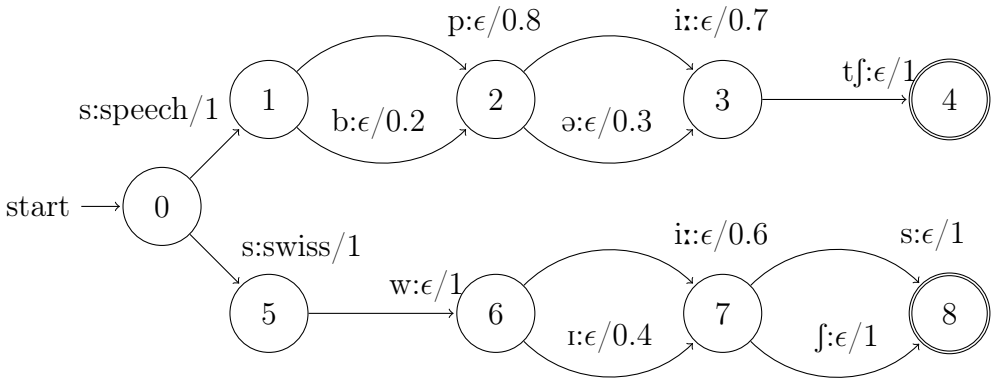

Figure 4: Weighted finite-state transducer example illustrating a relationship between two levels of representation: pronunciation lexicon and language model $L \circ G$ (The input label i, the output label o, and weight w of a transition are marked on the corresponding directed arc by $i : o/w$).

Another two operations of WFST, determinisation and minimisation, are meant for the goal of optimisation. A *determinisation* algorithm aims to eliminate redundant paths that helps to reduce the decoding time. It searches for the transitions with the same labelling between the same two states, calculates weights of paths with such transitions, and then removes or combines the identically labelled paths in

[20]Pair of paths is a pair of a $T1$ state and a $T2$ state (a pair of composition).

such a way that each state has maximum one transition with any given input label and no input ϵ-labels[21]. A *minimisation* algorithm aims to reduce the size of a deterministic WFST minimising a number of states and a number of transitions among all deterministic weighted automata.

The weighted transducer generates the *search graph* of an utterance of T frames that is used to find the best decoding path. To optimise the search through the search graph, Viterbi algorithm with beam pruning is typically employed. Viterbi algorithm is a dynamic programming search algorithm that finds the optimal path through the decoding graph [Forney, 1973]. It is close to the forward algorithm but instead of summing up the previous path probabilities, it takes the maximum over them. Given a beam size of K, at each step only the K most likely paths are left, when the other ones are pruned to reduce the search space.

With oracle WER[22] instead of one-best WER, decoding proved to considerably improve the final score ([Povey et al., 2012]). The beam size, however, influences the performance of the decoding: a lower K limits the choice of potentially better candidates but reduces the time of the decoding process; an optimal beam size is considered to be around 10.

Decoding typically outputs a word level lattice that contains probabilities from the acoustic and language models. This output can be already used to decide on the final sequence, or for training a model, or as input for further models. A set of alignments and lattices generated by decoding the training data with a unigram LM is often a starting point for training GMM-HMM and DNN-HMM systems [Veselỳ et al., 2013]. The alignments and lattices can be also generated with the cross entropy pre-trained DNN models.

3.4.2 Training

Training an acoustic model is the process of adjusting model parameters in accordance with a given sequence of observations to increase the likelihood of this sequence of observations for a modified model. As mentioned above, standard GMM-HMM based acoustic models are typically trained with EM algorithm using ML criteria[23] (Eq. 3.6) and DNN-HMM models are trained with CE loss function (Eq. 3.8). Both

[21] In speech recognition, pronunciation lexicon can be not determinisable, i.e. in the case of ambiguities, or homophones. Then, auxiliary phone symbols can be introduced at ends of such words to avoid ambiguities and that will create a transformed lexicon $\tilde{L}$ [Mohri et al., 2008].

[22] The oracle (lattice) error rate is "the word error rate we get if we chose the lattice path (the sentence) that has the lowest word error rate" [Jurafsky and Martin, 2009, p. 341].

[23] Training criterion is another way to call an objective function.

training methods are used for training generative models and aim to independently classify individual frames fitting the best model to the data. Speech recognition task is, however, a sequence classification problem. Therefore, sequence-discriminative training of acoustic models was proposed to optimise parameters towards the goals of speech recognition [Bahl et al., 1986; Povey, 2005]. There are different sequence-discriminative criteria, i.e. maximum mutual information (MMI), minimum phone error (MPE), state-level minimum Bayes risk (sMBR), and boosted MMI.

The MMI criterion [Bahl et al., 1986; Kapadia et al., 1993; Woodland and Povey, 2002] aims to maximise the mutual information between the distributions of the sequence of observations and the sequence of words:

$$F_{MMI}(\lambda) = \sum_{t=1}^{T} \log \frac{p_\lambda(O_t|M_{s_t})^\kappa P(s_t)}{\sum_s p_\lambda(O_t|M_s)^\kappa P(s)} \tag{3.10}$$

where λ represents the parameters of the acoustic model, O_t are observations (or training utterances), M_s is the HMM sequence corresponding to a sentence s, and s_t is the correct transcription (or sequence of states) for the t^{th} utterance, κ is a probability scale and $P(s)$ is a language model[24]. In other words, the numerator in Eq. 3.10 is the correct transcription, and the denominator is a model with all possible words. During the training with MMI, we are looking for such λ that will maximise the numerator and at the same time minimise the denominator discriminating in this way between the correct sequence and all other sequences.

The criteria of the Minimum Bayes Risk (MBR) family are aimed at minimising the expected error at the phoneme level (MPE [Valtchev et al., 1996; Povey and Woodland, 2002]), word level (MWE [Povey, 2005]) or HMM states (sMBR [Goel and Byrne, 2000]). These criteria are more directly related to WER, typical scoring measure in continuous speech recognition. Generally, they are close to the MMI function but the equation also includes the sum of accuracy values that are maximised during the training:

$$F_{MPE}(\lambda) = \sum_{t=1}^{T} \log \frac{\sum_s p_\lambda(O_t|M_s)^\kappa P(s) A(s, s_t)}{\sum_s p_\lambda(O_t|M_s)^\kappa P(s)} \tag{3.11}$$

where $A(s, s_t)$ is the raw phone accuracy of the predicted sequence of states s given the correct sequence s_t.

[24] In Eq. 3.10-3.12, the notation s is used to denote a sequence of elements of different levels (words, phonemes) to illustrate better the similar nature of these three criteria.

Finally, another sequence-discriminative criterion proposed by Povey and colleagues [2008] is the boosted MMI, which also takes accuracy in consideration but boosts its effect with a boosting factor b penalising the sentences that have more errors:

$$F_{bMMI}(\lambda) = \sum_{t=1}^{T} \log \frac{p_\lambda(O_t|M_{s_t})^\kappa P(s_t)}{\sum_s p_\lambda(O_t|M_s)^\kappa P(s) \exp(-bA(s, s_t))} \tag{3.12}$$

Therefore, the effect of bMMI is proportional to the number of errors in a candidate sentence.

The model-space training experiments comparing the MMI, MPE and bMMI discriminative training criteria show that MMI-based training works slightly better than MPE and the error-boosting introduced by bMMI leads to an additional small improvement [Povey et al., 2008]. At the same time, the previous studies indicate that the MPE function can be more robust than MMI when training data is noisy: "words and files which have not been transcribed correctly during preparation of the training data, or are so poorly pronounced as to have no realistic chance of being correctly recognised, will not be given as much importance by MPE as by MMI" [Povey, 2005, p. 23]. Another set of experiments on different data sets has confirmed that models with discriminative objective functions consistently outperform models trained with the ML or CE criteria [Veselỳ et al., 2013]. Yet, the difference between the different discriminative criteria was small for all the data sets.

Originally, NN were trained discriminatively only, but the results of further experiments showed that *generative pre-training* helps to reduce overfitting, and it also reduces the time required for *discriminative fine-tuning* with backpropagation [Yu et al., 2016; Hinton et al., 2012]. In [Erhan et al., 2010], it is reported that pre-training can be seen as a kind of data-dependent 'regularisation'[25] that helps the subsequent optimisation preformed by SGD. Firstly, a model is trained as a generative model, e.g. with CE function, then, sequence-level discriminative fine-tuning is performed with a discriminative objective function. The main idea behind sequence-discriminative training is taking into account the sequential nature of speech recognition and using an objective function more closely referred to the goal of the ASR training, i.e. WER score. Purely discriminative training of the whole DNN from random initial weights is possible, but brings lower accuracy [Yu et al., 2010]. Using generative training in combination with sequence-discriminative training improves word error rates by 7-9% relative, on average [Veselỳ et al., 2013]. Thus, generative pre-training followed by the sequence-discriminative fine-tuning is the most popular

[25] *Regularisation* is an algorithm used to improve the generalisation of a model. Usually a regularisation function is applied during the parameters optimisation [Goodfellow et al., 2016].

setup for hybrid acoustic modelling.

A common characteristic of discriminative objective functions is that all of them have a numerator and a denominator. To represent the numerator, we create a lattice based on the true transcriptions of a sequence. To represent the denominator, for each utterance we use a lattice containing all possible predictions and that is generated in decoding. For DNN-based models, such a denominator lattice is often pre-trained with cross entropy as a generative part of generative-discriminative setup. This leads to the high accuracy results but also to difficulties when training the DNN on GPUs (graphics processing unit). Recently, *lattice-free MMI* was introduced that allows the MMI training directly on GPUs applying the forward-backward algorithm for both numerator and denominator parts of the training criterion [Povey et al., 2016]. Instead of lattices, the phonetic-context decision tree is built using the same features (MFCC+LDA+MLLT) as a standard hybrid model. The denominator-lattice-free MMI training got 8% relative improvement over the sMBR objective with cross entropy pre-training.

3.5 Adaptation techniques

Due to the high variability of speech, there are many potential challenges leading to the decrease of model performance, including noise in the training data and significant mismatches between the training and decoding conditions. Adaptation is an essential component of speech recognition systems to compensate for speaker and channel variations. Above, I have already mentioned some methods used for feature adaptation (see 2.3), but some techniques can be applied to the model training itself. First of such techniques appeared more than 20 years ago to improve GMM models; along with them other methods are developed to follow the demand of DNN acoustic modelling.

3.5.1 GMM specific adaptation

One way to improve the performance of recognition system is linear transformation, which aims to maximise the average likelihood. A commonly used method is maximum likelihood linear transform (MLLT)[26] that can be applied in either feature-space (see 2.3) or model-space [Gales, 1998].

Other adaptation techniques aim to adjust model parameters to improve its perfor-

[26]MLLT is also called Semi-Tied Covariance (STC) [Gales, 1999].

mance in the conditions different from training. Adaptation can be supervised, when additional adaptation data is used, and unsupervised where a model prediction is used as the referent text. In GMM-HMM framework, such adaptation methods as Maximum Likelihood Linear Regression (MLLR) [Leggetter and Woodland, 1994] and Maximum a Posteriori Linear Regression (MAP-LR) are widely used [Chesta et al., 1999]. Linear regression adaptation methods are used to estimate cluster-dependent regression matrices in order to adapt speaker-independent HMM mean vectors to a new speaker.

3.5.2 DNN specific adaptation

Various approaches have been developed for neural network acoustic model adaptation [Miao et al., 2015; Wang et al., 2019] and they can be roughly categorised into four classes:

1. speaker-adapted layer insertion (linear transformation, which augments the original network with certain speaker-specific linear layers);
2. adaptation in the input space (a low dimensional speaker subspace; speaker adaptive training (SAT) — training with speaker-specific features (fMLLR, iVectors); speaker codes — a specific set of network units for each speaker is connected and optimised with the original speaker-independent network);
3. direct model adapting (to use new speaker's data to adapt the DNN parameters directly).

3.5.3 Adaptation with iVector

The iVector method allows efficient modelling of both the intra-domain and inter-domain variabilities into the low dimensional total variability space and can be used as a feature adaptation technique in standard ASR [Miao et al., 2015]. Saon et al. [2013] gave a good example illustrating why the iVector technique originally introduced for speaker recognition goals [Dehak et al., 2010] can be still useful in speech recognition. When two speakers have different pronunciation types of the same phoneme, e.g. /AA/ and /AE/ for the phoneme /AA/, an acoustic model without speaker-specific features will tend to classify these two variants as two different phonemes. Speaker features will help the model to learn to increase the output score for /AA/ phoneme for the second speaker but not for the first speaker with /AA/ type because, if there is something that sounds like /AE/ in the first speaker, it

should be classified as /AE/ and not as /AA/.

Training of the iVector extractor involves two main steps:

- the Universal Background Model (UBM), which contains generalised information about all the data, is estimated. The UBM is trained without making difference between speakers (or dialect) and is necessary for collecting general statistics from speech utterances. Formally, the UBM is a big GMM model with K diagonal covariance Gaussian components. The acoustic feature vectors x_t of all the observations can be defined as a sum of these components:

$$x_t = \sum_{k=1}^{K} c_k N(.; \mu_k; \Sigma_k) \tag{3.13}$$

  where c_k are mixture coefficients, μ_k are means and Σ_k are diagonal covariances.

- while adapting the background model to the training utterances, total variability matrices T_k are learnt and after used to extract data-specific iVectors. Speaker-adapted mean $\mu_k(s)$ of the speaker s is defined as:

$$\mu_k(s) = \mu_k + T_k w(s), k = 1...K \tag{3.14}$$

  where $w(s)$ is this speaker's iVector, which is a vector of values conveying the information specific for the target speaker comparing to the UBM.

3.5.4 Subspace Gaussian Mixture Models

Subspace Gaussian Mixture Models (SGMM) do not directly belong to the adaptation techniques. Nevertheless, SGMM are known for their ability to generalise well: the model parameters are derived from low-dimensional model and speaker subspaces that helps to capture acoustic and speaker correlations better [Povey et al., 2010; Povey et al., 2011]. The GMM-HMM models independently train a separate GMM for each HMM state. In the SGMM framework, the HMM states share the same structure and vary in their parameters, means and mixture weights. As the parameters are not specified directly, a vector space $v_j \in \mathbb{R}^S$ is introduced in each CD state j. The parameters vary in a subspace of the full parameter space and this variation is defined by a mapping from a vector space v_j to the general pdf (GMM) parameters. Thus, the term 'subspace' means here that the parameters for each state are defined in a subspace of the general space of parameters. The most basic

definition of SGMM is a GMM:

$$p(x|j) = \sum_{i=1}^{I} w_{ij} \mathcal{N}(x; \mu_{ij}, \Sigma_i) \tag{3.15}$$

where the parameters are defined as:

$$\mu_{ij} = M_i v_j \tag{3.16}$$

$$w_{ij} = \frac{\exp(w_i^T v_j)}{\sum_{i'=1}^{I} \exp(w_{i'}^T v_j)} \tag{3.17}$$

with covariance matrices Σ_i shared between the states and the derived parameters of mixture weights w_{ij} and means μ_{ij}.

The GSMM framework allows better level of generalisation and is typically recommended when only little data is available or little in-domain data [Povey et al., 2011]. It is often used in a combination with speaker-specific fMLLR features as a form of speaker adaptation.

3.6 Summary

In the chapter, the main principles of acoustic modelling have been presented. The two basic AM frameworks are GMM and DNN with the dynamic component in both cases enabled with HMM. There are different training techniques (discriminative, SGMM, etc.), to improve the performance of the acoustic models, including the adaptation techniques, which are especially important for the high variability data observed in Swiss German. In Chapter 6, I will come back to this question of acoustic modelling techniques where I will test different options to improve the baseline model.

4 ASR of Swiss German

The goal of this chapter is to provide an overview of the current status of ASR for Swiss German. First, the major difficulties of dialect speech-to-text in the context of Swiss German are discussed. Then, the available corpora for Swiss German are presented with a focus on the ArchiMob corpus, as being the main data resource in the present project. Finally, I overview the results of previous studies both in multi-dialectal ASR in general and for Swiss German in particular.

4.1 Challenges of Swiss German ASR

ASR for Swiss German differs from the ASR of standardised languages, first of all, because Swiss German is not a single language but a collection of several dialects, which, despite being related, are quite different. This causes certain challenges for turning speech into the text that are common for multi-dialect speech recognition [Najafian et al., 2016]: i.e. data collection, data annotation, building ASR system, and its evaluation.

4.1.1 Inter- and intra-dialect variability

As already mentioned in the introduction, linguistic situation in the German speaking part of Switzerland can be defined as *diglossia*, where two languages used for different types of communication are Standard German and Swiss German. Swiss German is characterised with high inter-dialect variability. Traditionally, four main dialect groups dividing along the north-south and the east-west axes are distinguished [Hotzenköcherle et al., 1984]. These four groups are usually associated with the dialects of Bern, Chur (the canton of Grisons), Brig (the canton of Valais), and Zürich. Such a division is rather convenient and fits well some research purposes (e.g. [Zihlmann]). At the same time, it is a simplification and does not cover many dialectal variations.

The results of hierarchical cluster analysis on the data from the *Sprachatlas der*

deutschen Schweiz (SDS) and the *Syntaktischer Atlas der deutschen Schweiz* (SADS) showed that different classifications of dialects can be used for different linguistic levels ([Scherrer and Stoeckle, 2016]; see clustering for phonological, morphological, sintactic, and lexical levels in Fig. 5). The difference between linguistic levels makes it imposssible to unambiguously define dialectal boundaries. Dialects change in a more continuous rather than categorical mode and we can speak about both inter- and intra-dialect variability.

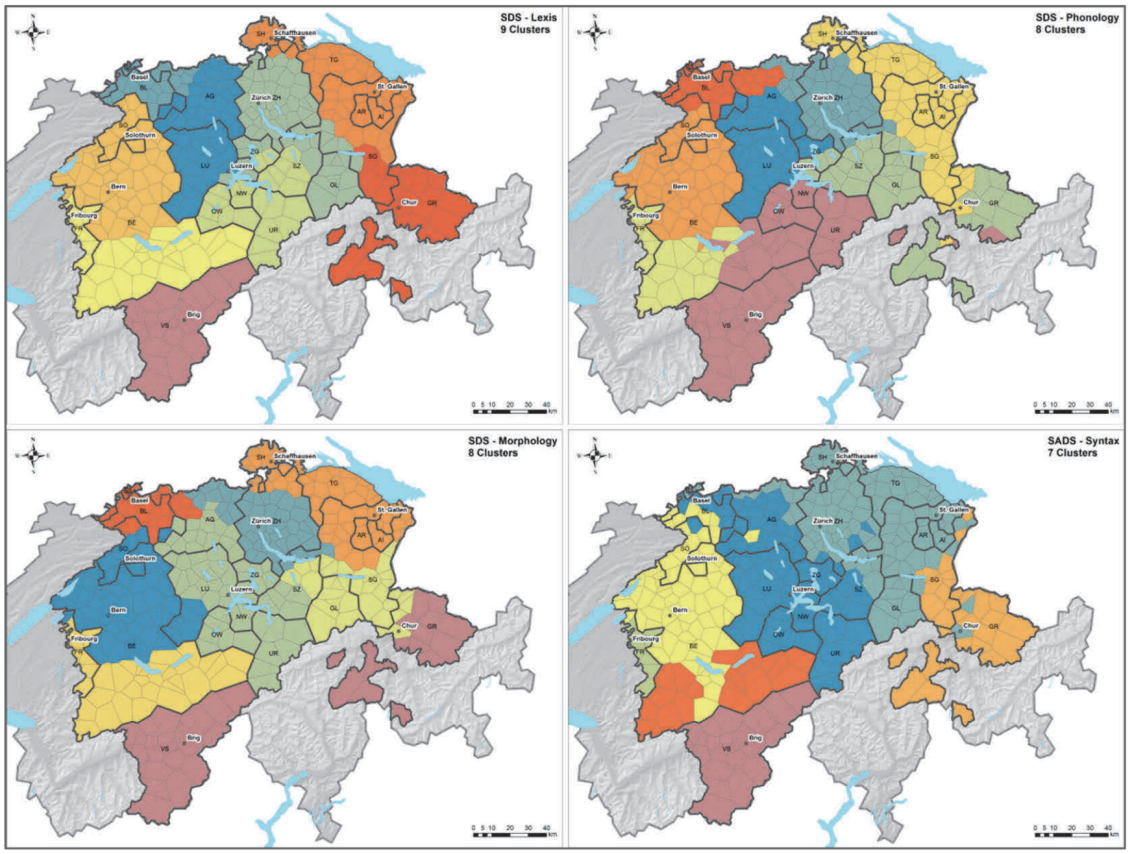

Figure 5: Distribution of dialects for different linguistic levels: (from left to right) lexicon, phonology, morphology, syntax [Scherrer and Stoeckle, 2016].

Since Swiss dialects vary a lot on all the linguistic levels — phonological, morphological, syntactic, and lexical — that causes particular difficulties for the ASR task. Morphological and lexical variability increases the data sparsity when the size of the lexicon grows up. This leads to many words with low frequency, and, at the evaluation stage, to a higher number of out-of-vocabulary (OOV) words. Syntactic variability reduces the certainty of language model causing more deletion and substitution errors.

The order of words or syntactic constructions can change across the dialects, which makes it harder than in a standard language to learn unified syntactic rules. For

example, the purposive clause construction of Standard German *um ... zu* 'in order to ...' *um ein Billet zu lösen* 'in order to buy a ticket' can have the following regional variants:

1. **zum + (z) + infinitive** (*zum e Billett löse* (St.Gallen) and *zom es Billett z'löse* (Luzern));
2. **für + (z) + infinitive** (*für n'es Billet z'lösä* Glarus, Valais and southern Bernese region) [Glaser et al., 2012, Map 3].

On the phonological level, it means that some dialects have phonemes (especially vowels), which are acoustically different or absent in the other dialects. For example, the middle vowel in the word *Fliege* is pronunced as /ü/, /ii/, /öi/, /öü/, /äü/, /äi/, /ie/, /ö/, /ü/, /ei/ depending on the region[27]. The first difficulty for the acoustic modelling caused by such diversity is that some phonemes can be represented with much more allophonic variants, which will increase the sparseness of the data. The second issue is that the set of all the phonemes is large but many of the phonemes are underrepresented in the data. In the last case, a training algorithm often does not have enough data to properly model some of the phonemes.

Another difficulty in the situation of many dialects is defining the tied CD states that can be variety specific (senones; see 3.1). The acoustic tree used for clustering into the senones is typically learnt on the full data. For the multi-dialect model, however, this method can be suboptimal, as modelling inter-dialect variations with intra-dialect senone contexts leads to a mixed-nature acoustic tree. To reduce this effect, Arsikere et al. [2019] proposed the *phone mapping* approach, according to which one of the dialects (and its acoustic tree) is taken as a canonical and the phone sets of all other dialects are mapped to the canonical phone set. Another possible solutions to deal with the acoustic variability can be 1) training separate acoustic models (AMs) for different dialects, or 2) add adaptation techniques to make the model more resistant to the variability. In this work, we will use a single acoustic model and general clustering on the entire data, as for learning which dialect should be used as a canonical one, more experiments are needed.

4.1.2 Absence of orthography

Languages without orthography are called non-standardised, because there are no officially accepted rules about their spelling. As originally, in the situation of diglos-

[27] `http://dialektkarten.ch/mapviewer/swg/index.en.html#dominance:1136_Fliege` (08.04.20).

sia, Swiss German serves for the purpose of spoken communication, it does not have a conventional writing system. The lack of standardisation is the second major challenge for ASR of Swiss German and it causes the following problems:

- without standardised writing, the full consistency in transcriptions of audio files can hardly be achieved because it will never be free from the subjectivity of annotators (e.g. using *ggsii, gsii* and *gsi* for the same word).
- there is no homogeneity between different annotated corpora of spoken dialects. Often the annotation discrepancy is an obstacle for using available data sets together, for example, in order to increase training data.
- many existing spelling variants (see Table 4) lead to extremely sparse data that impairs the quality of alignments between the acoustic data and transcriptions.

The optimal annotation guidelines should be defined to maximise inter-annotator agreement and to attain the best possible alignment between the acoustics and writing. Ideally, the guidelines should keep the balance to be self-descriptive but not redundant at the same time. Moreover, the main demand for writing to be good for the transcription purpose is its consistency. The safest way would be to address already existing conventions. In the case of Swiss German, the choice of spelling rules is not straightforward. Swiss German is a spoken variety with conventional orthography. In some situations, however, it is used in writing.

Mundartliteratur. One of such situations, for example, is a region-specific tradition as in the canton of Bern [Marti, 1985]. Caduff [2015] speaks about three periods of rise of *Mundartliteratur* 'dialect literature'. The first wave took place in 1900 with Otto von Greyerz, Simon Gfeller, Carl Albert Loosli; the second, known as *Modern Mundart*, happened in 1968 in connection with such names as Mani Matter, Kurt Marti and Ernst Eggimann[28]. The last period started a decade ago and the most famous writers in dialect now are Guy Krneta and Pedro Lenz, who published their novels "Unger üs" and "Goalie bin ig" in Bern German [Lenz, 2014]. Despite these few *Mundartliteratur* experiences, the writing conventions stay very individual and cannot be used as rules: "Die Schreibweise des Dialektes ist keineswegs in Stein gemeisselt. Zwar gibt es zu den Dialekten einzelne Regelwerke, aber im Gegensatz zum Duden[29] sind diese kaum bekannt, so dass Leserinnen und Leser gar nicht merken können, wo die Autoren von welchen Regeln abweichen. Vielmehr **obliegt es jedem einzelnen Schriftsteller, seine eigenen orthografischen Formen**

[28]A Wikipedia article about Ernst Eggimann written in Swiss German: `https://als.wikipedia.org/wiki/Ernst_Eggimann` (08.04.20).

[29]Duden is a famous dictionary of the Standard German language: `https://www.duden.de/` (08.04.20).

und Regeln zu erfinden — und sie auch wieder zu ändern"[30] [Caduff, 2015, p. 10].

Other spontaneous writing. Another case of written Swiss German has appeared due to the recent development of computer-mediated communication (CMC) ([Siebenhaar, 2006]; NOAH's corpus[31] [Hollenstein and Aepli, 2015]; the sms4science corpus[32] [Ueberwasser and Stark, 2017]). This kind of communication is usually used in chats and forums, hence it can be characterised as an unofficial and everyday communication type, i.e. 'spontaneous writing'. Recently the first multilingual corpus of CMC has been created in Switzerland. The corpus consists of more than million WhatsApp messages shared by Swiss people and contains 43% of messages written in Swiss German and only 7% — in Standard German [Ueberwasser and Stark, 2017].

Although well represented nowadays in online communication, spontaneous writing in Swiss German is highly diverse due to many factors: local dialect, standard influence, individual interpretation of phoneme to grapheme rules, individual interpretation of orthographic principles, regional-writing traditions, typing errors (especially because of the missing autocorrection option) [Siebenhaar, 2006, p. 483; Clematide et al., 2016]. Moreover, people are inconsistent in their individual spelling preferences. Therefore, following CMC writing for transcribing purposes cannot be considered an optimal solution either.

Dieth transcription. Finally, there is a spelling approach proposed by Swiss linguist Eugen Dieth [Eugen and Dialäktschrift, 1938] and mostly used in the academic context. Dieth's intention was to create a writing system, which would be based on the orthography of Standard German with an extension to the Swiss German phonology. The method was aimed to accommodate all Swiss German dialects and offered a set of rules for writing dialects without introducing any extra characters in addition to the ones of Standard German.

On the one hand, Dieth spelling principles are well adapted for Swiss German and, at the same time, understandable for non-linguists, which made them suitable as a foundation for a potential orthography. On the other hand, the absence of a dictionary left speakers the freedom to rely on what they hear that can no means be absolutely consistent. Besides the knowledge of dialects, the use of Dieth writing

[30] "The spelling of the dialect is by no means set in stone. There are individual sets of rules for dialects, but in contrast to the Duden, they are hardly known, so that readers cannot even notice where and from which rules the authors deviate. Rather, **it is up to each individual writer to create his/her own orthographic forms and invent rules** - and change them again." - translated by IN.

[31] `https://pub.cl.uzh.ch/wiki/public/kitt` (19.05.20).

[32] `http://www.sms4science.ch/` (19.05.20).

requires certain learning and practice.

Along with the basic spelling set, the extended 'narrow' spelling (*enge Dieth-Schriebung*) was proposed including additional diacritics to provide a more detailed representation of the phonetic quality. Yet, the Dieth guidelines do not provide enough consistency and there can be many ambiguous cases [Gordon et al., 2016]. Moreover, sometimes the use of a grapheme is dialect-specific, e.g. the grapheme <e> stands for [e], [ɛ], [ə] in different dialect. Nevertheless, in order to transcribe Swiss German material, linguists and dialectologists are mostly using the Dieth writing. Although in reality, the adaptation of additional guidelines is often needed, at the moment, the Dieth spelling stays the most conventional transcription method for Swiss German [Gordon et al., 2016].

4.1.3 Quality estimation of the non-standardised languages ASR

Non-standardised writing plays an important role in the evaluation of ASR. The usual metrics of evaluation, such as word error rate (WER), character error rate (CER) assume that only one word or character is correct for each particular position in a sequence (see 2.2). For languages without orthography, there can be multiple possible variants of spelling for the same word, all of which are correct. Hence, neither WER nor CER are flexible enough to provide the true picture of the system performance. WERd (word error rate for dialects) method of 'soft' evaluation for Arabic dialects was proposed by [Ali et al., 2017]. Mapping the hypothesis to the reference, spelling variants are considered correct, if they belong to the same pair of matched words (5.3.1). This flexible kind of evaluation also seems reasonable in the situation of Swiss German dialects. Therefore, for our evaluation along with WER and CER, I have chosen to use a flexible measure of evaluation too (see more in 5.3.1).

4.2 Swiss German speech resources

One of the necessary conditions on the way of training a good ASR system is the availability of a large collection of high-quality audio data transcribed and aligned manually. The potential sources of such language data collections are television, radio, proceedings of the Parliament sessions (e.g. the Europarl-ST corpus [Iranzo-Sánchez et al., 2019]), and the Internet. Creating such a linguistic resource for Swiss German is complicated by the fact that most of these large information sources with publicly available and easy accessible data cover the official side of Swiss life and,

hence, are in Standard German. Swiss German stays the unregistered language of daily communication. Another obstacle is that writing tradition is not standard and all the data collected is very variable and needs to be normalised (4.1.2). In this section, I will, first, overview the existing spoken corpora for Swiss German; then, I will give a more detailed description of the ArchiMob corpus, as the main data resource in this project.

4.2.1 Swiss German speech data sets

Swiss-German SpeechDat(II) FDB-2000. In the frames of the SpeechDat project [Höge et al., 1999], 28 databases for all official languages in the European Union and their major dialects including Swiss German were created. 2'000 Swiss-German speakers were recorded over the Swiss fixed telephone network and with the average duration of recording equals 3.5 minutes and a pronunciation lexicon with a phonemic transcription in SAMPA[33]. The databases were designed for training special-purpose recognisers and, hence, their vocabularies are quite limited and biased (for example, containing many digits). It makes the corpus not suitable for the large vocabulary SR purpose, as it will result in a big number of OOV words and will be biased to a specific vocabulary and contexts. Moreover, the usage of the corpus is not free[34].

MediaParl corpus. The MediaParl corpus is a bilingual corpus of parliamental debates in Swiss German and Swiss French recorded at the Valais parliament in Switzerland [Imseng et al., 2012]. A Swiss German part contains 8'526 sentences, or 20 hours of speech, in different accents spoken in Valais. For the pronunciation lexicon a phonemic transcription in SAMPA is used.

Radio Rottu Oberwallis5 (RRO) 'Walliserdeutsch' data set[35]. The audio data provided by a local radio station in the canton of Valais broadcasting in Swiss Standard German and Valais German dialect. The data set consists of 2'010 sentences or around 8.5 hours of speech (including training and test sets together); it is manually annotated with the dialect quasi acoustic transcription by native speakers of Valais German.

[33] Speech Assessment Methods Phonetic Alphabet (SAMPA) is a machine-readable form of the International Phonetic Alphabet (IPA): `https://www.phon.ucl.ac.uk/home/sampa/index.html`. (10.04.20)

[34] `http://catalogue.elra.info/en-us/repository/browse/ELRA-S0105/` (10.04.20).

[35] `https://www.idiap.ch/dataset/walliserdeutsch` (10.04.20).

SRF Meteo weather report data set. A Schweizer Radio und Fernsehen[36] Meteo weather report data set contains 290 Meteo weather report broadcasts with a total of 6.5 hours of Swiss Standard German speech transcribed with the Standard German orthography.

Schawinski Transcripts. This is a small corpus of spoken Zürich German consisting of six TV shows with Roger Schawinski[37]. The corpus contains 2'500 spoken utterances manually transcribed according to the ArchiMob annotation guidelines based on the Dieth spelling approach.

Phonogrammarchiv Zürich Text-Korpus (PAZTeK). PAZTeK is a collection of more than 3'000 recordings over a period between 1909 and 1996 from all over Switzerland[38]. It contains 15'500 spoken Swiss German utterances transcribed closely to the Dieth spelling guidelines. The corpus, however, is still under development.

The ArchiMob corpus. The ArchiMob corpus is a publicly available large spoken corpus of 14 Swiss German dialects containing interview recordings with different speakers. The corpus is approximately 70 hours long, is manually transcribed by native speakers, and the original manual transcriptions are further semi-automatically normalised (see more 4.2.2).

Table 2 below is an overview of all the presented corpora for spoken Swiss German:

Table 2: Swiss German spoken corpora (SSG stands for Swiss Standard German)

Corpus	**Size, hours**	**Variety**
SpeechDat(II)	100	SSG
MediaParl	20	SSG+Valais
Walliserdeutsch data set	8.5	SSG+Valais
SRF Meteo	6.5	SSG
Schawinski Transcripts	3	Zürich
PAZTeK	33	Swiss German
ArchiMob	70	14 Swiss German dialects (see Table 3)

The choice of a data set has an important impact on the performance of a recognition system. The ArchiMob corpus is planned to be used because it is the largest

[36] `https://www.srf.ch/` (10.04.20).

[37] `https://www.srf.ch/sendungen/schawinski/sendungsportraet` (10.04.20).

[38] `https://www.phonogrammarchiv.uzh.ch/de/sammlung/corpora.html` (10.04.20).

available annotated corpus of continuous speech with the broad coverage of Swiss German dialects. This choice allows training a multi-dialect ASR system for Swiss German. From the transcription and data organisation point of view, the Schawinski transcripts is the most similar to the ArchiMob corpus and can be used as an additional data. Yet, it differs in a certain way: 1) on the topic of interviews and 2) on the age of the speakers. The second difference is especially important for acoustic modelling, as including different age groups brings more variability into the acoustic signal (see 4.2.2: Peculiarities). The small size of the corpus would not help to balance the age groups of speakers but it can be used as a test set to see how well the system deals with out-domain data.

4.2.2 ArchiMob

The audio material for the ArchiMob corpus was taken from the video interviews recorded by the Archimob association[39] in 1999-2001. Having recorded 555 interviews with Swiss citizens who experienced the Second World War, the Archimob project aimed to collect first-person stories about the life in Switzerland in the middle of the 20th century.

43 interviews have been selected according to the sound quality and linguistic criteria. The interviews with Swiss German speakers only who were not exposed to contact with other languages or dialect varieties were taken. The duration of each interview is between one and two hours. The latest release of the corpus consists of approximately 70 hours of raw speech data from 14 different Swiss German dialects ([Scherrer et al., 2019a]; Table 3). A dialect is defined according to the canton, where a speaker of this dialect comes from (see Table 3). The ArchiMob corpus provides the information about the speakers' origins, and this approximation would allow us to avoid ambiguity.

Table 3: Number of interviews per canton

Canton	**N**	**Canton**	**N**
Zürich	12	Aargau	6
Bern	5	Luzern	5
Basel-Stadt	4	Nidwalden	2
Glarus	2	Graubünden	1
Uri	1	Basel-Landschaft	1
Schwyz	1	Schaffhausen	1

[39] `http://www.archimob.ch/f/index.html` (11.04.20).

Canton	N	Canton	N
Valais	1	St. Gallen	1

Two types of transcription. All the recordings were digitised and audio was extracted. The interviews were manually split into utterances of 4-8 seconds, aligned with the sound source and transcribed by five annotators. The ArchiMob corpus of Swiss dialects provides two types of transcription. The first one is an acoustic-phonological transcription done according to the orthographic guidelines based on the writing principles proposed by Dieth, also called *dialectal* (4.1.2). Annotators, native speakers of Swiss German, manually transcribed speech material according to their intuition of correct pronunciation. This 'dialectal' writing provides close correspondence between grapheme labels and the acoustic signal but is extremely noisy due to the lack of consistency.

The second type of transcription is derived from the first ('dialectal') transcription and is its normalised version. Normalisation here means that all the writing variants of the same word are mapped to a single 'normalised' form in high German (Table 4). When a Swiss German word has no etymologically related equivalent in Standard German, a reconstructed common Swiss German form is used as normalisation: for example, *öpper* 'someone' is normalised as *etwer* and not *jemand* that is a direct translation to Standard German but has different semantic root [Scherrer et al., 2019b, p. 9]. Thus, such normalisation cannot be seen as translation: a sentence structure always stays unmodified and only word writing is changed.

Table 4: Normalised and Dialectal (Dieth) transcriptions

Normalised	Dialectal
abbauen	abbaue, abboue, abbuue
abend	aabe, aabed, aaben, aabet, aabid, aabig, abed, abend, abet, abig, abud, obet, obig, oobig, zabig, äbig, òòbed, òòbig
mitbekommen	mitbecho, mitbechoo, mitbichoo, mitbikho

After normalisation the vocabulary size is reduced from 48'537 to 31'755 tokens that has considerably reduced the lexicon sparsity. Hence, normalisation has potentially

a positive effect on the language model [Kew, 2020]. Its benefit to the acoustic modelling is less clear, because the variation of allophones within phonemes (first of all vowels) is much higher when using the normalised transcriptions. Therefore, our experiments will be done on the original dialectal transcriptions.

Peculiarities. One of the characteristics of the ArchiMob corpus that is worth to be mentioned in the context of acoustic modelling is that all the interviewed speakers were older than 67 with the average age of 82[40]. It is important for ASR, as due to the articulation changes in older people, the acoustic features of their voices usually differ from the acoustic characteristics of young people. Elderly voices can have such acoustic effects of ageing, as F0 and amplitude instability, jitter[41], shimmer[42] and increased breathiness that according to speech recognition experiments can decrease the recognition accuracy [Vipperla, 2011]. Perception experiments have also proved that elderly voices are less intelligible for listeners compared to the young voices [Shuey, 1989; Williams and Giles, 1991].

Predominance of older speakers in the ArchiMob corpus can influence the general performance of AMs. Speaker independent (SI) models trained on the data balanced in terms of age and gender return better results for young adults than old adults test data: with WER of 15.9% and 20.4% correspondingly [Vipperla, 2011, p. 78]. The results can be improved by the adaptation of the data towards test groups but difference in WER scores between young and old speakers stays big: 14.9% and 18.4% correspondingly [p. 112]. On the one hand, in the current project, speech data used for training, development and main evaluation sets are all taken from the ArchiMob corpus and the homogeneity of the data in regard of age should minimise the train-test discrepancy problem. On the other hand, lower intelligibility of old people speech can still have a strong impact on the recognition results.

4.3 Related work

There are not many studies on Swiss German dialects ASR. This can be partially explained by the lack of audio data in dialects: television usually uses Swiss German, or accented Standard German. Regional radio and television channels transmit in dialects, but the amount of data is typically not enough for training a high-quality

[40] The age of the speakers is explained by the original purposes of the recordings: collecting historically important interviews with Swiss people who lived during the 20th century and were able to evidence the changes in the last 50-70 years.

[41] Jitter — "measure of temporal perturbations in glottal source periods" [Vipperla, 2011, p. 80].

[42] Shimmer — "measure of amplitude perturbations in glottal source periods" [Vipperla, 2011, p. 80].

ASR system. Another obstacle is the absence of standard writing that has been discussed in the previous section 4.1.2. It involves two issues: a) the difficulty in training when the same word has multiple variants of pronunciation, and b) the low readability of the output of a system trained on Swiss German. In the unnormalised form, it is hard for reading and understanding and additional postprocessing including normalisation or translation into Standard German is needed before the results can be used. First, I will talk about the previous speech-to-text experiments for multi-dialect goal in general; then, about the studies on the Swiss German data.

4.3.1 Studies on multi-dialect ASR

Meeting the challenge of multi-dialect acoustic modelling, there is a choice to follow one of two ways: a) training dialect-specific AMs, or b) training a single unified AM that would generalise well for many different dialects.

Dialect-specific models usually outperform a single model but sometimes training a model for each dialect can be not possible, e.g. when not enough data per dialect is available. To deal with the data sparsity problem, in [Yang et al., 2018] (English dialects), dialect-independent parts of an AM are jointly training with data from all dialects and dialect-dependent parts are trained separately with dialect-specific data. Another method to get dialect-specific models and benefit from all available data is adapted in [Li et al., 2018] (English dialects) where they trained a single AM with all the data and then fine-tuned it on the dialect-specific data. Training several models for one language, however, is not always convenient, especially for a conversation with a few speakers of different dialects, and computationally more expensive; it demands a more certain definition of dialects, and in speech recognition applications often a preliminary step of dialect identification will be required. Therefore, most of the studies on multi-dialect speech-to-text aim to improve the single AM approach applying different adaptation methods [Elfeky et al., 2016 (Arabic dialects); Ali et al., 2016 (Arabic dialects); Yoo et al., 2019 (English dialects)].

One way for better training a single multi-dialect AM is a conditioning input approach for NN AMs, which is a method to dynamically generate the scaling and shifting parameters in layer normalisation [Ba et al., 2016]. Dynamic generation works by fixing the mean and the variance of the summed inputs within each layer. Originally proposed for image-modelling [Dumoulin et al., 2016], conditioning was successfully applied for speaker adaptation in [Kim et al., 2017] (English dialects) and for more general-level adaptation in [Yoo et al., 2019]. Another possibility is adding a dialect-specific input, for example, iVectors (see also 3.5.3) or dialect in-

formation [Jain et al., 2018] (English dialects). Jain et al. [2018] used as dialect information utterance-level dialect embedding extracted from a dialect classifier. Utterance iVectors are used in many previous studies on speech recognition, as they work well to facilitate not only speaker but also channel and background normalisation [Senior and Lopez-Moreno, 2014; Najafian et al., 2017 (Arabic dialects); Arsikere et al., 2019 (English dialects)].

The most similar to Swiss German dialects situation is probably the one with Arabic dialects. There are many Arabic dialects that strongly differ from each other on all linguistic levels but the standardised orthographic writing is only available for Modern Standard Arabic (MSA). The Arabic SR system trained with 200 hours of MSA achieved 31.72% WER [Ali et al., 2014]. The Arabic multi-dialect SR system trained on 1'200 hours and evaluated with data from different dialects has got 14.7% WER [Ali et al., 2016].

4.3.2 Studies on ASR for Swiss German

There are only few studies on ASR for Swiss German and most of them are region-specific. The first speech-to-text system was trained when the MediaParl corpus appeared [Imseng et al., 2012]. The system was trained on the Swiss German data set from the MediaParl corpus with the triphones-based GMM-HMM acoustic model and the WER equaled 68.4%. Later, three other speech-to-text systems were trained on the same data set: with three-layer ANN-HMM [Razavi et al., 2014], with three-layer CNN-HMM [Palaz, 2016] and with three-layer sMBR discriminative ANN-HMM optimised towards the state-level error rate [Dubagunta and Doss, 2019][43]. The results for the improved models were 25.5%, 23.5% and 18.7% of WER correspondingly.

Another system trained on the *Walliserdeutsch* data set and proposed by Garner et al. [2014] achieved 19.4% of WER with the DNN-HMM hybrid architecture used for the acoustic model. The audio data was a mixture of Swiss Standard German and Valais dialect. A low error rate in this case is explained 1) by the fact that the language model was trained on both training and evaluation sets, 2) as well as that the data was mostly from one dialect.

Stadtschnitzer and Schmidt [2018] built a recogniser for the Swiss Standard German speech, which output was further translated into Standard German. The authors used two different corpora for training two systems: 1) on a large corpus of Standard

[43] In all three systems, the bigram language model provided with the MediaParl corpus was used [Imseng et al., 2012].

German and 2) on a small corpus of Swiss German. The standard German corpus is GerTV1000h German Broadcast corpus that consists of approximately 1'005 hours of German speech data from different domains [Stadtschnitzer et al., 2014]. The Swiss German corpus is the SRF Meteo weather report data set. Both systems were evaluated with the same test set in Swiss German and the results on a smaller Swiss-specific data were considerably better comparing to the Standard German one: 23.8% against 56.4%. Such a big difference evidences again the large gap between Standard and Swiss German. Nevertheless, Swiss Standard German does not include all the variability of Swiss German dialects, it is quite consistent, and the system's evaluation with different dialects will probably return lower results.

The difference in the evaluation on different dialects is well illustrated with the results from [Scherrer et al., 2019b]. In the first study on large vocabulary ASR with a broader coverage of Swiss German dialects, a recognition system was trained on Zürich dialect data and evaluated with 6 other dialects. The results in Table 7 show that the F-score for different dialects vary from 22.37 to 49.68. The data from several Swiss dialects was taken from the first release of the ArchiMob corpus (see 4.2.2). Therefore, so far, there has been no such a universal ASR system that would be trained on many different Swiss German dialects at the same time.

4.4 Summary

If since the last few years speech technology has already gained high accuracy results for standardised languages ([Chiu et al., 2018, Xiong et al., 2018, Wang et al., 2019]; section 2.4), for non-standardised languages with high regional variation and limited training resources, such as Swiss German, it remains a big challenge. Since in the present work, I aim to work on the multi-dialectal ASR problem, the ArchiMob corpus is taken as a data resource, as being the largest available open spoken corpus of Swiss German dialects. The contribution of this work to the ASR for Swiss German is a) training the first multi-dialectal ASR system for Swiss German; b) training the multi-dialectal ASR system using the dialectal transcriptions. The improvement of the baseline model (Chapter 5) will be mostly focused on the normalisation and adaptation of the data in order to reduce the effect of data sparsity caused by its variability (Chapter 6).

5 Replicating the baseline model

In this chapter, I will present the ASR model that has been chosen as a baseline for my experiments with acoustic modelling. The original model was built with Kaldi ASR framework (see 2.4) by Spitch AG. The initial training data is around 21 hours of recordings from the first release of the ArchiMob Swiss dialects corpus (Chapter 4). After replicating the Spitch AG baseline (section 5.1), I trained the baseline setup with more in-domain data from the second ArchiMob release: approximately 60 hours of sound (section 5.2). Pipeline details of the baseline system and its evaluation results are reported and discussed. The goals of the chapter are 1) to provide an overview of the starting point for further acoustic modelling experiments, 2) to describe the procedure of adding more data of a different format, 3) to fix the data split for training/development/evaluation sets that stays the same for further experiments, 4) to report the evaluation results for the system trained on the first and on the second releases of the corpus.

5.1 Replication with the initial data set

In the frames of collaboration between Spitch AG and the University of Zürich, Spitch AG provided the basic Kaldi[44] ASR pipeline adapted for the ArchiMob data [Campillo, 2018], which I use as a start point for my experiments. The original pipeline trains and evaluates the system on 24 interview recordings (about 23 hours of training and 2 hours of evaluation data) that already had transcriptions in EXMARaLDA format (EXB) available by that moment [Schmidt and Wörner, 2012]. The pipeline is based on the *wsj* Kaldi recipe that was created for training speech recognition on the Wall Street Journal corpus. Compared to the original recipe, which contains the whole training-decode-evaluation pipeline in a single `run.sh` script, Spitch AG pipeline is split into three core bash scripts, which correspond to three major steps of speech recognition: 1) training of the acoustic models

[44]Kaldi version 5.3. was used. The current Kaldi version is 5.5. (05.03.20).

(`train_AM.sh`[45]), 2) integrating the language model (`compile_lingware.sh`[46]), and 3) decoding step along with the evaluation of the model (`decode_nnet.sh` — for acoustic models based on neural networks (for other acoustic models see section 5.3.2)). Before running the training script, the input raw data must have been preprocessed in order to have the format needed by Kaldi. All the experiments were conducted on the university server, first, on Ubuntu 16.04, then, on Ubuntu 18.04 with the GPU installed.

5.1.1 Data preparation

General preprocessing. The original input data for the Spitch AG pipeline was 1) video recordings of the whole interviews (approximately 1.5h per document), and 2) their transcriptions in the EXMARaLDA format, which we call annotations. The EXMARaLDA annotations contain two aligned parts: a) transcriptions of phrases (or *events*), and b) the timeline with information about the beginning of each phrase (absolute values in seconds). From video files audio recordings were extracted. Consequently, the python script `process_exmaralda_xml.py` enables the parallel processing of audio recordings and annotations, and chunks all the recordings according to the timeline from the annotations where each element is an utterance. As a result, the preprocessing outputs:

a) audio chunks time aligned with the corresponding transcriptions of utterances;

b) a single .csv file, which contains all transcriptions and is used further as an input annotation file for Kaldi; besides the transcription itself, .csv file also keeps the additional information about the events, such as their duration, and whether they are speech or non-speech[47] fragments.

At the beginning of the `train_AM.sh` script, the further preprocessing of the ArchiMob data is done to prepare files for Kaldi: all necessary scripts are called from the `prepare_Archimob_training_files.sh` bash script. At this step, a *pronunciation lexicon* is also created (see 2.1). A simple pronunciation lexicon is mainly built with one-to-one grapheme to phoneme correspondence. However, there are some grapheme clusters that should be mapped to a single phone. To map such clusters correctly, a manually created list of graphemic clusters is used. This list

[45] The names of the scripts of the original pipeline were introduced by Francisco Campillo [Campillo, 2018].

[46] *Lingware* — the name is taken from one of the Spitch AG product names.

[47] Non-speech fragments can be, for example, noise at the beginning of the interview or silence between the replicas.

contains, for example, such mappings as *sch - sch*, *ää - ä*, *ts - z*, *tsch - tcsh, z ch*, or *v - f*. When more then one mapping variant is possible for a grapheme (or a cluster), all variants are kept in the lexicon:

```
aarbetsdientscht a r b e z d i e n tcsh t
aarbetsdientscht a r b e z d i e n z ch t
```

The `prepare_Archimob_training_files.sh` script uses the .csv transcription file, chunked audio files, and a list with graphemic clusters as input and aims:

1. To map all non-speech fragments to a special symbols (`<SPOKEN_NOISE>`, `<SIL_WORD>`).

2. To generate files that are needed for training acoustic models or decoding with Kaldi:
   `wav.scp` — a file that contains mappings from utterance ids to wave locations; `spk2utt` — a file that contains mappings from speaker ids to utterance ids; `utt2spk` — a file that contains mappings from utterance ids to speaker.

3. To create a simple *pronunciation lexicon* that is used later to build a better mapping between the audio signal and the transcription symbols.

Kaldi data preparation. The second part of data preparation is done with the internal Kaldi scripts called from the `utils/prepare_lang.sh` and aims to create additional data files needed for training and decoding. These files are a) separate files for dictionary elements (phones, silence symbols, states, words) and b) files with mapping between different elements (phonemes to their possible states, words to their phoneme transcriptions, words to their state transcriptions; for more information see Appendix Table 8). Word-to-states transcription mapping also includes additional labels to distinguish the cases when two words with different graphemic representations have the identical phoneme transcription:

```
aafò 1.0 a_B f_I ò_E #1
aafòò 1.0 a_B f_I ò_E #2
```

5.1.2 Feature extraction

The acoustic features are MFCC features with Cepstral Mean and Variance Normalization (2.3) extracted from the signal with sample frequency of 8kHz (`steps/make_mfcc.sh`, `steps/compute_cmvn_stats.sh`). Further feature transformations were additionally applied on the top of basic MFCCs already during

the training and depending on the type of the acoustic model (see above about the features transformation methods: section 2.3), including:

- adding delta and delta-delta features (`add-deltas.cc`);
- LDA+MLLT transformation (`transform-feats.cc`).

If additional feature transformations are applied, then, it is important to have the same feature transformations during the decoding stage to keep the uniform processing of train and test data.

5.1.3 Training acoustic models

Having the data and feature files as input (see the full list of files in Appendix Table 8) training of acoustic models starts. With the Spitch AG baseline script, six acoustic models are trained in the pipeline: a monophone GMM model, three triphone GMM models (tri, tri_lda, tri_mmi), and two neural network-based models (nnet2, discriminative):

1. GMM mono: Gaussian Mixture Models for monophones (e.g., /a/) trained with maximum likelihood.
2. GMM tri: Gaussian Mixture Models for basic triphones (e.g., /p-a+t/) trained with MFCC+Δ+$\Delta\Delta$ features and maximum likelihood.
3. GMM tri+lda: triphone system with LDA+MLLT — trained with maximum likelihood, but with an initial kind of LDA transformation on the acoustic features.
4. GMM tri+lda+MMI: triphone GMM system with the LDA transformation and discriminative training; the above system was trained discriminatively using first feature-space boosted MMI (fBMMI) and then model-space boosted MMI [Povey et al., 2008].
5. NNET: p-norm neural network model, trained with maximum likelihood.
6. NNET-DISC: the same neural network from NNET, but trained with a discriminative criterion sMBR.

During the training process, acoustic models are not simply trained separately, but as a sequence of models in order to improve the performance of each next more complicated model using the alignment between phonemes and acoustic signal built by a previous model (see 3.4). The first model in a sequence is a context-independent

(CI) monophone model that is trained from a flat start. When a model is trained, the alignment built with this model is used afterward to train the following model. A new better alignment of a new model is used for the next model and, therefore, each subsequent model benefits from stronger input alignments. By the end of the training stage, there are separate folders for each of the models, each of which has a model file (`final.mdl`) and a folder with the alignment files (`\ali`). If feature transformation is applied, final feature matrix is also included (`final.mat`).

5.1.4 Decoding

The second core bash script of the pipeline compiles the acoustic and language models, as well as the linguistic data derived from the preprocessing step, to create a fully expanded decoding graph (`utils/mkgraph.sh`). The decoding graph consists of four components $HCLG = H \circ C \circ L \circ G$ (see more in 3.4.1; eq. 3.9; Table 5) and represents the WFST-based decoding in Kaldi. To enable code-level integration with WFST, in many algorithms, Kaldi compiles against the open-source WFST toolkit — **OpenFst**[48] — which is used as a library. The decoding graph, therefore, compiles all the word-level grammar (G.fst), pronunciation dictionary (lexicon, L.fst), context-dependency (CLG.fst), and HMM structure (Ha.fst) transducers by the means of OpenFst:

Table 5: Transducers used in the ASR decoding

	Transducer	**Input sequence**	**Output sequence**
H	HMM	HMM states	CD phones
C	context-dependency	CD phones	phones
L	pronunciation lexicon	phones	words
G	language model	words	words

The H component of the graph contains the HMM definitions. It takes as input such information as HMM-states, phones, transition-states, and transition-ids, which encode the pdf-ids, i.e. indices for the probability distribution function (p.d.f., Gaussians); its output symbols are context-dependent (CD) phones. The C component defines the context-dependency: its input symbols are CD phones (a_B, a_E, a_I, a_S; see 3.2) and its output symbols are phones (a)[49]. The L component maps the

[48] `http://www.openfst.org/twiki/bin/view/FST/WebHome`.

[49] From the Kaldi documentation: `http://kaldi-asr.org/doc/graph.html`, `http://`

sequences of phones to most likely words. Finally, the G component of language model chooses the best sequence of words.

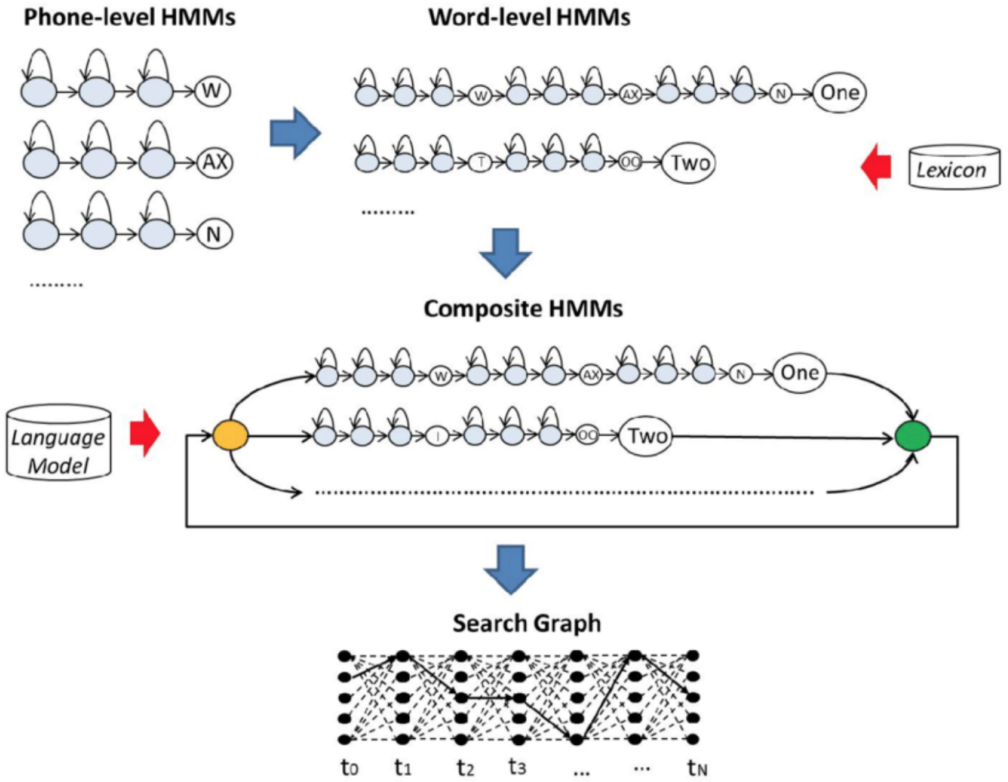

Figure 6: Transducer components of the decoding graph in Kaldi (from [Ravanelli, 2017]).

The default language model is a statistical 3-gram language model in a text format called ARPA. The model was trained on the dialectal transcriptions with the MIT toolkit[50].

The last core script (`decode_nnet.sh`) extracts features from the test data, executes decoding on the test data using the decoding graph, and evaluates the system output. A summary of the whole framework offered by Spitch AG is shown in Table 6:

`kaldi-asr.org/doc/hmm.html#transition_model_identifiers`.

[50] `https://github.com/mitlm/mitlm`.

Table 6: Framework of the ASR system for ArchiMob prepared by Spith AG; from [Campillo, 2018]

Step	Input	Output	Scripts
1. Data preparation	ArchiMob wavefiles, Exmaralda files	ArchiMob csv, chunked wavefiles	extract_audio.sh process_exmaralda_xml.py
2. AM training	ArchiMob csv, chunked wavefiles	Acoustic models	train_AM.sh
3. Lingware	Acoustic models, vocabulary, language model	HCLG	compile_lingware.sh simple_lm.sh
4. Evaluation	Acoustic models, references wavefiles, HCLG	WER	decode_nnet.sh

I used the bash scripts provided by Spitch AG as the basic pipeline to train, decode and evaluate with Kaldi. Nevertheless, some modifications were introduced to optimise the code and eliminate its redundancy. The pipeline was originally split into three separate bash scripts to emphasise three main stages of the SR process: i.e. training, graph compilation and decoding (here at the test stage). In order to clarify what elements are important for each particular stage, some of the files were created twice by different scripts: e.g. pronunciation lexicon and language files are created in both `train_AM.sh` and `compile_lingware.sh`. To simplify running further experiments, I put the pipeline into two bash scripts merging the compiling and decoding steps into a single `compile_decode.sh`.

5.1.5 Evaluation

The original system was trained on 22 interviews (14 interviews in Zürich dialect and 8 in other Swiss dialects) and evaluated on 2 interviews in Zürich dialect. The system got WER of 75.20% [Campillo, 2018, Table 8, p. 12]). Later, the system was trained on 22 interviews with speakers of Zürich dialect only and evaluated on 6 other Swiss dialects [Scherrer et al., 2019b, Table 3]. The best F-score obtained was 49.68 for Grisons dialect but results differ a lot between the dialects: e.g. F-score=29.83 for Bern dialect and F-score=22.37 for Luzern dialect.

Table 7: Scores for the system trained and evaluated on different dialects

Train	**Test**	**Score**
14 interviews of Zürich + 8 interviews of other dialects	2 interviews Zürich	WER: 75.20
22 interviews of Zürich	Bern	F-score: 29.83
	Basel	F-score: 45.40
	Grisons	F-score: 49.68
	Lucerne	F-score: 22.37
	Uri	F-score: 30.46
	Valais	F-score: 36.96

The results depend on:

- The amount of training data. The baseline models were trained on approximately 25 hours of continuous speech that is very little for the task of ASR.
- In-domain and out-domain evaluation. Speech from different interviews in the ArchiMob corpus can be very different because of the inter-dialectal variability. When evaluated with the same or different dialect than used in the training, the result can vary a lot: compare the evaluation of the system trained on Zürich dialect and evaluated on Luzern (F-score 22.37) vs Grison dialect (F-score 49.68) (Table 7).
- High variability inside training data. Besides the acoustic and linguistic difference due to the dialects, the training data is also very diverse because of: richness of the Dieth's transcription approach used for annotation, which, for example, distinguishes vowels of different length, and inter-annotators difference.

Possible data specific improvements:

- **Language model.** The limited amount of data and its variability lead to the high percentage of out-of-vocabulary (OOV) words in the evaluation: about 8% of the words. When language model includes all the available data (both the training and evaluation sets), WER considerably dropped down from 75% to 33% [Campillo, 2018]. Besides the data augmentation, a better language model can be trained on the normalised version of transcriptions (Chapter 4). Such a language model eliminates multiple writing variants, which, otherwise,

are recognised as different words.

- **Acoustic model.** The information about the phone length from the transcriptions can not be modelled properly with the HMM. That is why, according to the clusters, all double vowels should be mapped in the pronunciation lexicon to a single one. Because of this, some words become homophone, as the distinctions between them is not caught by the acoustics:
  gsi 1.0 g_B s_I i_E #1
  gsii 1.0 g_B s_I i_E #2
  guete 1.0 g_B u_I e_I t_I e_E #1
  guuete 1.0 g_B u_I e_I t_I e_E #2
  gwont 1.0 g_B w_I o_I n_I t_E #1
  gwoont 1.0 g_B w_I o_I n_I t_E #2
  hèrzig 1.0 h_B è_I r_I z_I i_I g_E #1
  hèèrzig 1.0 h_B è_I r_I z_I i_I g_E #2

- **Underestimating the performance.** The richness and complexity of the Dieth's transcription (see Chapter 4), as well as the absence of a standard written form for every word lead to the high sparsity of the training data. The same word often has several written variants and then not only single correct reference. Thus, evaluation of languages with a non-standardised orthography is not always optimal with WER (see Chapter 4). One possible way to reduce the negative effect of transcription variability is to use flexible WER, which would allow more than one correct spelling variants, for example, by analogy with the WERd approach proposed by Ali and colleagues [2017] (see more in sections 4.1.3 and 5.3.1).

In order to improve the recognition results of the baseline model, in this thesis I will focus on the acoustic modelling; the language model used in the experiments will be always the same as of the baseline system — a statistical 3-gram language model (5.1.4). I will also try different evaluation approaches, including flexWER and out-of-domain evaluation, to achieve better understanding of the improvements. The first step towards the improvement is adding more in-domain data, which is especially crucial for training DNN models.

5.1.6 Summary

The baseline setup is a hybrid DNN-HMM model when a sequence of six models is trained to improve the result of a final GMM alignment and, hence, the final DNN model. The result of evaluation is highly dependent on the distribution of dialects in

the training and evaluation data, as well as on the amount of training data. In the next section (5.2), the results of models trained and evaluated on more data from the second ArchiMob release are presented. The training, development and evaluation sets containing all the dialects and speakers from the corpus are also proposed.

5.2 Replication with increased in-domain data set

Since the time when the first Spitch AG pipeline was built, more interviews have been annotated, and the amount of available transcribed speech data has increased up to 43 interviews, or around 70 hours of raw speech material. Since adding more relevant training data often helps to improve the results, especially with more complicated models, our first step towards the improvement was training the system on the full ArchiMob corpus data (the second release)[51]. In this section, I will focus on the procedure of adding the new data and will present the evaluation results for GMM and DNN models included in the pipeline of the baseline setup trained on the full ArchiMob corpus.

5.2.1 Adding data from the second ArchiMob release

The freshly added annotations in the ArchiMob corpus are in the XML format, which has replaced the EXB format [Scherrer et al., 2019b]. The choice of XML is due to its higher flexibility that allows direct modifications, if needed, as well as adding new information, including new transcriptions, meta-information, and tags (Fig. 7).

```
<tli id="T3" time="6.333331016613322"/>
<tli id="T4" time="10.433329516841944"/>
<tli id="T5" time="12.433329516841944"/>
<tli id="T6" time="14.546661345526596"/>
<tli id="T7" time="16.199994074074073"/>
<tli id="T8" time="19.933326041761926"/>
<event start="T3" end="T4">/ lauft /</event>
<event start="T4" end="T5">mich würd interessiere /</event>
<event start="T5" end="T6">wie si iri jugendzit /</event>
<event start="T6" end="T7">als iischtiig /</event>
<event start="T7" end="T8">ää$ erläbt hend /</event>
<event start="T14" end="T15">und wo hend si / gschtudiert /</event>
```

```
<u start="media_pointers#d1008-T4" xml:id="d1008-u4" who="interviewer">
  <w normalised="mich" tag="PRF" xml:id="d1008-u4-w1">mich</w>
  <w normalised="würde" tag="ADV" xml:id="d1008-u4-w2">würd</w>
  <w normalised="interessieren" tag="VVINF" xml:id="d1008-u4-w3">
  interessiere</w>
</u>
<u start="media_pointers#d1008-T5" xml:id="d1008-u5" who="interviewer">
  <w normalised="wie" tag="KOUS" xml:id="d1008-u5-w1">wie</w>
  <w normalised="sie" tag="PPER" xml:id="d1008-u5-w2">si</w>
  <w normalised="ihre" tag="PPOSAT" xml:id="d1008-u5-w3">iri</w>
  <w normalised="jugendzeit" tag="NN" xml:id="d1008-u5-w4">jugendzit</w>
</u>
```

Figure 7: Examples from the EXB (left side) and XML (right side) formats.

The audio recordings originally were one-interview-long files. Recently, they have been chunked into short audio fragments, where each replica is in a separate .wav

[51] https://www.spur.uzh.ch/en/departments/research/textgroup/ArchiMob.html.

file and, thus, no further split of audio data is needed. In order to align the transcriptions and .wav files, the information about time (the beginning and the end of each fragment) is registered in the XML transcription files with the *start* attribute (Fig. 7). Therefore, the new data differs from the data used for the baseline system training:

a) in format of the input transcriptions (XML instead of EXB);

b) in the length of the audio files (utterance-long instead of interview-long).

This difference has required certain modifications in the data preparation step.

5.2.1.1 Preparing CSV transcription file

In order to enable choosing an input format between EXB or XML as an option, I have correspondingly adapted the `process_exmaralda_xml.py` and `archimob_chunk.py` scripts. As in the original version, the output of the preprocessing is a .csv file, where each line is an utterance. This format is convenient to create necessary data files for Kaldi (5.1.1), which requires all utterance transcriptions to be organised horizontally, i.e. an utterance per line. In the new version, however, the output .csv file contains some additional information per utterance with the following nine columns:

utterance-id	unique ID starting with an interview ID following with a speaker ID
Dieth	dialectal transcription provided by annotators following Dieth transcription rules
normalised	normalised dialectal transcription
speaker-id	speaker ID
audio-id	the original name of the corresponding audio file
anonymity	binary: whether or not the utterance contains any personal information masked with ***
speech-in-speech	binary: whether or not the utterance overlaps with another according to the time information
missing-audio	binary: whether or not the corresponding audio files exists and is not empty
no-relevant-speech	binary: whether or not the utterance has a non-speech content

The `process_archimob_csv.py` script processes all utterances in order to fill the four binary columns, i.e. anonymity, overlapping of replicas, absence of a corresponding audio file, and whether it is a speech relevant fragment or not. These binary features are used later on to filter the utterances that are not suitable for training data. For example, 'anonymity' marks those fragments that contain personal information about speakers, e.g. their names, and cannot be shared. Personal

information was replaced in transcriptions with asterisks symbols ***, which creates noise during the alignment between the transcription and audio. Overlapping occurs when two speakers talk at the same time. Overlapping replicas are also to be filtered, as the alignment is based on the time information and it is not possible to align to more than one transcription at a time. Finally, empty utterances and utterances that miss corresponding audio files for some reason (or when an existing audio file is empty) are filtered out as well. In the result of filtering, 11'550 utterances overall, or 14.0% of all the data, have been excluded (see more in Table 8).

To enable the choice between one of the two possible transcription types — dialectal and normalised ones — the `prepare_Archimob_training_files.sh` and `process_archimob_csv.py` scripts have been also modified to include it as an additional option.

5.2.1.2 Synchronisation between csv and audio files

As the first version of the script had to deal with the interview-long audio files, it extracts audio chunks along with the preparation of the transcription file and assigns the same names to both audio files and the corresponding transcriptions. Now, we already have the chunked audio files, however, their original names (audio-id) do not fit well the format required for Kaldi. Kaldi sorts all the utterances by speaker and by number of an utterance, thus, it is important to have speaker information to be included in an utterance-id[52]. Another script (`sync_csv_wav.py`) was written to rename audio files according to the utterance-ids of their transcriptions. For example, *d1008_ T8.wav, d1008_ T9.wav, d1008_ T10.wav* files are renamed into *1008_ B-0008.wav, 1008_ B-0009.wav, 1008_ B-0010.wav* where '1008' is an id of an interview, 'B' is a speaker-id and the rest is the utterance number. Such post-synchronisation is possible, because the XML annotations contain information about the audio-transcription linking ('media_pointers' information from the *start* attribute for the audio file-id and *xml:id* attribute for the utterance-id; see Fig. 7).

Additionally, to avoid data discrepancy during the training, the renaming script verifies again the information about the missing elements — missing audio files for some transcriptions, missing annotations for some existing audio files (for example, noise at the beginning of the recordings or unclear speech) — and about the overlapping utterances.

[52]"...if you have speaker information in your setup, you should make the speaker-id a prefix of the utterance-id; this is important for reasons relating to the sorting of these files" (Kaldi documentation: `https://kaldi-asr.org/doc/data_prep.html` (12.03.2020)).

Therefore, the integration of the new ArchiMob data mostly consists of data preparation. It includes a) adapting the scripts to process the new input format, b) adding an option to choose the transcription type (with the dialectal transcriptions as the default one), c) cleaning the data from fragments unsuitable for training, d) fixing minor inaccuracies (e.g. an encoding problem with a transcription symbol that was not recognisable by Kaldi and had not appeared in the first release of the ArchiMob corpus).

5.2.2 Training/development/evaluation data split

After the data augmentation, the following step before proceeding with training models and experimenting is to define the data split. Due to the high dialectal variability of Swiss German, the proportion of different dialects in training and evaluation data sets can be crucial for the evaluation results. I have already mentioned above (5.1.5) that the first data splits were mostly based on the idea of having different or the same dialects in the training and evaluation sets, and the scores for the same system but evaluated on different dialects already differed a lot [Scherrer et al., 2019b]. At the same time, the splits were done on the interview-level, the interviews were not shared between the sets, which means speakers from the evaluation set never appeared in the training set.

After adding the data from the second ArchiMob release, we[53] have adapted a different approach for the data split: the data was split on the utterance-level and the utterances from all the interviews appeared in all three data sets. This kind of data split led to homogeneous distribution of different dialects and speakers between the training and evaluation sets and is expected to provide an averaged evaluation results. Moreover, the utterances selected from different interviews for the evaluation set had been earlier used in the previous study on part-of-speech evaluation [Samardzic et al., 2016]. There is an advantage of using this earlier established test set, since it contains linguistic annotation that can be useful for further analysis of the evaluation.

Besides keeping the balanced diversity of the training, development and evaluation data, the proportion of split is to be defined. While the conventional train-test split is usually 0.8/0.2 or 0.9/0.1 correspondingly, in order to reserve as much training data as possible, we have decided to use a smaller evaluation set. Our test set is only 2.3% of all the data, or 1'870 of 82'439 utterances.

[53] 'We' in this context refers to me and Tannon Kew, who worked with the ArchiMob corpus at the same time as I did [Kew, 2020].

For tuning the system and evaluating it at the development stage, a small development set was also formed from the training set. It consists of 2'266 utterances, or 2.7% of all the data, which were randomly sampled from all the interviews. In the previous section, I wrote that many utterances had to be filtered because of different criteria including anonymity, sound quality, overlapping, etc. In Table 8 below, the sizes of data sets before and after filtering are reported:

Table 8: Number of utterances in train/dev/test sets

SET	N of all utterances	N of utterances after filtering
TRAIN	78'303	67'693
DEV	2'266	1'710
TEST	1'870	1'486
all	82'439	70'889

Three .json formatted files keep separately all the ids for train, development and evaluation sets. The data split is done with the `split_data.py` script written by [Kew, 2020] that takes the .csv file with all the data utterances and .json file with set ids as arguments and creates three new .csv files with the selected utterances only.

5.3 Evaluating systems trained on the first VS the second releases of the ArchiMob data

The system was trained and evaluated on all the data from the second release of the ArchiMob corpus and with the new `train/dev/eval` split. The language model was retrained with the `simple_lm.sh` script with the same settings as in the first version (3-gram statistical model). In Table 9, the evaluation results for the *discriminative hybrid DNN* model (NNET-DISC) described in section 5.1 are given when: 1) trained on the first release and 2) trained on the second release. In both cases the models were trained on the *dialectal Dieth's transcriptions*:

Table 9: Evaluation of the baseline setup trained on the first release and on the second release of the ArchiMob corpus ('ins' — insertions, 'del' — deletions, 'sub' — substitutions)

Model	ArchiMob	Training data, utterances	WER, %	CER, %
NNET-DISC (baseline)	release 1	36'498	68.92	33.78
NNET-DISC	release 2	67'693	**55.58**	23.41
			7'004 / 12'601	12'480 / 53'307
			649 ins	3'358 ins
			1'340 del	4'000 del
			5'015 sub	5'122 sub

The system trained on the first release is taken as a **baseline**. Training on the full data from the second corpus release gained considerably higher scores compared to the baseline scores: 68.9% against 55.58%. In further experiments, all models are trained on the training set from the second corpus release.

5.3.1 Flexible WER evaluation

To score the performance of the systems, besides using the standard metrics of WER and CER, the soft evaluation measure called FlexWER was introduced by Tannon Kew in his master thesis about language modelling for Swiss German speech-to-text [Kew, 2020]. The general idea behind FlexWER is similar to the evaluation measure WERd proposed by Ali et al. [2017] for the Arabic dialects. In [Ali et al., 2017], a large collection of writing variants of words and phrases was gathered from Twitter with the context window of 2 and organised into a spelling variant table containing spelling variant pairs with variant frequencies and normalised edit distance. Then, mapping the hypothesis to the reference, spelling variants from the same pair and with the edit distance below the threshold are considered correct.

In our case, instead of collecting all potential spelling variants from side sources, the word-level mapping between the acoustic and the normalised transcriptions provided by the ArchiMob corpus was used (see also section 4.1.2). Since in our test set we were dealing with a closed vocabulary from the corpus, there was no need to introduce any additional metrics, such as a distance threshold, and a direct mapping

between two transcriptions was used. A word from the hypothesis transcription was considered correct if it referred to the same normalised word as the mapped reference word (see Table 4).

The models' performance measured with FlexWER measure is reported in Table 10. We can see that there is considerable difference between WER and FlexWER of approximately 20% for all the models. Due to its flexibility, FlexWER allows more realistic evaluation of a speech recognition model performance for Swiss German. In the plot 8 below, the distribution of errors of different types is shown. As expected, there is a noticeable drop in the number of substitution errors, when insertion error rate stays almost unchanged:

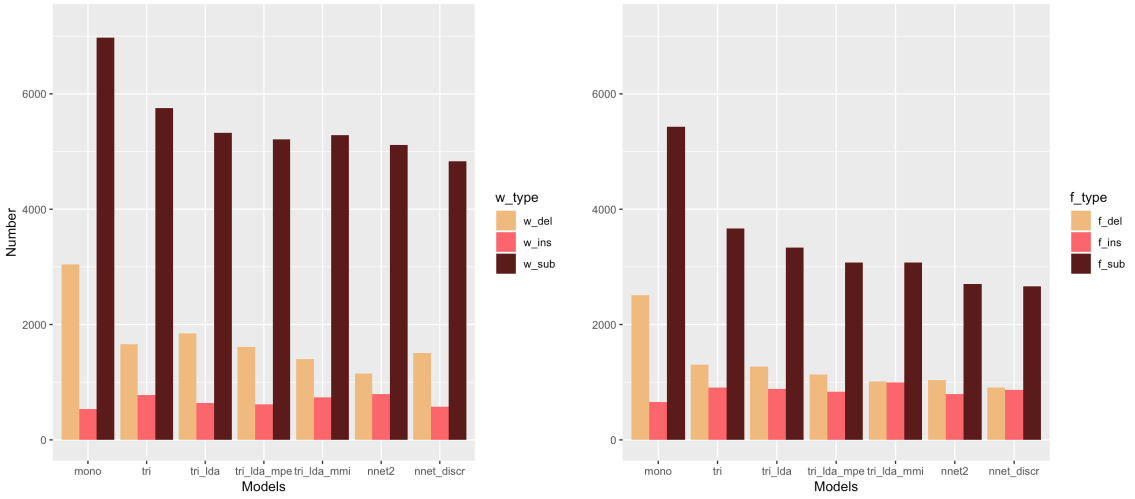

Figure 8: Distribution of insertions, deletions and substitutions for WER and FlexWER scoring.

5.3.2 Evaluation results for GMM and NN acoustic models

The setup used for the acoustic model is a DNN-HMM hybrid trained with the alignments built on the top of a sequence of GMM models (see 3.3). To understand better the improvement gained with different GMM models, besides the final NNET-DISC model, I evaluated the other 5 acoustic models in a pipeline. I expected better performance with the NN acoustic models [Wang et al., 2019]. Evaluating both GMM and NN-based models, however, allows 1) comparison of two approaches and measuring how big is the impact of different implemented methods on our data, moreover, 2) understanding how good the GMM model used for the final alignment was.

The scripts used for decoding in Kaldi differ depending on the feature processing needed for different models. The decoding script used in the original

pipeline is an interface for the Kaldi decoder for neural networks acoustic models (`steps/nnet2/decode.sh`). It is designed to process the features exactly in the same way as during the training of neural network models (here the NN with p-norm and with a discriminative sMBR criterion), and does not support decoding with GMM models. In order to evaluate the system with GMM acoustic models, I used a different decoder (`uzh/decode_gmm_wer_cer.sh` based on the `steps/decode.sh`). The results from different GMM and DNN models are presented in Table 10:

Table 10: Evaluation of the baseline acoustic models trained on the ArchiMob_r2

AM	WER, %	CER, %	FlexWER, %
GMM mono	83.77	49.64	64.61
GMM tri	69.16	35.53	47.24
GMM tri+lda	66.90	32.76	44.3
GMM tri+lda+MMI	62.37	29.26	39.58
NNET	57.06	24.02	34.53
NNET-DISC	**55.58**	**23.41**	**33.65**

As expected DNN-HMM models achieve the best WER scores. The best GMM-HMM model used for the final alignment is a discriminative MMI model that also coincides with the previous studies, e.g. by [Rath et al., 2013].

5.4 Summary

Increasing the amount of in-domain data from 25 to 60 hours brought 13.32% of absolute improvement to the baseline hybrid DNN-HMM model trained discriminatively. The best alignment model is a discriminative triphone GMM model with the MMI criterion. The FlexWER measure is used to provide a more real estimation of model performance, which error rate is around 20% better than the one of the standard WER measure.

6 Improving the acoustic model

Improvements in acoustic recognition accuracy can be achieved by a) applying advanced techniques to attain stronger features and more robust training (see Chapters 2, 3), b) tuning the acoustic model parameters, c) using different model architectures, and d) adding more training data or using pre-trained models. Some of the methods, like using more training data, are more general and would be beneficial for most of the cases: as it was already shown in the previous chapter (5.2). Other approaches, such as parameter tuning, are more data dependent and their outcomes are less predictable. The evaluation of different acoustic models of the baseline has illustrated well the contribution of some acoustic modelling techniques: e.g. using CD states, LDA feature transformation, discriminative training, or using DNN acoustic models (Table 10). In this chapter, I will present and discuss some further experiments with different techniques, parameters and architecture intended to improve the baseline results focusing on those that can be beneficial for Swiss German speech recogniser. Some modifications of the baseline demanded more departures from initial set up, e.g. including using the newest version of Kaldi.

The main challenges of the Swiss German data are discussed in Chapter 4, section 4.1, and two of them, data variability and lack of standardised spelling, are especially important for acoustic modelling. Both of them increase the data sparsity and as a result a model requires better generalisation. After having increased the amount of training data that was described in the previous chapter (section 5.2), in this chapter, I focus separately on the improvement of GMM and DNN acoustic models in regard of the challenges of the data comparing to the baseline model. In the first section, I will report the results of 1) the improvement of an alignment model, as an initial basis for learning acoustic-state relations, and 2) adding adaptation and generalisation techniques, in order to make the model more robust to the inter-dialect and inter-speaker diversity. For the alignment case, I will, first, test different discriminative criteria for the GMM-HMM model, and, second, the impact of different number of senones and Gaussians, as well as their proportion, on the accuracy of states recognition. Then, I describe the results of the models with the adaptive training (SAT) and with the improved generalisation using subspace GMM

models (SGMM).

The acoustic modelling is a sequential problem, which means that the current output (e.g. a state of a phoneme) depends on the previous input (an acoustic feature vector) and the length of the input (a sequence of speech frames, or feature vectors) is not fixed. For sequential modelling, training with the architecture that catches long-term dependencies can gain some improvement. Since the standard feed-forward NN are not usually optimal to learn long relations between the elements, in the second section, I describe the results of the time-delay neural networks (TDNN) architecture with the iVector features.

6.1 GMM level improvements

The baseline system has a basic pipeline of acoustic models that consists of four GMM-HMM models to provide an alignment and two DNN models on top of this alignment (section 3.4.1). The GMM models include a CI monophone model trained from a flat start, three CD models: with delta features, with LDA-MLLT training and discriminatively trained with the MMI criterion (see more in section 5.1.3). At the same time, for their training the default parameters only were used and no adaptation techniques were applied either on feature or on training levels. I assume that the improvement of a GMM model, which alignment is used for training a DNN, can help to improve the performance of the final DNN-HMM as well. Therefore, as the first step, I have tested additional techniques and different parameters for the GMM training. In this section, I will speak, first, 1) about training with different discriminative criteria; then, 2) about increasing the number of senones and Gaussians; and, finally, 3) about adaptive training using speaker-adaptive features and subspace training. All the results will be reported and discussed together in the last part of the section (6.1.4).

6.1.1 Discriminative criteria

In the baseline model, sequence-discriminative training helped to considerably improve results of acoustic models comparing to generative training. Sequence-discriminative training does not classify individual frames independently optimising training parameters to an 'abstract' loss function but instead uses optimisation towards the goals of speech recognition, such as maximum mutual information (MMI), minimum phone error (MPE), state-level minimum Bayes risk (sMBR) (see more in 3.4.2). Since in the baseline, only the MMI criterion has been used, in addition,

I decided to train models towards the MPE and sMBR criteria as well.

According to the previous research, the difference between three criteria is typically not big but depends on the data [Povey, 2005]. The setup of all models is very similar: all models are CD triphone ones (3.2) and use the same LDA features (2.3); the alignment is build with the GMM-tri+LDA-MLLT model (3.4.2), as well as the denominator lattices necessary for the discriminative training. The models differ only in the training criterion. In Table 11, the results for three models are shown.

6.1.2 Parameters tuning: number of senones and Gaussians

I assume that increasing the number of senones and Gaussians used in the baseline model, i.e. 2k senones and 10k Gaussians, can improve acoustic modelling providing training with more detailed information about the acoustic signal. Senones are clusters of triphone HMM states, which are used to optimise training and to avoid using all the possible combinations of triphones (section 3.2). Increasing the number of senones increases the complexity of a model and until some point improves the accuracy of acoustic modelling. There are not many studies investigating the most optimal number of senones but in some works, it was shown that the accuracy can drop again after a certain point [Hwang and Huang, 1992, Batista et al., 2018]. Thus, the number can be very different and depends on the data and on the language. In GMM-based acoustic models, it is a hyperparameter that can be tuned for each particular setup, and in different studies, it differs from around 1k [Yu et al., 2013] to 20k [Hinton et al., 2012] but typically stays within 10k [Povey et al., 2010; Gaida et al., 2014; Miao et al., 2015; Wang et al., 2018].

Senone parameter is also present in DNN training algorithms. Since DNN-based models work as classifiers, their goal is to find the best senone label given an acoustic input and the number of senones in the case of DNN is defined by the size of the output layer. In [Xiong et al., 2018], the authors experiment with the sets of 9k and 27k senones. There is, however, almost no difference between two sets, with only minor advantage from 9k-set. The possible reason for no improvement can be that 9k is already a large set of senones providing sufficient information for successful modelling.

Another hyperparameter that is closely related to the number of senones and characterises the GMM-based models only is a number of Gaussians. A GMM is composed of Gaussian mixtures (or Gaussian functions) and a number of them can be understood as a number of clusters per model (3.2). With different numbers of mixtures, therefore, a model can be more or less general. In acoustic modelling, the number

of Gaussians is usually varied from about 15 [Povey et al., 2010; Abushariah et al., 2012] to about 60 [Ali et al., 2014] mixtures per senone (or GMM) where the complexity of an acoustic model (number of calculations) is growing up with a higher number of Gaussians.

In the baseline setup for our system (Chapter 5), the default number of senones was 2k with the number of Gaussians equals 10k that means approximately 5 mixtures per senone. With the relatively big set of phones in the ArchiMob corpus — 56 phonemes and 224 states correspondingly — 2k senones cover only a very limited number of the existing CD states missing the information available from the acoustic contexts. At the same time, taking into account the high acoustic variability between the dialects, 5 Gaussians per state can be not sufficient either to catch all the important acoustic patterns for a given state (see Fig. 3). Comparing to the other studies mentioned in the two previous paragraphs, the values of both parameters seem to be rather small as well. I hypothesise that increasing the number of these two parameters can help to increase the sensitivity of acoustic mapping and will improve the quality of the alignment.

As the default baseline configuration was 2k-10k, first, I doubled the values of both parameters up to 4k senones and 20k Gaussians. The same pipeline of GMM models (as in Table 10) was trained. With the new configuration of 4k-20k, the proportion between the number of senones and Gaussians stayed inchanged, when there are 5 Gaussians per senone. In order to learn more about the impact of number of Gaussians per state, in the second round of experiments, 4k senones and 40k Gaussians were used, it means 10 Gaussians per state. Therefore, models with the following configurations have been trained:

- 2k senones, 20k Gaussians, 5 Gaussians per senone;
- 4k senones, 20k Gaussians, 5 Gaussians per senone;
- 4k senones, 40k Gaussians, 10 Gaussians per senone.

Comparing configurations of different Kaldi recipes, we can see that different GMM models can have different configurations: for example, the number of senones is always set higher for speaker adaptive training, the proportion of Gaussians per state is typically higher for models with the discriminative training. It can be explained by the fact that more complicated architectures can model better the variability of acoustically close sounds and, therefore, they benefit more from the larger number of senones and Gaussians. In our case, due to the time constraint, the experiments were limited by two combinations only and the number of parameters stayed the same for training all the GMM triphone models in the pipeline. All the results are

presented in the results section below in Table 11.

6.1.3 Improving generalisation

Besides the standard Cepstral Mean and Variance Normalization (CMVN) and LDA feature transformation (see 2.3), the baseline pipeline does not include any additional techniques to improve the generalisation of the model. Yet, adaptation can help to improve model performance reducing the effect of data variability. An overview of different GMM and DNN adaptation techniques is given above (see section 3.5). In order to improve the performance with adaptation, I have tried Speaker Adaptive Training (SAT) with fMLLR-features. In addition, I trained a model with the Subspace Gaussian Mixture Models (SGMM) approach that can boost the generalisation ability of a model (see 3.5.4). My first assumption was that speaker-specific features would help to smooth inter-speaker variability, including dialect difference between speakers (discussed in 3.5.3). The second assumption was that the subspace GMM training would allow additional smoothing on a higher level, including noise, speaker and dialect variability.

Speaker adaptive training (SAT). One of the ways to apply adaptation is using speaker-specific features, for example, fMLLR (see section 2.3). SAT model is trained with fMLLR-adapted features on top of either delta or LDA+MLLT features. This type of acoustic models usually achieves good scores, generalising better to unseen speakers at the evaluation stage [Leggetter and Woodland, 1994; Gales, 1998].

Subspace Gaussian Mixture Models (SGMM). The SGMM models are discussed in section 3.5.4. In the SGMM framework, a separate GMM is independently trained for each HMM state. Then, the HMM states share the same structure and vary only in their parameters, the means and mixture weights. The subspace approach is especially recommended when only little data is available and considered good for normalisation due to the full parameter space shared between the states [Povey et al., 2011]. Thus, the SGMM training has been tested on our data as well.

One of the central components in the SGMM training is the Universal Background Model (UBM) introduced in the context of iVectors (see 3.5.3). The UBM is separately trained as a pre-step and used after to initialise SGMM for all the states from the common start. The basic Kaldi setup was used with 5k senones and 8k Gaussians and speaker-specific fMLLR features. Like the standard GMM models, SGMM models can be according to the generative or sequence-discriminative approach. Therefore, two SGMM models were trained: a) with EM training algorithm

and b) discriminatively trained towards the MMI criterion.

6.1.4 Results

In this section, I summarise the results achieved by the different GMM models and their configurations described above and report the scores in Table 11 and Fig. 9. All the scores here are the result of evaluation on the development set.

The assumption about the benefit from the increase of senone and Gaussian parameter values has been confirmed (Fig. 9). It is especially noticeable with GMM-models that directly depend on the number of mixtures per model when the increase in the proportion of Gaussians per senone gained even better scores. For all five triphone-based GMM models (*tri*, *tri_lda*, *tri_mpe*, *tri_smbr*, *tri_mmi*), increasing the number of senones gained the average absolute improvement of 2.49% of WER and 1.64% of CER. The further increase of the number of Gaussians per state led to higher scores with the average improvement of 4.27% of WER and 3.05% of CER comparing to the models with the default configuration. In Fig. 9, only plots for WER and CER scores are included, because the relative improvement for the FlexWER scores has a similar trend as for WER: the average absolute improvement of FlexWER is 3.37%.

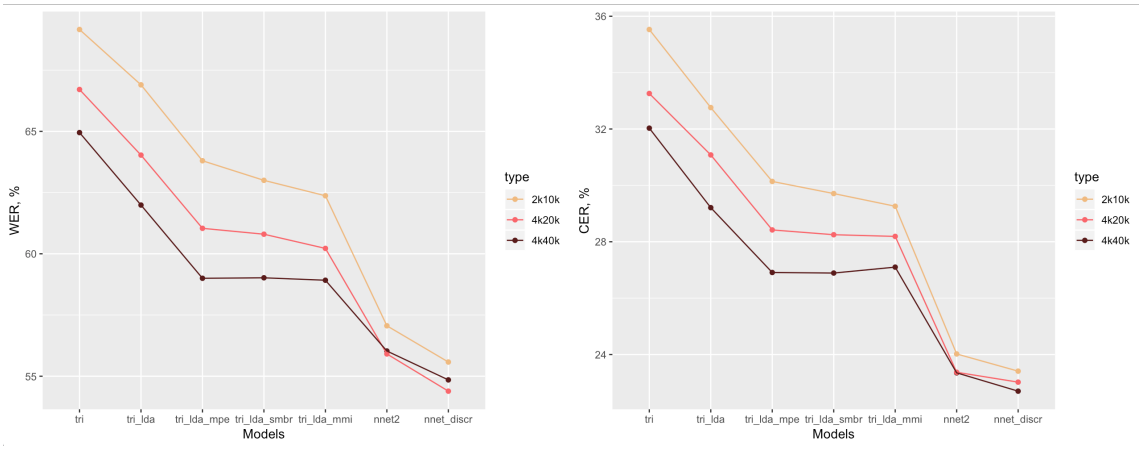

Figure 9: Models with different numbers of senones and Gaussians.

Some minor improvement was also registered for DNN-based models, which benefit from the improved alignment from the *tri_mmi* model. The best score for the NNET2 model is 55.91% of WER with the settings 4k-20k and absolute improvement of 1.15%. For the CER measure, there is no difference between 4k-20k and 4k-40k values and improvement to 2k-10k is 0.67% of CER. The discriminative NN model got slightly better results with 4k-20k for WER measure and with 4k-40k for

CER: with the absolute improvement of 1.19% of WER and 0.71% of CER. Further experiments can include a more flexible approach using different parameter values for different acoustic models. Moreover, controlling the size of the output layer can be also tried for DNN models.

The experiment with different discriminative criteria has demonstrated that using different criteria does not have a big impact on the improvement to the baseline model results. Compared to the performance of discriminative models with different training criteria, the model trained towards MMI slightly outperforms the other models optimised to MEP and sMBR and, hence, will be used for building the alignment (Table 11). Moreover, in Fig. 9, it is shown that the difference between the discriminative models with different training criteria reduces when the number of senones and Gaussians is increasing: with the 4k-40k configuration all three models perform with almost the same WER.

The assumption about the fMLLR speaker-specific features has not been confirmed, as on our data the SAT model trained with the fMLLR features attained low results and outperformed only the CI monophone GMM model (see Table 11). It might have happened because of only a small number of speakers, which have enough training data: 43 speakers. Interviewers and 25 other random participants had mostly only few short replicas, which are not enough for computing fMLLR. The second possible explanation can be the fact that all main speakers are presented in the training, as well as in the development and evaluation data sets; then, in such a case, the speaker adaptation did not play an important role.

The assumption that subspace training (SGMM) can be helpful with little and variable data is partially confirmed. The SGMM models achieved better results than the baseline GMM discriminative models with 2k10k: 60.65% against 62.37% of WER. At the same time, the GMM discriminative model with 4k40k still outperforms SGMM. To avoid the effect of parameters, the SGMM models were additionally trained with 4k senones and 40k Gaussians. The increase of parameter values this time did not achieve any gains. Thus, the SGMM models are less sensitive to the number of senones and Gaussians. The results for both fmllr-SAT and SGMM experiments are presented in Table 11 and compared with the results of the CD GMM-lda models trained with 2k10k and 4k40k configurations.

Table 11: Evaluation of systems with different GMM models (compared to the DNN-HMM systems; development set)

AM	WER, %	CER, %	FlexWER, %
GMM+MPE 2k10k	63.80	30.14	41.43
GMM+sMBR 2k10k	63	29.71	40.81
GMM+MMI 2k10k	62.37	29.26	39.58
GMM fmllr+SAT 2k10k	70.52	35.78	47.65
NNET fmllr+SAT 2k10k	65.92	30.85	43.24
NNET-DISC fmllr+SAT 2k10k	62.47	28.43	39.98
GMM-lda 2k10k	66.90	32.76	44.3
GMM+MMI 2k10k	62.37	29.26	39.58
GMM-lda 4k40k	61.99	29.21	40.63
GMM+MMI 4k40k	58.92	27.1	37.27
NNET-DISC 2k10k	55.58	23.41	33.65
NNET-DISC 4k40k	54.85	22.7	32.87
SGMM 5k8k	64.18	28.43	41.86
SGMM+MMI 5k8k	60.65	27.59	37.17
SGMM 4k40k	63.62	29.27	39.78
SGMM+MMI 4k40k	60.8	27.4	37.21

Besides SAT and SGMM, I have tested two other widely used approaches for data adaptation and better generalisation. In the first case, it is using iVectors, which were initially introduced to catch speaker characteristics in speaker recognition task (section 3.5.3). In the second case, it is data perturbation that makes the model more robust adding noise to the data. Another benefit from data perturbation is that it increases the amount of the data. These two techniques, however, I have used in combination with a different NN architecture: time-delay neural networks (TDNN). As the TDNN architecture is beneficial for long-term dependencies modelling, I will talk about it separately in the following section.

6.2 Improvements for NN

The experiments with GMM models in the previous section demonstrated that if some additional techniques lead to an improvement, this improvement is not very

big, and its contribution to the score of the final DNN model is even less noticeable. Thus, the more essential improvement is expected from the side of DNN architecture. In this section, I will present the results of the time-delay neural networks (TDNN) architectures trained with the iVector features and data augmentation, as at the moment, this is one of the most successful acoustic models for the hybrid speech recognition systems.

6.2.1 Modelling long-term temporal dependencies

Two DNN-HMM models trained in the baseline pipeline are feed-forward network architectures (see section 3.3). The first one was trained with the p-norm activation function [Zhang et al., 2014] and the second DNN was optimised with discriminative training towards the sMBR criterion. Both of them achieved significant improvement over the GMM models. Standard feed-forward DNN architectures, however, are not optimal for modelling long-term temporal dependencies between the acoustic events.

In the ArchiMob corpus, the average length of a utterance is 7.28 words and 31.24 characters (excluding silences and noise). There are two ways to improve the acoustic training of long sequences. One approach is to use feature representations that catch long-term spectro-temporal dynamics of the signal and can be used in combination with the standard feed-forward DNN. Such features, for example, are TRAPs [Hermansky and Sharma, 1999], wavelet based multi-scale spectro-temporal representations [Mesgarani et al., 2004] and deep scattering spectra [Andén and Mallat, 2014] (cited from [Peddinti et al., 2015]). Another way is to improve modelling on the training side addressing such architectures as a) recurrent neural networks (RNN), for example, long-term short-term memory networks (LSTM), or b) adapted feed-forward DNN, like time-delay neural networks (TDNN).

The disadvantage of RNN is a higher training time demanded by the sequential nature of the learning algorithm. The TDNN architecture introduced by Waibel et al. [1989] and Waibel [1989], while modelling long-term temporal dependencies, allows parallelisation and needs much less computation time comparing to standard RNN. Thus, to improve modelling of long-term temporal dependencies in the acoustic signal, I decided to test the TDNN architecture for the acoustic model training on our data.

TDNN has a modular and incremental design that allows the formation of nonlinear decision surfaces: “the time-delay arrangement enables the network to discover acoustic-phonetic features and the temporal relationships between them independent of position in time and hence not blurred by temporal shifts in the input”

[Waibel et al., 1989, p. 328]. At the lower layer, local short duration features are formed learning on narrow contexts. Higher layers are able to attend to larger time spans and learn wider temporal relationships (Fig. 10). Hence, each layer in the architecture operates at a different temporal resolution, which increases from lower to higher layers and creates large networks from sub-components. Peddinti et al. [2015] proposed a sub-sampling technique for TDNN implemented in Kaldi that helps to further reduce the computational cost, as hidden activations are computed at each level at only few time steps. In Fig. 10, the TDNN architecture with and without sub-sampling is shown. With sub-sampling approach, frames in the hidden layers are spliced with gaps between them: for example, the input can be spliced at the current frame minus 7 and the current frame plus 2:

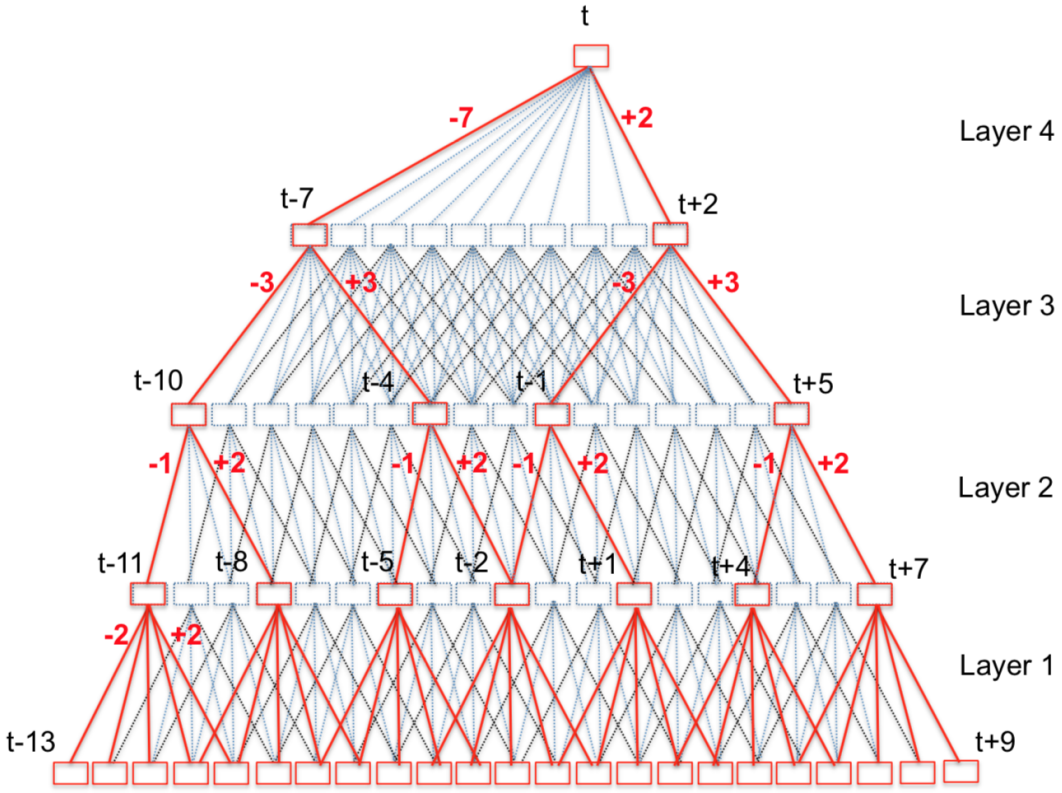

Figure 10: TDNN architecture with sub-sampling (red) and without sub-sampling (blue+red)[Peddinti et al., 2015].

As TDNN model long-term temporal dependencies from short-term speech features, MFCC features from the previous experiments are also suitable for the TDNN scenario. For speaker and environment adaptation, in the Kaldi TDNN implementation, iVector modelling is used, which has shown its posivite impact to instantaneous and discriminative adaptation of the neural networks [Karafiát et al., 2011] and has already become a state-of-the-art technique in ASR architectures [Verma and Das,

2015; Xie et al., 2019].

6.2.2 iVector adaptation

The iVector adaptation is introduced in the section 3.5.3.

Speaker adaptation. Originally, the iVector technique was proposed for the speaker verification task, as it extracts speaker-specific features comparing an input feature vector to the supervector of the UBM [Dehak et al., 2010]. iVectors are usually extracted at the speaker level to represent each speaker's acoustic characteristics. Therefore, along with a set of acoustic features used for phonetic discrimination and a set of speaker relevant features is created which is used to scale the first set predictions and to reduce inter-speaker variability.

The iVector features are often used in ASR setups [Saon et al., 2013; Miao et al., 2015] and, despite being referred to speaker adaptation, iVectors representations also encapsulate other types of variability (noise, channel, etc.) that helps to improve speech recognition robustness.

Accent adaptation. Another speech processing task where iVectors are applied is accent recognition [Chen et al., 2010; DeMarco and Cox, 2012; Bahari et al., 2013; Behravan et al., 2013]. Feature variations within the same language but across the dialects and accents are to be caught. The potential accent adaptation is also considered useful for our speech recognition case, as many Swiss dialects are present in the data set. At the same time, since each speaker speaks exactly one particular dialect, speaker adaptation already covers accent adaptation.

In the ArchiMob corpus, there are 43 main speakers and 14 dialects that overlap in the unequal proportion: most of the speakers have Zürich dialect followed by Aargau, Bern, Luzern and Basel-Stadt dialects (see Table 3). All other dialects are represented with one or two speakers only. On the one hand, in general, training the TDNN model with two different iVector focuses, speaker and accent adaptations, can give different results. On the other hand, I assume that with the ArchiMob data the difference will be not noticeable because both speaker and accent oriented iVectors would contain similar acoustic features and, thus, have a high level of mutual information. Hence, only speaker-specific iVectors were used.

For the speaker recognition purpose, the target iVector model is typically trained in an unsupervised way. The model learns to generalise from the data, does not need labelled data but can be sensitive to the amount of data. The data amount issue can be related to the whole TDNN training. One possible solution to increase

the amount of training data is creating the simulated data, for example, with data perturbation.

6.2.3 Simulated data augmentation

Data perturbation is an audio-level speech augmentation method that produces copies of the original acoustic data through the acoustic modification of the signal [Ko et al., 2015]. The implementation of this technique is computationally cheap but the increased amount of data typically brings improvement especially on small training sets. There are different types of data perturbation including rate, speed, volume, and vocal tract length (VTLP) perturbations [Jaitly and Hinton, 2013]. Performing speed perturbation of the original data is usually done with warping factors between 0.9 and 1.1 where 1.0 means no change, >1.0 is an increase and <1.0 is a decrease of the signal speed.

Volume perturbation of the training data has a positive effect when iVector is included. In order to improve learning the iVector normalization, it emulates mean shifts in the MFCC domain adding more variability to the variance in audio volume. With better normalisation, volume perturbation allows making trained neural networks more invariant to the test data volume.

Applying data augmentation techniques such as speed and volume perturbation helps to increase the size of the training data to three times the original size. In the scenario with the DNN and iVector training, this data augmentation is especially important.

6.2.4 Training TDNN

For training TDNN acoustic model, the latest Kaldi[54] implementation from the *wsj* recipe was used: `wsj/s5/local/chain/run_tdnn.sh`. In this implementation, the TDNN are trained with data perturbation and the iVector features. The diagonal UBM with 512 components and the iVector extractor are trained with: `steps/online/nnet2/train_diag_ubm.sh` and `steps/online/nnet2/train_ivector_extractor.sh` bash pipelines. Training on CPU takes around 1.5 weeks; training with GPU allows considerable acceleration and takes around 15 hours.

On each frame, a 100-dimensional iVector is appended to the 40-dimensional MFCC

[54]Kaldi 5.5., 20.03.20.

input. Speed- and volume-perturbations were done on the training data prior to extracting the main training features:
`utils/data/perturb_data_dir_speed_3way.sh`,
`utils/data/perturb_data_dir_volume.sh`.
For the speed perturbation, the speed function of SoX toolkit[55] was used to resample the signal. As a result, two copies of the original data were created with 90% and 110% of the original rate. After the speed modification, the length of the signal is changed correspondingly; because of this, the alignment of the acoustic signal to transcription was repeated.

6.2.5 Results

The systems were evaluated on the development set. TDNN achieved considerably better results comparing to the discriminatively trained NN model according to all three evaluation metrics (Table 12). The main improvement was gained due to the decrease in the deletion errors.

Table 12: Evaluation of the system with the TDNN acoustic model in comparison with the baseline model (development set)

AM	WER, %	CER, %	FlexWER, %
NNET-DISC-2k10k (baseline)	55.58	23.41	33.65
NNET-DISC-4k40k	54.85	22.7	32.87
TDNN-iVector-4k40k (speaker adaptation)	43.18	15.23	23.3

Modelling long-term temporal dependencies in combination with the speaker adaptation iVector features and simulated data augmentation helped to achieve considerable improvement over the previous best results, with the absolute improvement of 11.67% WER. The time constraints and highly time-consuming training do not allow further experimenting in order to understand the contribution of each method separately from the others. Nevertheless, future experiments can involve using more complicated advanced architectures, e.g. with projected long short-term memory networks (LSTMP) in the TDNN architecture [Cheng et al., 2017].

[55] `http://sox.sourceforge.net/`.

6.2.6 Summary

In this chapter, I have described the results of the experiments on the improvement of the baseline framework. The following methods and techniques have been tested: 1) different discriminative criteria and different numbers of senones and Gaussians to achieve better GMM-based alignment; 2) speaker-specific features, including fMLLR and iVectors, and subspace GMM training to improve the generalisation ability of the model; 3) TDNN architecture in order to model long-term dependencies in a sequence of the acoustic elements; finally, 4) data augmentation with data perturbation.

The models with the increased number of senones and Gaussians, the subspace GMM training, and the TDNN architecture in combination with iVectors and data perturbation outperformed the baseline framework. The best results have been achieved with the TDNN model.

7 Performance analysis

All the evaluations presented in Chapters 5 and 6 are done on the development set. In the last chapter, I come back to the best models to evaluate them on both in-domain and out-of-domain test sets. Such a testing approach allows more independent evaluation of model performance and estimates better its ability to generalise. Later in the chapter, I analyse the distribution of error types, as well as the recognition of different phonemes and different dialects. The analysis of the recognition output provides a useful insight into the model performance and allows a qualitative comparison between the models.

7.1 In-domain and out-of-domain data evaluation

In section 4.2.2, the content and the particular characteristics of the ArchiMob corpus are discussed. The corpus covers 14 Swiss German dialects but, at the same time, it has a few general limitations: a relatively small number of speakers, predominance of a specific age group, limited number of topics presented in the interviews. Therefore, evaluation of the models on the test set sampled from the ArchiMob corpus can give an only one-sided estimation of the performance. We will call it in-domain evaluation with the in-domain test set. I assume that the in-domain evaluation returns considerably better scoring results than the out-of-domain one. The second assumption is that better models are more robust with different type of data and the difference between their performance on in- and out-of-domain data is smaller.

To learn more about the performance of the models outside the same type of data, an additional test set for out-of-domain evaluation was used. For this purpose, I took the Schawinski Transcripts corpus (3 hours 10 minutes) introduced in section 4.2. The choice was due to the convenience, as the Schawinski Transcripts corpus has been created in accordance with the same guidelines as the ArchiMob corpus and was transcribed following the same Dieth oriented principles. Along with that, the Schawinski audio had been recorded under quite different conditions: at the televi-

sion studio, often with music at the background, and with frequently overlapping replicas.

The preprocessing of the Schawinski data was similar to the ArchiMob data preparation (see section 5.2.1.1). The Schawinksi transcriptions are in the EXMARaLDA format (EXB) and they were preprocessed together with the long-interview audio files with the `process_exmaralda_xml.py` script with the 'EXB' input format option. The script outputs the chunked audio files and the .csv transcription file, labels of which enable further filtering with the `process_archimob_csv.py`. In the result of filtering, 2'525 utterances of the initial 4'836 chunks were kept (see Table 13).

Table 13: Statistics for the in-domian and out-of-domain test sets

TEST SET	N of all utterances	N of utterances after filtering
ArchiMob	1'870	1'486
Schawinski	4'836	2'525

Table 14 shows the evaluation results on both test sets. The models chosen for the test evaluation are: 1) the baseline model; 2) the discriminatively trained NN with the increased number of senones and Gaussians; and 3) the TDNN with iVectors and data perturbation. The scores achieved on the in-domain set are pretty much the same, like the ones on the development set. As expected, the results of the out-of-domain evaluation are noticeably worse comparing to the in-domain with the absolute difference of 13.5% of WER and 16.5% of FlexWER on average.

Table 14: Evaluation of the models on the in-domian and out-of-domain test sets

Model	Test ArchiMob		Test Schawinki	
	WER, %	FlexWER, %	WER, %	FlexWER, %
NNET-DISC-2k10k (baseline)	55.18	32.59	68.76	49.42
NNET-DISC-4k40k	54.39	32.09	68.03	48.59
TDNN-iVector-4k40k	42.38	**21.53**	59.81	**37.83**

The hypothesis about the better generalisation of better models has not been confirmed. The TDNN model performed 17.43% of WER better on the ArchiMob than on the Schawinski test set. Hence, the difference between the in-domain and out-of-

domain evaluation of the TDNN model is even a bit bigger. Nevertheless, it is still the best result achieved among all the other models.

7.2 Analysis of recognition results

Sometimes, better understanding of the results helps to choose the better methods for further improvement. Analysis of the recognition output can provide useful insights into the nature of the data. Thus, I decided to address two important aspects of the recognition output: a) the difference in the recognition of different phonemes, and b) the difference in the per dialect evaluations.

7.2.1 Phoneme recognition

Some phonemes can be recognised better than others. It happens because of 1) their higher frequency in the data and 2) depending on their acoustic characteristics. For example, as discussed earlier (section 4.1.1), the highest variability on the phonetic level is observed in vowels. Hence, the highest variation in the recognition of vowels is expected. The quality of consonants is acoustically easier to define and, then, the consistency in transcribing consonants might be higher comparing to vowels leading to their better recognition. Obstruents (like plosives and sibilants) are expected to be recognised easier than sonorant consonants (e.g. /j/), as the latest are closer in their nature to vowels and, therefore, are more prone to modifications and elision. Since the goal of the acoustic modelling is to find the best matching between the acoustic signal and the phonemes (or states), the insight into the distribution of error types and/or recognition of different phonemes can bring a better understanding of its performance.

Fig. 11 illustrates the distribution of errors for vowels and consonants for the NNET-DISC-4k40k model with the in-domain evaluation. Vowels are presented in the two upper plots and consonants are in the two lower plots. The two left plots show the absolute frequency of occurrence of phonemes in the evaluation set and the absolute number of errors for each phoneme; the right plots show the proportion of error types (correct, deletion, insertion, substitution — highlighted with different colours) for different phonemes.

The assumption about better recognition of more frequent phonemes is confirmed for vowels. More than half of the vowels are very infrequent. Such vowels are recognised correctly only in 50% or less percent of the cases; the two most infrequent vowels

(/ã/, /à/) have been substituted with a different vowel in 100% of the cases.

The set of vowels of the ArchiMob transcriptions is very rich: 18 vowels comparing, for example, to 11 vowel phonemes in Zürich dialect proposed in [Fleischer and Schmid, 2006] (see Fig. 13 in Appendix). At the same time, the most common type of errors is substitution. One of the possible ways to reduce the number of substitutions could be reducing a number of vowels in the transcription vowel set consistently replacing the infrequent vowels with the qualitatively closest analogue. Yet, further experiments are needed to check this assumption.

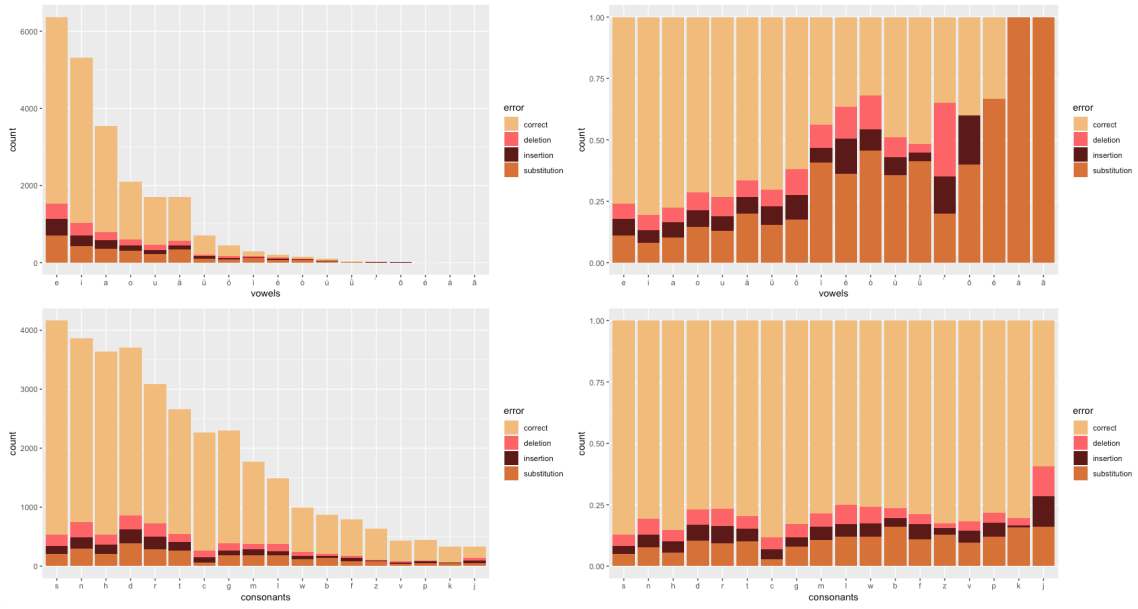

Figure 11: Distribution of errors for vowels (two upper plots) and consonants (two lower plots) for the NNET-DISC-4k40k model evaluated on the in-domain data.

Consonants are less error-prone than vowels, they are more homogeneous in recognition and the frequency of occurrence (Fig. 11; two lower plots). An exception among the consonants, as it was predicted, is a sonorant /j/, which has almost double more recognition errors.

The results from the NNET-DISC-4k40k model can be compared to the output from the TDNN-iVector model that gained the best evaluation scores among other models: see Fig. 14 in Appendix. In the case of TDNN, the overall results are noticeably better for both vowels and consonants but the proportion of errors for /j/ is still essentially higher than of all other consonants.

7.2.2 Per dialect evaluation

As already discussed in 5.1.5, Table 7, the evaluation results can differ a lot when tested on different dialects depending on how much a dialect deviates from the majority of the training data. Thus, the hypothesis is that the recognition error rate among the dialects can depend on a) the proportion of a dialect in the training set (refer to Table 3) and b) the level of variation of a dialect from the rest of the dialects. The WER scores for different dialects achieved with the NNET-DISC-4k40k and TDNN-iVector models are showed in Fig. 12:

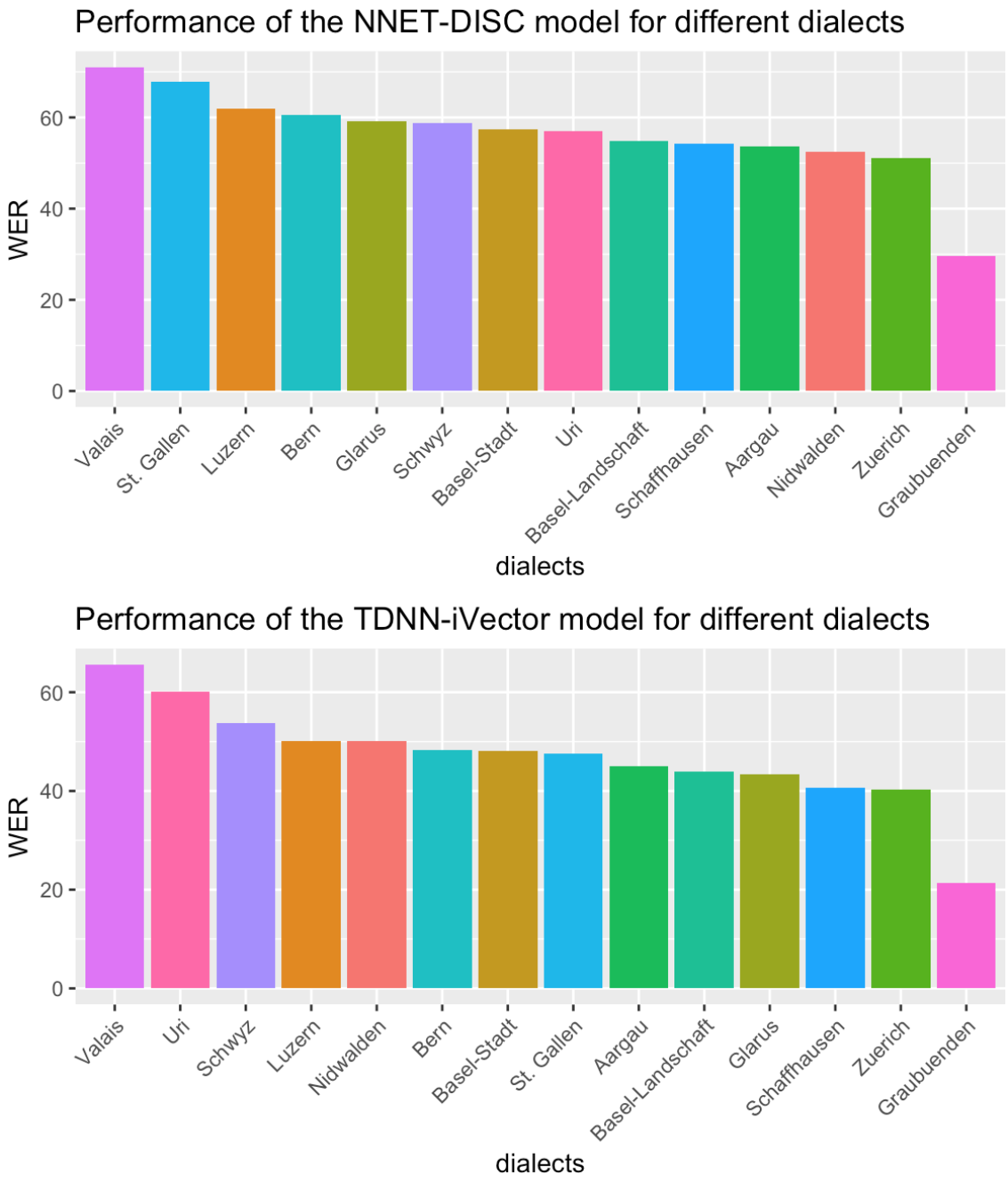

Figure 12: WER for different dialects with in-domain evaluation.

The first assumption regarding the effect of the proportion of a corresponding dialect in the training data does not seem to be supported: Graubünden dialect has the best results but is represented in the ArchiMob corpus with only one interview. Basel-Landschaft and Schaffhausen dialects also have only one interview in the corpus per dialect but show lower error rate scores.

The second assumption is about the similarity between different dialects and demands further investigation. Some tendencies, however, can be noticed from the plots above: the system performs considerably worse when evaluated on Valais dialect comparing to the evaluation on the other dialects. On the other hand, evaluation on the data from Graubünden dialect is consistently good outperforming the results gained with Zürich dialect, which is the largest part of the training data (12 interviews).

In the context of dialect recognition, it is important to mention that since many dialects have only one interview in the ArchiMob corpus, this means that they are also represented by only one speaker. Thus, recognition of a dialect, in this case, is very speaker-specific. For getting a more reliable picture of dialects recognisability with a unified ASR system, more speakers are needed per dialect.

7.3 Discussion

In this thesis, the first multi-dialect ASR for Swiss German was proposed and evaluated on in-domain and out-of-domain data. One of the main challenges for the multi-dialect ASR is the variability of the data that, besides being inter-dialectal, can also be inter-speaker and inter-domain. The ArchiMob corpus used as the training data consists of audio interviews in 14 regional varieties, and, at the same time, is specific in regard to speakers and topics of the recordings. All these factors have a large impact on the evaluation results that can be very different depending on the test data.

Focusing on the acoustic modelling aspect in order to diminish the effect of variability and to improve the system's ability for generalisation, different techniques and methods have been tested. Some of them have helped to outperform the baseline results. However, the best way towards improvement is by using more advanced neural networks architectures, which prove to be able to learn and generalise better. These results are consistent with the previous studies where considerable progress within the last few years is observed due to the use of DNN (compare the results from [Ali et al., 2014] to [Ali et al., 2016] in section 4.3.1 where the use of the new

DNN architectures in combination with more training data led to the considerable decrease of WER from 31.72% to 14.7%[56]). At the same time, DNN are sensitive to the amount of training data, which was another challenge in the present project.

My work has a number of important limitations due to the time, space and data constraints. A general limitation of my work is that the data used for training is taken from a relatively closed domain (narratives on a similar topic). It makes the system perform considerably worse on the out-of-domain data. The result, however, is predictable taking into account the size and representativeness of the corpus, as well as the number and the average age of speakers and the specificity of the topics covered in the interviews. Yet, this data limitation is mainly caused by the fact that only a very limited amount of transcribed data is available for Swiss German.

Another essential limitation is that due to the time constraint, I have tried only three NN acoustic models when there are few other strong architectures for acoustic modelling that could perform better: e.g. LSTM or projected long short-term memory networks (LSTMP) in combination with the TDNN architecture. End-to-end modelling was out of the scope in the thesis (partially because of the data constraint) but it can also potentially improve the current best system.

The further performance gains of the system can be achieved using a stronger language model; the language modelling component was left beyond the scope of the present work. The experiments aimed to improve the language model for Swiss German in the models trained on the ArchiMob corpus are described in a recent thesis by Tannon Kew [Kew, 2020].

Besides language modelling, there is still a wide field for research in Swiss German ASR. The next important steps in the future can be:

- More detail evaluation of the performance for different dialects in order to understand better the similarity between the dialects. This can be used for better data balancing when new data are added or generated, or for the improvement of the pronunciation lexicon.
- Extension of the spoken corpus to contain more topics, speakers, age groups covered.
- End-to-end modelling when more data is available, which is currently a quickly developing and promising ASR approach with many methods to increase the performance.

[56]In the two studies, different evaluation sets are used, which makes their comparison not fair but illustrates well the general tendency.

8 Conclusion

In the thesis, I have addressed a non-trivial ASR question of multi-dialect large vocabulary continuous speech recognition with a particular focus on the acoustic modelling. The goal of the project was to propose the first ASR system with a unified acoustic model for Swiss German dialects from 14 different cantons. The main challenge to attain the goal was the high variability of the data between dialects leading to a high level of inconsistency and data sparsity. The challenge had defined the research questions of the thesis: 1) the techniques and architectures, which can be useful in order to improve the acoustic model in the multi-dialect scenario and with limited data resources, and 2) problems remaining after adapting the standard approaches.

All the experiments were done on the data from the first multi-dialect fully transcribed corpus of spoken Swiss German, the ArchiMob corpus. The data used for the training is manually transcribed with approximate phonemic, or 'dialectal', transcriptions, which provide close correspondence between grapheme labels and the acoustic signal.

The hybrid DNN-HMM approach was used as the ASR framework. I have started with a baseline HMM-DNN discriminative model and extended it adding more training data and using a different DNN architecture with data augmentation and more acoustic features. The results showed that the best performance was obtained with the TDNN model with the iVectors speaker-adapted features and data perturbation. All the additional techniques used to improve the GMM models have gained none or only minor improvement to the output NN model of the pipeline, which proves their relatively small impact.

Some problems stay unsolved even using the best acoustic model. First, the system performs worse on the out-of-domain data; it evidences that there is still much room to improve its generalisation capacity. Second, evaluation on different dialects results into very different scores. When building a multi-dialect ASR it is never possible to achieve the same performance for all the varieties. Yet, a more steady performance would be beneficial for the purpose of usage of the system.

The main outcome of the thesis is the first multi-dialect ASR for Swiss German trained on the dialectal transcriptions, which performs with 43.18% WER and 23.3% FlexWER on the in-domain data and with 59.81% WER and 37.83% FlexWER on the out-of-domain data. This system is a strong baseline for future research on ASR for Swiss German.

References

M. A.-A. M. Abushariah, R. N. Ainon, R. Zainuddin, A. A. M. Alqudah, M. E. Ahmed, and O. O. Khalifa. Modern standard arabic speech corpus for implementing and evaluating automatic continuous speech recognition systems. *Journal of the Franklin Institute*, 349(7):2215–2242, 2012.

A. Acero and X. Huang. Augmented cepstral normalization for robust speech recognition. In *Proc. of IEEE Automatic Speech Recognition Workshop*, pages 146–147, 1995.

A. Ali, Y. Zhang, P. Cardinal, N. Dahak, S. Vogel, and J. Glass. A complete kaldi recipe for building arabic speech recognition systems. In *2014 IEEE spoken language technology workshop (SLT)*, pages 525–529. IEEE, 2014.

A. Ali, P. Bell, J. Glass, Y. Messaoui, H. Mubarak, S. Renals, and Y. Zhang. The mgb-2 challenge: Arabic multi-dialect broadcast media recognition. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 279–284. IEEE, 2016.

A. Ali, P. Nakov, P. Bell, and S. Renals. Werd: Using social text spelling variants for evaluating dialectal speech recognition. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 141–148. IEEE, 2017.

J. Andén and S. Mallat. Deep scattering spectrum. *IEEE Transactions on Signal Processing*, 62(16):4114–4128, 2014.

H. Arsikere, A. Sapru, and S. Garimella. Multi-dialect acoustic modeling using phone mapping and online i-vectors. *Proc. Interspeech 2019*, pages 2125–2129, 2019.

J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

M. H. Bahari, R. Saeidi, D. Van Leeuwen, et al. Accent recognition using i-vector, gaussian mean supervector and gaussian posterior probability supervector for

spontaneous telephone speech. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7344–7348. IEEE, 2013.

L. Bahl, P. Brown, P. De Souza, and R. Mercer. Maximum mutual information estimation of hidden markov model parameters for speech recognition. In *ICASSP'86. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 11, pages 49–52. IEEE, 1986.

C. T. Batista, A. L. Dias, and N. C. S. Neto. Baseline acoustic models for brazilian portuguese using kaldi tools. In *IberSPEECH*, pages 77–81, 2018.

L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

H. Behravan, V. Hautamäki, and T. Kinnunen. Foreign accent detection from spoken finnish using i-vectors. In *INTERSPEECH*, volume 2013, page 14th, 2013.

M. Belenko and P. Balakshin. Comparative analysis of speech recognition systems with open code. *Mezhdunarodnyj nauchno-issledovatel'skij zhurnal*, (04 (58) Chast' 4):13–18, 2017.

Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

C. Caduff. Sprachgrenzen: Mundartliteratur:" frou" oder" pfrou"? *Literarischer Monat*, (21):10–13, 2015.

F. Campillo. Kaldi-based framework to train and evaluate acoustic models with the archimob corpus. user manual. *Technical report of the University of Zürich*, 2018.

N. F. Chen, W. Shen, and J. P. Campbell. A linguistically-informative approach to dialect recognition using dialect-discriminating context-dependent phonetic models. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5014–5017. IEEE, 2010.

G. Cheng, V. Peddinti, D. Povey, V. Manohar, S. Khudanpur, and Y. Yan. An exploration of dropout with lstms. In *Interspeech*, pages 1586–1590, 2017.

C. Chesta, O. Siohan, and C.-H. Lee. Maximum a posteriori linear regression for hidden markov model adaptation. In *Sixth European Conference on Speech Communication and Technology*, 1999.

C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina, et al. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4774–4778. IEEE, 2018.

S. Clematide, K. Frick, N. Aepli, and J.-P. Goldman. Crowdsourcing swiss dialect transcriptions for assessing factors in writing variations. *Bochumer Linguistische Arbeitsberichte*, pages 62–67, 2016.

G. E. Dahl, D. Yu, L. Deng, and A. Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on audio, speech, and language processing*, 20(1):30–42, 2011.

S. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE transactions on acoustics, speech, and signal processing*, 28(4):357–366, 1980.

N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):788–798, 2010.

A. DeMarco and S. J. Cox. Iterative classification of regional british accents in i-vector space. In *Symposium on machine learning in speech and language processing*, 2012.

S. P. Dubagunta and M. M. Doss. Segment-level training of anns based on acoustic confidence measures for hybrid hmm/ann speech recognition. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6435–6439. IEEE, 2019.

V. Dumoulin, J. Shlens, and M. Kudlur. A learned representation for artistic style. *arXiv preprint arXiv:1610.07629*, 2016.

E. Eide and H. Gish. A parametric approach to vocal tract length normalization. In *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, volume 1, pages 346–348. IEEE, 1996.

M. Elfeky, M. Bastani, X. Velez, P. Moreno, and A. Waters. Towards acoustic model unification across dialects. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 624–628. IEEE, 2016.

M. G. Elfeky, P. Moreno, and V. Soto. Multi-dialectical languages effect on speech recognition: Too much choice can hurt. *Procedia Computer Science*, 128:1–8, 2018.

D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.

D. Eugen and S. Dialäktschrift. Leitfaden einer einheitlichen schreibweise für alle dialekte, 1938.

C. A. Ferguson. Diglossia. *word*, 15(2):325–340, 1959.

J. Fleischer and S. Schmid. Zurich german. *Journal of the International Phonetic Association*, 36(2):243–253, 2006.

G. D. Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

C. Gaida, P. Lange, R. Petrick, P. Proba, A. Malatawy, and D. Suendermann-Oeft. Comparing open-source speech recognition toolkits. In *Technical Report of the Project OASIS*. 2014.

M. Gales and S. Young. The application of hidden markov models in speech recognition. *Foundations and trends in signal processing*, 1(3):195–304, 2007.

M. J. Gales. Maximum likelihood linear transformations for hmm-based speech recognition. *Computer speech & language*, 12(2):75–98, 1998.

M. J. Gales. Semi-tied covariance matrices for hidden markov models. *IEEE transactions on speech and audio processing*, 7(3):272–281, 1999.

M. J. Gales and P. C. Woodland. Mean and variance adaptation within the mllr framework. *Computer Speech & Language*, 10(4):249–264, 1996.

P. N. Garner, D. Imseng, and T. Meyer. Automatic speech recognition and translation of a swiss german dialect: Walliserdeutsch. In *Proceedings of Interspeech*, number CONF, 2014.

J.-L. Gauvain and C.-H. Lee. Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains. *IEEE transactions on speech and audio processing*, 2(2):291–298, 1994.

F. Germann, M. Ulasik, and M. Cieliebak. Quality assessment of automatic speech recognition systems. *Presentation at the Swiss Text conference*, 2019.

E. Glaser, C. Bucheli Berger, G. Seiler, A. Ender, A. Leemann, and B. Wälchli. Is a syntactic dialectology possible? contributions from swiss german. *Trends in Linguistics. Studies and Monographs (TiLSM)*, (247):93–120, 2012.

V. Goel and W. J. Byrne. Minimum bayes-risk automatic speech recognition. *Computer Speech & Language*, 14(2):115–135, 2000.

Y. Goldberg and G. Hirst. Neural network methods in natural language processing. morgan & claypool publishers(2017), 2017.

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

R. A. Gopinath. Maximum likelihood modeling with gaussian distributions for classification. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, volume 2, pages 661–664. IEEE, 1998.

H. Gordon, C. Bless, P. Ströbel, and F. Stadler. Archimob corpus release 1.0 documentation. 2016.

A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

F. Grézl, M. Karafiát, S. Kontár, and J. Cernocky. Probabilistic and bottle-neck features for lvcsr of meetings. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, volume 4, pages IV–757. IEEE, 2007.

R. Haeb-Umbach and H. Ney. Linear discriminant analysis for improved large vocabulary continuous speech recognition. In *Proc. ICASSP*, volume 1, pages 13–16. USA: ICASSP, 1992.

E. Haugen. Dialect, language, nation 1. *American anthropologist*, 68(4):922–935, 1966.

H. Hermansky. Perceptual linear predictive (plp) analysis of speech. *the Journal of the Acoustical Society of America*, 87(4):1738–1752, 1990.

H. Hermansky and S. Sharma. Temporal patterns (traps) in asr of noisy speech. In *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No. 99CH36258)*, volume 1, pages 289–292. IEEE, 1999.

H. Hermansky, D. P. Ellis, and S. Sharma. Tandem connectionist feature extraction for conventional hmm systems. In *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 00CH37100)*, volume 3, pages 1635–1638. IEEE, 2000.

G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

H. Höge, C. Draxler, H. v. d. Heuvel, F. T. Johansen, E. Sanders, and H. S. Tropf. Speechdat multilingual speech databases for teleservices: across the finish line. In *Sixth European Conference on Speech Communication and Technology*, 1999.

M. A. Hogg, N. Joyce, and D. Abrams. Diglossia in switzerland? a social identity analysis of speaker evaluations. *Journal of language and social psychology*, 3(3): 185–196, 1984.

N. Hollenstein and N. Aepli. A resource for natural language processing of swiss german dialects. 2015.

R. Hotzenköcherle, N. Bigler, R. Schläpfer, and R. Börlin. *Die Sprachlandschaften der deutschen Schweiz*, volume 1. Sauerländer Aarau; Frankfurt aM; Salzburg, 1984.

M.-Y. Hwang and X. Huang. Subphonetic modeling for speech recognition. In *Proceedings of the workshop on Speech and Natural Language*, pages 174–179. Association for Computational Linguistics, 1992.

D. Imseng, H. Bourlard, H. Caesar, P. N. Garner, G. Lecorvé, and A. Nanchen. Mediaparl: Bilingual mixed language accented speech database. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 263–268. IEEE, 2012.

J. Iranzo-Sánchez, J. A. Silvestre-Cerdà, J. Jorge, N. Roselló, A. Giménez, A. Sanchis, J. Civera, and A. Juan. Europarl-st: A multilingual corpus for speech translation of parliamentary debates. *arXiv preprint arXiv:1911.03167*, 2019.

A. Jain, M. Upreti, and P. Jyothi. Improved accented speech recognition using accent embeddings and multi-task learning. In *Interspeech*, pages 2454–2458, 2018.

N. Jaitly and G. E. Hinton. Vocal tract length perturbation (vtlp) improves speech recognition. In *Proc. ICML Workshop on Deep Learning for Audio, Speech and Language*, volume 117, 2013.

D. Jurafsky and J. H. Martin. *Speech and Language Processing (2Nd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2009. ISBN 0131873210.

S. Kapadia, V. Valtchev, and S. J. Young. Mmi training for continuous phoneme recognition on the timit database. In *1993 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 491–494. IEEE, 1993.

M. Karafiát, L. Burget, P. Matějka, O. Glembek, and J. Černockỳ. ivector-based discriminative adaptation for automatic speech recognition. In *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*, pages 152–157. IEEE, 2011.

V. Këpuska and G. Bohouta. Comparing speech recognition systems (microsoft api, google api and cmu sphinx). *Int. J. Eng. Res. Appl*, 7(03):20–24, 2017.

T. Kew. Language representation and modelling for swiss german asr. *Thesis of the University of Zürich*, 2020.

T. Kim, I. Song, and Y. Bengio. Dynamic layer normalization for adaptive neural acoustic modeling in speech recognition. *arXiv preprint arXiv:1707.06065*, 2017.

B. Kingsbury. Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3761–3764. IEEE, 2009.

T. Ko, V. Peddinti, D. Povey, and S. Khudanpur. Audio augmentation for speech recognition. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

P. Ladefoged and K. Johnson. *A course in phonetics*. Nelson Education, 2014.

Y. LeCun, Y. Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

C. Leggetter and P. C. Woodland. Speaker adaptation of continuous density hmms using multivariate linear regression. In *Third International Conference on Spoken Language Processing*, 1994.

P. Lenz. *Der Goalie bin ig: Roman*, volume 4. Schwabe AG, 2014.

B. Li, T. N. Sainath, K. C. Sim, M. Bacchiani, E. Weinstein, P. Nguyen, Z. Chen, Y. Wu, and K. Rao. Multi-dialect speech recognition with a single sequence-to-sequence model. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4749–4753. IEEE, 2018.

Y. Li, H. Erdogan, Y. Gao, and E. Marcheret. Incremental on-line feature space mllr adaptation for telephony speech recognition. In *Seventh International Conference on Spoken Language Processing*, 2002.

F.-H. Liu, R. M. Stern, X. Huang, and A. Acero. Efficient cepstral normalization for robust speech recognition. In *Proceedings of the workshop on Human Language Technology*, pages 69–74. Association for Computational Linguistics, 1993.

A. L. Maas, P. Qi, Z. Xie, A. Y. Hannun, C. T. Lengerich, D. Jurafsky, and A. Y. Ng. Building dnn acoustic models for large vocabulary speech recognition. *Computer Speech & Language*, 41:195–213, 2017.

W. Marti. *Bärndütschi Schrybwys: eine Wegweiser zum Aufschreiben in berndeutscher Sprache: mit einer Einführung über allgemeine Probleme des Aufschreibens und einem Wörterverzeichnis nebst Beispielen*. Francke, 1985.

R. Matarneh, S. Maksymova, V. Lyashenko, and N. Belova. Speech recognition systems: A comparative review. 2017.

I. P. Medennikov. Metody, algoritmy i programmyje sredstva raspoznavanija russoj telefonnoj spontannoj rechi. *M.: Prospekt*, 2015.

N. Mesgarani, S. Shamma, and M. Slaney. Speech discrimination based on multiscale spectro-temporal modulations. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages I–601. IEEE, 2004.

Y. Miao, H. Zhang, and F. Metze. Speaker adaptive training of deep neural network acoustic models using i-vectors. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(11):1938–1949, 2015.

A.-r. Mohamed, G. E. Dahl, and G. Hinton. Acoustic modeling using deep belief networks. *IEEE transactions on audio, speech, and language processing*, 20(1): 14–22, 2011.

M. Mohri, F. Pereira, and M. Riley. Speech recognition with weighted finite-state transducers. In *Springer Handbook of Speech Processing*, pages 559–584. Springer, 2008.

T. K. Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60, 1996.

M. Najafian, S. Safavi, J. H. Hansen, and M. Russell. Improving speech recognition using limited accent diverse british english training data with deep neural networks. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2016.

M. Najafian, W.-N. Hsu, A. Ali, and J. Glass. Automatic speech recognition of arabic multi-genre broadcast media. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 353–359. IEEE, 2017.

D. Palaz. Towards end-to-end speech recognition. Technical report, EPFL, 2016.

B. B. Partee, A. G. ter Meulen, and R. Wall. *Mathematical methods in linguistics*, volume 30. Springer Science & Business Media, 2012.

S. H. K. Parthasarathi, B. Hoffmeister, S. Matsoukas, A. Mandal, N. Strom, and S. Garimella. fmllr based feature-space speaker adaptation of dnn acoustic models. In *Sixteenth annual conference of the international speech communication association*, 2015.

V. Peddinti, D. Povey, and S. Khudanpur. A time delay neural network architecture for efficient modeling of long temporal contexts. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

D. Povey. *Discriminative training for large vocabulary speech recognition*. PhD thesis, University of Cambridge, 2005.

D. Povey and P. C. Woodland. Minimum phone error and i-smoothing for improved discriminative training. In *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages I–105. IEEE, 2002.

D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah. Boosted mmi for model and feature-space discriminative training. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4057–4060. IEEE, 2008.

D. Povey, L. Burget, M. Agarwal, P. Akyazi, K. Feng, A. Ghoshal, O. Glembek, N. K. Goel, M. Karafiát, A. Rastrow, et al. Subspace gaussian mixture models

for speech recognition. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4330–4333. IEEE, 2010.

D. Povey, L. Burget, M. Agarwal, P. Akyazi, F. Kai, A. Ghoshal, O. Glembek, N. Goel, M. Karafiát, A. Rastrow, et al. The subspace gaussian mixture model—a structured model for speech recognition. *Computer Speech & Language*, 25(2):404–439, 2011.

D. Povey, M. Hannemann, G. Boulianne, L. Burget, A. Ghoshal, M. Janda, M. Karafiát, S. Kombrink, P. Motlíček, Y. Qian, et al. Generating exact lattices in the wfst framework. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4213–4216. IEEE, 2012.

D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur. Purely sequence-trained neural networks for asr based on lattice-free mmi. In *Interspeech*, pages 2751–2755, 2016.

J. Psutka, L. Müller, and J. V. Psutka. Comparison of mfcc and plp parameterizations in the speaker independent continuous speech recognition task. In *Seventh European Conference on Speech Communication and Technology*, 2001.

S. P. Rath, D. Povey, K. Veselỳ, and J. Cernockỳ. Improved feature processing for deep neural networks. In *Interspeech*, pages 109–113, 2013.

M. Ravanelli. Deep learning for distant speech recognition. *arXiv preprint arXiv:1712.06086*, 2017.

M. Razavi, R. Rasipuram, and M. Magimai-Doss. On modeling context-dependent clustered states: Comparing hmm/gmm, hybrid hmm/ann and kl-hmm approaches. In *2014 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 7659–7663. IEEE, 2014.

H. Sak, F. de Chaumont Quitry, T. Sainath, K. Rao, et al. Acoustic modelling with cd-ctc-smbr lstm rnns. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 604–609. IEEE, 2015.

T. Samardzic, Y. Scherrer, and E. Glaser. Archimob-a corpus of spoken swiss german. 2016.

G. Saon, H. Soltau, D. Nahamoo, and M. Picheny. Speaker adaptation of neural network acoustic models using i-vectors. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 55–59. IEEE, 2013.

Y. Scherrer and P. Stoeckle. A quantitative approach to swiss german–dialectometric analyses and comparisons of linguistic levels. *Dialectologia et Geolinguistica*, 24(1):92–125, 2016.

Y. Scherrer, T. Samardžić, and E. Glaser. Archimob: ein multidialektales korpus schweizerdeutscher spontansprache. *Linguistik online*, 98(5):425–454, 2019a.

Y. Scherrer, T. Samardžić, and E. Glaser. Digitising swiss german: how to process and study a polycentric spoken language. *Language Resources and Evaluation*, pages 1–35, 2019b.

T. Schmidt and K. Wörner. *Multilingual corpora and multilingual corpus analysis*, volume 14. John Benjamins Publishing, 2012.

A. Senior and I. Lopez-Moreno. Improving dnn speaker independence with i-vector inputs. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 225–229. IEEE, 2014.

E. M. Shuey. Intelligibility of older versus younger adults' cvc productions. *Journal of communication disorders*, 22(6):437–444, 1989.

B. Siebenhaar. Code choice and code-switching in swiss-german internet relay chat rooms. *Journal of Sociolinguistics*, 10(4):481–506, 2006.

M. Stadtschnitzer and C. Schmidt. Data-driven pronunciation modeling of swiss german dialectal speech for automatic speech recognition. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

M. Stadtschnitzer, J. Schwenninger, D. Stein, and J. Köhler. Exploiting the large-scale german broadcast corpus to boost the fraunhofer iais speech recognition system. In *LREC*, pages 3887–3890, 2014.

A. Stępkowska et al. Diglossia: A critical overview of the swiss example. *Studia Linguistica Universitatis Iagellonicae Cracoviensis*, (129):199–209, 2012.

S. Ueberwasser and E. Stark. What's up, switzerland? a corpus-based research project in a multilingual country. *Linguistik online*, 84(5), 2017.

V. Valtchev, J. Odell, P. C. Woodland, and S. J. Young. Lattice-based discriminative training for large vocabulary speech recognition. In *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, volume 2, pages 605–608. IEEE, 1996.

P. Verma and P. K. Das. i-vectors in speech processing applications: a survey. *International Journal of Speech Technology*, 18(4):529–546, 2015.

K. Veselỳ, A. Ghoshal, L. Burget, and D. Povey. Sequence-discriminative training of deep neural networks. In *Interspeech*, volume 2013, pages 2345–2349, 2013.

R. Vipperla. Automatic speech recognition for ageing voices. 2011.

A. Waibel. Modular construction of time-delay neural networks for speech recognition. *Neural computation*, 1(1):39–46, 1989.

A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang. Phoneme recognition using time-delay neural networks. *IEEE transactions on acoustics, speech, and signal processing*, 37(3):328–339, 1989.

D. Wang, X. Wang, and S. Lv. An overview of end-to-end automatic speech recognition. *Symmetry*, 11(8):1018, 2019.

K. Wang, J. Zhang, Y. Wang, and L. Xie. Empirical evaluation of speaker adaptation on dnn based acoustic model. *arXiv preprint arXiv:1803.10146*, 2018.

E. W. Weisstein. Covariance matrix. `https://mathworld.wolfram.com/CovarianceMatrix.html`, 2003. [Online; accessed 28.05.20].

A. Williams and H. Giles. Sociopsychological perspectives on older people's language and communication. *Ageing & Society*, 11(2):103–126, 1991.

P. C. Woodland and D. Povey. Large scale discriminative training of hidden markov models for speech recognition. *Computer Speech & Language*, 16(1): 25–47, 2002.

X. Xie, X. Liu, T. Lee, and L. Wang. Fast dnn acoustic model speaker adaptation by learning hidden unit contribution features. *Proc. Interspeech 2019*, pages 759–763, 2019.

W. Xiong, L. Wu, F. Alleva, J. Droppo, X. Huang, and A. Stolcke. The microsoft 2017 conversational speech recognition system. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5934–5938. IEEE, 2018.

X. Yang, K. Audhkhasi, A. Rosenberg, S. Thomas, B. Ramabhadran, and M. Hasegawa-Johnson. Joint modeling of accents and acoustics for multi-accent speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2018.

S. Yoo, I. Song, and Y. Bengio. A highly adaptive acoustic model for accurate multi-dialect speech recognition. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5716–5720. IEEE, 2019.

T. Yoshioka, A. Ragni, and M. J. Gales. Investigation of unsupervised adaptation of dnn acoustic models with filter bank input. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6344–6348. IEEE, 2014.

D. Yu and L. Deng. *AUTOMATIC SPEECH RECOGNITION.* Springer, 2016.

D. Yu, L. Deng, and G. Dahl. Roles of pre-training and fine-tuning in context-dependent dbn-hmms for real-world speech recognition. In *Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2010.

D. Yu, K. Yao, H. Su, G. Li, and F. Seide. Kl-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7893–7897. IEEE, 2013.

D. Yu, L. Deng, F. T. B. Seide, and G. Li. Discriminative pretraining of deep neural networks, Jan. 12 2016. US Patent 9,235,799.

X. Zhang, J. Trmal, D. Povey, and S. Khudanpur. Improving deep neural network acoustic models using generalized maxout networks. In *2014 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 215–219. IEEE, 2014.

U. B. Zihlmann. Vowel and consonant length in four alemannic dialects and their influence on the respective varieties of swiss standard german.

Appendix

Path	Description
1. Data preparation	
data/lang/L.fst	the Finite State Transducer form of the lexicon
data/lang/L_disambig.fst	as L.fst including # disambiguation symbol
data/lang/oov.txt	<SPOKEN_NOISE>
data/lang/phones.txt	a_B 16
data/lang/words.txt	aabetruggt 7
data/lang/phones/...	
initial_data/data/spk2utt	<speaker-id> <utterance-id1> <utterance-id2> ...
initial_data/data/utt2spk	<utterance-id> <speaker-id>
initial_data/data/utt2dur	<utterance-id> duration
initial_data/data/wav.scp	<utterance-id> <path-to-wav>
initial_data/ling/extra_questions.txt	SIL SPN NSN
initial_data/ling/lexicon.txt	<word in grapheme> <word in phonemes>
initial_data/ling/lexiconp.txt	<word in grapheme> <probability> <word in phonemes>
initial_data/ling/nonsilence_phones.txt	a list of all phones
initial_data/ling/optional_silence.txt	SIL
initial_data/ling/silence_phones.txt	a list of all silence symbols
initial_data/data/temp/vocabulary.txt	a list of all words
temp/lang_temp/align_lexicon.txt	<word in grapheme> <word in states>
temp/lang_temp/lexiconp.txt	<word in grapheme> <probability> <word in states>
temp/lang_temp/lexiconp_disambig.txt	<word in grapheme> <probability> <word in states> <# disambiguation symbol>
temp/lang_temp/phone_map.txt	<phoneme> <a list of its states>
2. Feature extraction	
feats\cmvn_data.ark	cepstral mean and variance normalization
feats\cmvn_data.acp	<speaker-id> <path-to-cmvn>
feats\raw_mfcc_data.*.ark	mfcc features
feats\raw_mfcc_data.*.scp	<utterance-id> <path-to-features>

Table 15: Files created by the end of data preparation and feature extraction steps.

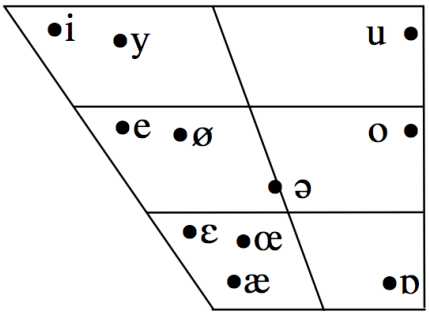

Figure 13: Zürich German vowel system [Fleischer and Schmid, 2006].

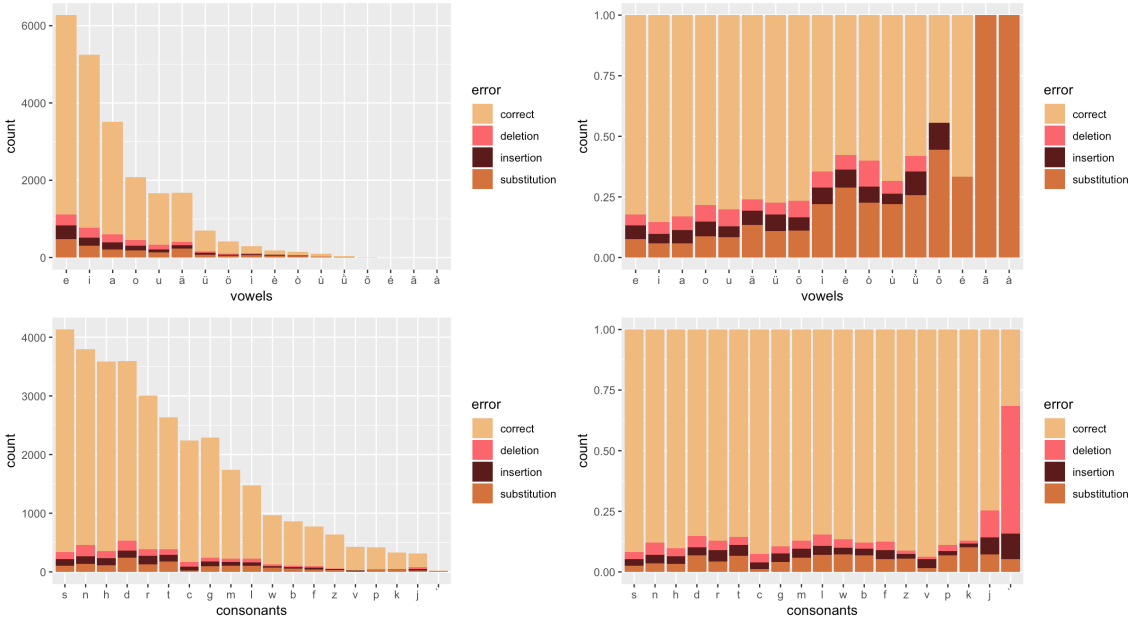

Figure 14: Distribution of errors for vowels (two upper plots) and consonants (two lower plots) for the TDNN-iVector model evaluated on the in-domain data.