Non-linear Cross-lingual Mappings for Low-resource Speech Recognition

Muhammad Umar Farooq

Supervisor: Prof. Thomas Hain

Department of Computer Science
University of Sheffield

This dissertation is submitted for the degree of
Doctor of Philosophy

July 2024

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text. Some of the material has been presented at or submitted to international conferences and journals (Ahmad et al., 2023, Farooq and Hain, 2023, 2022, Farooq et al., 2022, 2023). This dissertation contains fewer than 65,000 words including appendices, footnotes, tables and equations, but excluding the bibliography, and has fewer than 150 figures.

Muhammad Umar Farooq
July 2024

Acknowledgments

Allah Almighty is to be praised and countless salutations and blessings on His Last Prophet *Hazrat* Muhammad (salutations be upon him) who illuminated the world with the holy Quran, the ultimate source of knowledge.

I would like to thank my supervisor Prof. Thomas Hain, whose unwavering support and guidance have been invaluable throughout my PhD journey. He stands at the forefront of those from whom I have learned immensely in my life. Also to thank my panel members Anton Ragni, Prof. Guy Brown and Prof. Richard Clayton for their comments and advice.

It deeply moves me, and words fail to fully convey the support I've received from my parents. They have been my pillars of strength, the wellspring of my motivation, and the only reason for my aspirations in life. I am also immensely grateful to my siblings, who have played an invaluable role in shaping my life. I still remember the day trying to find a formula in a glass of water when Abu Bakar told 3 years old me that water has a chemical formula. I feel blessed to have Maria for being such an exceptional active listener, and Amna, my partner-in-crime from our school days all the way through to our graduation ceremony together.

It might be normal for everyone to find a PhD overwhelming and stressful, but I can't say that due to my wonderful colleagues. I can't imagine how my PhD journey would have been without Rehan and Asif. The support I received from Rosanna and Anna during my first PhD milestone was truly touching. While there are many names in my mind, I must mention Alex, Protima, Amit, and everyone else at Voicebase/LivePerson centre, MINI, and SPANDH who were supportive and with whom I had wonderful interactions.

Contents

List of Figures

List of Tables

Acronyms

AM Acoustic Model.

ASR Automatic Speech Recognition.

BPE Byte Pair Encoding.

BPPT Back-Propagation in time.

CD Context-Dependent.

CE Cross-Entropy.

CER Character Error Rate.

CI Context-Independent.

CNN Convolutional Neural Network.

CS Code-Switching.

CTC Connectionist Temporal Classification.

CTS Conversational Telephone Speech.

CV CommonVoice.

DNN Deep Neural Network.

e2e end-to-end.

ELU Exponential Linear Unit.

ES Elitist Sampling.

FC Fully-Connected.

FFN Feed-Forward Neural Network.

FLP Full Language Pack.

FWM Frame-wise maximum.

G2P Grapheme-to-Phoneme.

GMMs Gaussian Mixture Models.

GRU Gated Recurrent Unit.

HMM Hidden Markov Model.

IARPA Intelligence Advanced Research Project Activity.

IPA International Phonetic Alphabets.

KD Knowledge Distillation.

KL Kullback-Leibler.

LDA Linear Discriminant Analysis.

LDC Linguistic Data Consortium.

LF-MMI Lattice-Free Maximum Mutual Information.

LFCCs Linear-Frequency Cepstral Coefficients.

LHUC Learning Hidden Unit Contributions.

LID Language Identification.

LLP Limited Language Pack.

LM Language Model.

LS LibriSpeech.

LSTM Long Short-Term Memory.

LVCSR Large Vocabulary Continuous Speech Recognition.

MC Mapping Confidence.

MESD Multi Encoder Single Decoder.

MFCC Mel-Frequency Cepstral Coefficient.

MFCCs Mel-Frequency Cepstral Coefficients.

MLLR Maximum Likelihood Linear Regression.

MLLT Maximum Likelihood Linear Transformation.

MLP Multi-Layer Perceptron.

MLS Multilingual LibriSpeech.

MMI Maximum Mutual Information.

MPE Minimum Phone Error.

MSE Mean-Squared Error.

MTL Multi-Task Learning.

MUST Multilingual Student-Teacher.

OOD Out-of-Domain.

PER Phoneme Error Rate.

PLP Perceptual linear predictive.

POS Parts-of-Speech.

PTER Phonetic Token Error Rate.

ReLU Rectified Linear Unit.

RNN Recurrent Neural Network.

RNN-T RNN-Transducer.

SAT Speaker Adapted Training.

SAW Self-Adaptive Weighting.

sMBR state-level minimum Bayes risk.

SNR Signal-to-Noise Ratio.

SOTA State-of-the-art.

SWBD Switchboard.

TA Teacher averaging.

TDNN Time-Delayed Neural Network.

TF-IDF Term Frequency Inverse Document Frequency.

VAE Variational Auto-Encoder.

VTLP Vocal Tract Length Perturbation.

WER Word Error rate.

WFST Weighted Finite State Transducers.

WSJ Wall Street Journal.

Abstract

Multilingual speech recognition systems usually benefit low-resource languages but suffer degradation in the performance of several languages compared with their monolingual counterparts. With an objective to improve speech recognition performance for a target language, closer languages are chosen to build a multilingual system. The number of shared phonemes among them is usually taken into account to estimate the languages' closeness. However, various close languages such as English, German, Dutch and many others with significant phonemes overlap yield higher error rates from multilingual speech recognition systems when compared with their monolingual speech recognition systems. Limited attention has been paid towards investigating the performance trends of multilingual speech recognition systems and their relation with acoustic-phonetic similarities across the languages.

The objective of this research is to estimate cross-lingual acoustic-phonetic similarities and their impact on multilingual speech recognition systems. To that end, a novel data-driven approach is proposed to analyse the output from several monolingual acoustic models given a target language speech signal. This technique measures the similarities between posterior distributions from various monolingual acoustic models given a target language speech signal. Neural networks-based 'mapping models' are trained that transform the distributions from hybrid DNN-HMM acoustic models of different languages into a directly comparable form. The analysis shows that the 'closeness' among the languages can not be truly estimated by the size of the shared phonemes set. Entropy analysis of the proposed mapping models exhibits that a language with lesser overlap can be more amenable to cross-lingual transfer, and hence more beneficial in the multilingual setup.

The proposed mapping models are then exploited to improve low-resource speech recognition. A novel approach of hybrid DNN-HMM acoustic model fusion is proposed in a multilingual setup. Posterior distributions from different monolingual acoustic models, given a target language speech signal, are fused. Mapping models are trained for source-target language pairs to transform posteriors from a source acoustic model to the target language. These models require limited data as compared to the acoustic model training. Multilingual model fusion yields a relative average gain of 4.56% and 3.77% for selected languages from the Babel data set when compared with multilingual and monolingual

baselines respectively. Cross-lingual model fusion shows that comparable results can be achieved without using posteriors from the language-dependent ASR system.

Substantial phonemes overlap across the languages and the relatively smaller size of the universal phoneme set is expected to make the mapping task less challenging compared with mapping models for end-to-end ASR systems where tokens are usually graphemes or sub-word units. Furthermore, end-to-end speech recognition systems have been dominated over hybrid DNN-HMM models. So, the concept of learnable cross-lingual mappings is extended for end-to-end speech recognition to study if mappings could be learnt for end-to-end speech recognition systems. Mapping models are also employed to transliterate the source languages to the target language without using parallel data. Finally, the source audio and its transliteration are used for data augmentation to re-train the target language ASR system. The retrained speech recognition system with data augmentation results in a relative gain of up to 5% over the baseline monolingual speech recognition system for the selected languages from the Babel data set.

Student-teacher learning or knowledge distillation has been previously used to address data scarcity issues for the training of speech recognition systems. However, a limitation of knowledge distillation training is that the student model classes must be a proper or improper subset of the teacher model classes. It prevents distillation from even acoustically similar languages if the character sets are not the same. In this work, the aforementioned limitation is addressed by proposing a novel multilingual knowledge distillation approach that exploits the earlier proposed mapping models. A pre-trained mapping model is used to map posteriors from a teacher language ASR system to the student language ASR system. These mapped posteriors are used as soft labels for knowledge distillation. Various teacher ensemble schemes are experimented with to train an ASR system for low-resource languages. A model trained with MUST learning reduces relative character error rate up to 9.5% in comparison with a baseline monolingual ASR.

Chapter 1

Introduction

Speech and language technologies have revolutionised the conventional modes of human-machine interactions. The use of physical input hardware such as keyboards and joysticks etc. is now being replaced by machines which can listen, understand and respond to human speech and language. Lately, personal voice assistants such as *Siri*, *Cortana* and *Google Assistant* etc. have gained substantial popularity. A key component of such systems is an Automatic Speech Recognition (ASR) system which converts an input speech signal to the text.

1.1 Automatic speech recognition systems

Over the past decade, automatic speech recognition has been one of the most rapidly growing research topics. The dramatic improvement in ASR performances, with the introduction of recent modelling techniques such as transformers and self-supervised models, has resulted in the increasingly widespread use of speech recognition systems in real-world scenarios. Nowadays, speech recognition systems can be seen playing a role in many fields of life such as health care (Sherwani et al., 2007), agriculture (Patel et al., 2009), informational retrieval, military and security applications (Schultz and Kirchhoff, 2006a) are a few to name. Typical speech recognition systems, consisting of data-driven models trained using machine learning techniques, require:

1. Large amount of transcribed speech data for training of the **Acoustic Model (AM)**
2. A large text corpus for learning a **Language Model (LM)**
3. A **pronunciation model** (or lexicon) which maps words to distinct underlying representations such as sounds (phonemes) or characters etc of a language.

An acoustic model takes a speech signal as an input and outputs the probabilities of the language representations. These representations can be phonemes, characters or any other predefined tokens such as Byte Pair Encoding (BPE) (Sennrich et al., 2016)

tokens. The language model, on the other hand, helps to convert the probabilities of these representations to coherent language words. Language models are referred to as models that assign probabilities to word sequences (Jurafsky and Martin, 2008b) and are trained on (usually huge) text corpora of the language. Given a sequence of words, it predicts which words or sequence of the words are more likely (and thus semantically more plausible) than the others. For example, a language model would know that "*I'm from Pakistan, born and bred*" is more likely than "*I'm from Pakistan, born and bread*" although both *bred* and *bread* sound the same. Although pronunciation models have been required to build an ASR system for very long, their use has been drastically decreased with a paradigm shift towards Deep Neural Network (DNN) based end-to-end (e2e) speech recognition systems. The acoustic model is the main component of most State-of-the-art (SOTA) speech recognition systems. In e2e ASR systems training, an implicit language model is also learnt (Gong et al., 2022). However, external language models are also frequently integrated with these systems, significantly improving their performance (Chorowski and Jaitly, 2016, Hori et al., 2018).

Many companies and government organisations are investing in the development of speech recognition technologies with the rapid increase in the availability of super-fast processing units, open data resources and the emerging trend of smart devices (Chuangsuwanich, 2016). Most commercial applications aim to deal with clients from around the globe, and thus commercial interests are not limited to speech technologies of only one language but the systems which could process multiple languages (and ideally all the languages spoken around the globe). However, out of nearly 7000 languages that are spoken around the world, just 23 languages are spoken by more than half of the world's population (Ethnologue). Since the collection of language resources is usually very expensive and laborious, it is very hard to collect sufficiently large amounts of data for most of the languages to build their speech recognition systems (Lamel et al., 1995). Some spoken languages do not even have writing systems and it is difficult to get linguistic knowledge of some languages (Schultz and Kirchhoff, 2006b). Hence, only a few languages have been predominately focused on for the development of speech recognition systems (Burget et al., 2010, Kohler, 1996, Lamel et al., 1995).

Although it is very hard to build a speech recognition system for all the languages given the aforementioned constraints let alone the computational costs and technical issues, gradual efforts have been seen to build multilingual speech recognition systems with as many as possible languages (Conneau et al., 2021, Hou et al., 2020, Li et al., 2022a, Pratap et al., 2020a). A speech recognition system is considered multilingual if at least one of the two components (i.e. acoustic model and language model) is multilingual (Tong, 2020). Though modern speech recognition systems have achieved pretty impressive performance for the English language, multilingual ASR systems received scant attention until the recent past. They have stolen the limelight lately because of manifold objectives. Before delving deeper into multilingual speech recognition systems

and the motivation of this work, some crucial foundational terminology is defined in the following section to enhance the readability of this thesis.

1.2 Terminology

In this section, several terms are defined with the meaning adopted for this thesis. Some of the terminology has been used in different contexts in the literature. So, it is vital to define them here to better understand the rest of the thesis. Definitions here are intended to explain what they mean when encountered in this work.

- **Automatic speech recognition system:** The task of an ASR system is to map a waveform to the appropriate string of words (Jurafsky and Martin, 2008a). This term is used for a system consisting of machine learning model(s) which is capable of converting speech into text.
- **Acoustic model:** A machine learning model which takes speech as an input and outputs the probabilities of the language(s) representations i.e. phonemes, graphemes or any other predefined tokens.
- **Language model:** A statistical or machine learning model which outputs the probabilities of the next language token given the previous token(s).
- **Pronunciation model:** A pronunciation model or dictionary is a mapping of words to their pronunciations. It is usually a hand-crafted mapping of words to underlying phonemes by language experts. However, a pronunciation dictionary or model can have mappings from words to their graphemes or sub-words. It is also referred to as **lexicon**.
- **Monolingual speech recognition:** A speech recognition system with all of its models (acoustic and language) trained using the data of only one single specific language. It is alternatively termed as a language-dependent and language-specific speech recognition system as well.
- **Multilingual speech recognition system:** A speech recognition system is referred to as a multilingual speech recognition system when any of the above-defined models i.e. acoustic or language model is trained using the data from multiple languages. It is alternatively termed as a language-independent system as well. Unless specified otherwise, the term 'multilingual model' (either acoustic or language) refers to a model for which the training data of multiple languages is mixed before training and then batched randomly from the pool during training. Such models are usually evaluated on the languages seen in the training data. When an ASR system consists of both an acoustic model and an explicit language model, it would be considered multilingual if any of those models is multilingual. However, since most of the work in this thesis revolves around acoustic modelling, this

term widely refers the multilingual acoustic models unless explicitly mentioned otherwise.

- **Cross-lingual speech recognition:** The definition of this term relates more to the inference stage rather than the training. Cross-lingual speech recognition refers to using a multilingual speech recognition system to infer languages that are not included in its training data. Though it holds for both acoustic and language models, it mostly refers to acoustic modelling in this thesis unless specified explicitly.

- **International Phonetic Alphabets representations:** International Phonetic Alphabets (IPA) (Association, 1999) is an alphabetic system to represent the distinct sounds (***phones***) of spoken languages in the written form. It has been developed by the International Phonetic Association and is based on Romic alphabets.

- **Phone:** A phone is the smallest unit of speech perceived as a distinct sound by a listener (Hardcastle et al., 2010); for example [p], [b] and [ʔ]. They are usually represented by IPA symbols in the square brackets.

- **Phoneme:** The representation of a group of phones in a particular language which count as if they were the same for practical purposes (i.e. two phones in a language are one phoneme if replacing one with the other does not change the meaning of the word) (Jones, 1957). For example, the acoustic realisation of phoneme \p\can be [p^h] or [p] in English where changing either of the phones with the other, does not change the meaning of a word.

- **Allophones:** A set of phones which are used to pronounce one phoneme of a language (Society, 1961) and linguistically non-variant for that language. For example, phones [p] and [p^h] are allophones of phoneme \p\in English while these might be considered different phonemes in some other language such as they are in Urdu.

- **Grapheme:** A grapheme is the smallest unit of the writing system, serving to represent a phoneme (Coulmas, 1996). One or more than one graphemes can represent a sound. It can be the characters of a language as well as the punctuation marks etc.

- **Sub-word units:** In terms of speech recognition, a sub-word unit is any representation between the smallest units such as phonemes and graphemes to the larger units of words. It could be as small as a phoneme or a grapheme but usually smaller than a word.

After defining the necessary terminology, different approaches towards multilingual speech recognition, motivation and the challenges faced in multilingual speech recognition technologies are discussed. Then the motivation, objectives and contribution of this work will be presented briefly.

1.3 Multilingual ASR systems

In this section, different approaches towards multilingual ASR systems, the motivation of multilingual speech recognition systems and the main challenges faced by the research community are discussed.

1.3.1 Approaches towards multilingual speech recognition

In the last section, a multilingual ASR system has been defined as a system with models trained on multilingual data. However, there are various possible approaches to building a multilingual ASR system. Though it is discussed in detail during the literature review in Chapter 3, some examples of exploiting resources of multiple languages for speech recognition are given below.

- **Different acoustic and language models:** Usually ASR systems also make use of explicit language models (especially in the case of conventional Hidden Markov Model (HMM) based models). So, multilingual resources can be utilised to train either acoustic, language or both models. However, in any of these cases, the resulting ASR would be multilingual. For example, an acoustic model trained on data from multiple languages but with the language-dependent language models gives gain over the monolingual models (Billa, 2018). Though only the acoustic model is multilingual and the language model is monolingual, it would still be regarded as a multilingual ASR.

- **Multilingual ASR system as a feature extractor:** Studies have been done where multilingual ASR systems have been used to extract features for a target language ASR training (Ghoshal et al., 2013, Grézl et al., 2014, Veselý et al., 2012). The language-dependent models trained using these features prove to outperform the baseline monolingual ASR system. This is usually the case for e2e ASR systems where a multilingual ASR system is trained first. Then the target language data is passed through the pre-trained multilingual ASR system to extract representations from a hidden layer as the features. These extracted features are used to train a language-dependent ASR which proves to perform better than a system trained using conventional features (Thomas et al., 2012).

- **Multilingual ASR system for transfer learning:** In a similar nature to the above, language-independent ASR systems are used to train a monolingual ASR by transfer learning (Hou et al., 2020, Schultz and Waibel, 2001, Tong et al., 2018). This approach is particularly evident in end-to-end systems, where a language-specific system is initialised with the weights of a pre-trained multilingual model. This initialisation leverages the broad linguistic knowledge of the multilingual model, providing a strong foundation that enhances the performance of the target language model through fine-tuning. This model can be transferred to a monolingual model in two ways explored by (Tong et al., 2018) i.e. updating the whole

model during the re-training for target language data or updating only a few final layers. Their results show that monolingual models initialised from multilingual models outperform the models trained from random initialisation.

- **Multilingual ASR system as a unified model for multiple languages:** A very common and simple approach to training a multilingual model is to train a single model using training data of all the languages. However, this can also be done in two ways i.e. a model trained for a single task (Hou et al., 2020) or multiple tasks (Heigold et al., 2013). For a single task, data from all the participating languages is used to predict the output tokens of all the languages i.e. the output tokens would be $\cup_{i=1}^{L} O_{l_i}$ where O_{l_i} are the output tokens of l_i language and L is the total number of languages. In the multi-task approach, an e2e model contains some shared layers between all the languages and then some language-specific layers. So, the model is trained in a multi-task fashion where output tokens of each language are a separate task.

1.3.2 Motivation of multilingual speech recognition systems

The aforementioned approaches towards building multilingual speech recognition systems can be used in many different ways to build a multilingual system. These approaches are discussed in the literature review (Chapter 3) but the scope of this section is limited to the motivation and objectives of building multilingual or cross-lingual systems. *Why are scientists interested in building multilingual speech recognition systems?* As an answer to this question, the key motivations for building multilingual ASR systems are briefly described here.

- **A single model to recognise the speech of a large number of languages:** A possible way to build a speech recognition system for multiple languages is to cascade two systems i.e. Language Identification (LID) system followed by individual ASR systems for each language. In a cascaded system, the language spoken in an utterance is detected first using the LID system and then the corresponding ASR system is used to convert speech into text. However, if an error occurs in the initial stage of the cascaded systems, it is propagated to the following stages consequentially i.e. if the LID system misidentifies the language, the wrong ASR system is invoked and ultimately the whole output text would be wrong. Furthermore, it becomes cumbersome to train and maintain separate speech recognition systems as the number of languages increases (Li et al., 2022a, Pratap et al., 2020a). So, one objective of building multilingual speech recognition systems is to train a single model which could recognise speech from a large number of languages. It is referred to as *"massively multilingual"* as well.

- **Improving low-resource speech recognition:** Another objective of multilingual speech recognition is to exploit cross-lingual resources to improve the performance of speech recognition systems for low-resource languages (Choi and Park,

2022, Hsu et al., 2020, Klejch et al., 2022, Xu et al., 2022). Since modern speech recognition systems require a lot of transcribed data that low-resource languages lack, a multilingual speech recognition system can be trained using closer or related languages. It is usually expected that the closer languages help to increase the training data of shared representations across the languages and multilingual models perform better than the monolingual models for such languages. Some studies have also shown that a monolingual model bootstrapped from a pre-trained multilingual model also outperforms a monolingual model trained from random initialisation (Tong et al., 2018). Using multilingual models just to extract features to train the monolingual models also gives some gains (Thomas et al., 2012).

- **Code-switched speech recognition:** Code-Switching (CS), the spontaneous use of two or more languages in a single conversation, is a prevalent linguistic phenomenon in multi-cultural societies or countries where native and official languages are different. The speakers keep on mixing the dominant language (referred to as *matrix language*) and the second language (*embedded language*) in their daily communications. CS renders a monolingual ASR system confused about the language and muddles the context when the system confronts the embedded language (Farooq et al., 2020). This limitation of monolingual ASRs in the context of code-switched speech recognition has also attracted researchers towards multilingual speech recognition. Speech recognition systems that are capable of handling the complex acoustic and linguistic relationships between multiple languages are robust to code-switching speech.

1.3.3 Challenges for multilingual ASR systems

In this section, the key challenges in building multilingual speech recognition are described. The first two issues are more prominent and exacerbated in the case of e2e-based models where most of the languages have distinct output representations (graphemes or sub-word units).

- **The long tail problem:** In large-scale multilingual (synonymously referred to as massively multilingual) model training, a single model is trained for a large number of languages (Hou et al., 2020, Pratap et al., 2020a). However, the data across the languages usually is very skewed. There are some resource-high languages with thousands of hours of data and then the low-resource languages with a few tens of hours. It raises the challenge of efficient data sampling during training steps.

- **Convergence:** When a single model is trained for a large number of languages where most of the languages have different output representations (in the case of e2e systems), it becomes challenging for the model to learn all the tasks at the same time. Such scenarios make it hard for a model to learn smoothly and converge. This is an active research area and approaches such as lifelong learning

(Li et al., 2022a) and curriculum training (Pratap et al., 2020a) have been studied to overcome this problem.

- **Benchmark data sets**: Until recently, there have been very few benchmark data sets to train and evaluate multilingual ASR systems. The most standard ones included GlobalPhone (2002 and 2013) (Schultz, 2002, Schultz et al., 2013) and data set released under the IARPA Babel project (Gales et al., 2014) but both of them have not been freely available open resources. So, a lot of work on multilingual ASR has been reported on a wide range of non-standardised different data sets. It raises the issue of the lack of a standard benchmark data set and thus a standard baseline. This has led to the research community building multilingual speech recognition systems using non-standard data sets and comparing the proposed techniques with their baselines (Datta et al., 2020, Kannan et al., 2019). Although the performance gains of the proposed techniques can be seen in comparison with the given baselines, it is hard to compare two different techniques. A rise in standardised multilingual data sets has been seen since 2020 and various multilingual data sets such as Multilingual LibriSpeech (MLS) (Pratap et al., 2020b) and CommonVoice (CV) (Ardila et al., 2020) have been made publicly and freely available to the community. Additionally, some works show performances of their proposed systems on their in-house data sets which also makes it difficult to compare their approaches with other works in the literature. However, that is not an issue specifically associated with multilingual speech recognition research.

- **Criteria to mix languages for a multilingual ASR**: To improve low-resource languages, resources of a few languages (usually on a scale of 5-10) are exploited (Burget et al., 2010, Kannan et al., 2019, Madikeri et al., 2020). In most such cases, the performance from multilingual approaches improves the speech recognition performance when compared with monolingual counterparts of the low-resource languages. Usually, the choice of these languages is based on *'close languages'* (Datta et al., 2020). However, there is no solid definition of *closeness* in the literature and seems an open-ended term. It raises a question about which, how many and how close languages are needed to combine for the speech recognition improvement of a low-resource language.

1.4 Motivations and objectives

So far, the different definitions, motivations and challenges in multilingual ASR research have been discussed. In this section, the motivation for this work is described briefly. In this work, the research is primarily motivated by the last challenge in Section 1.3.3 i.e. what is language closeness and on which criteria languages are mixed to train a multilingual ASR for low-resource speech recognition improvement. Before diving into technical aspects, the problem is stated from linguistics aspects and ongoing research in the domain of multilingual ASR.

Summarising all the motivations of multilingual ASR (described in Section 1.3.2) into a single point, the ultimate objective of multilingual speech recognition is to build an '*universal ASR*'. A universal ASR would ideally be able to convert speech of any language to text. Massively multilingual is one way to reach the destination using very large data and computational resources. On the other hand, attempts are being made to achieve the same objectives with minimal effort in terms of data and computation (Li et al., 2020, 2022b). However, these approaches are quite challenging and far from SOTA performances yet.

The main challenge is to devise ways to optimise the use of multilingual resources to get the maximum gain not only in terms of the performance of ASR systems but also computational costs. According to a renowned linguist *Peter Ladefoged*, the sounds produced by humans which have some lexical information (phones) are never similar to those sounds which do not have any lexical information such as whistling, clicking teeth and waging tongue side to side etc. (Ladefoged, 1996). Although some phones might not be known yet due to lack of available data or some new phones may come into existence in future, those will be similar to the previously known phones or will be formed by rearrangement of properties of the known phones (Ladefoged, 1996). It implies that the sounds (phones) of all the languages are somehow related to each other. This argument appears quite plausible upon closer examination. Assuming that all languages have very diverse sounds, the statistical analysis of phones across the languages shows that some of the phones are more common in a lot of languages than others (Moran and McCloy, 2019). This important question of phonology was answered in the form of the Quantal theory of speech by *Ken Stevens* (Stevens, 1989). The quantal theory formalises the intuition that some sounds are easier to produce than others and are common across languages. For example, \m\, \k\and \j\are among the most common consonants with presence in 96% and 90% Phoible[1] inventories (Moran and McCloy, 2019).

Given the linguistic theories that the different sounds of diverse languages with some lexical information are somehow related, the motivation of multilingual and cross-lingual speech recognition is to make use of resources of other languages to recognise the speech of a target language or all the existing languages. However, the question that arises here is about how to optimally use those resources. It calls for devising ways to use available resources meaningfully for building a speech recognition system. Though linguistic theories about the similarities in the distinct speech sounds have been presented, linguists also admit the variations in different instances of an identical sound produced by the same speaker not to mention different speakers and languages. These variations are expected to become more prominent across the speakers and even further across the languages. For example, a spoken phone \p\of 80ms by a certain speaker might be 82ms long in another instance of the same speaker. Usually, the speech data for speech recognition is annotated at the phoneme level and an acoustic model is expected to learn

[1] Phoible is an online repository of phonological inventories of a huge number of languages. https://phoible.org

the variations of underlying acoustic realisations (phone and allophones). In the case of building a multilingual speech recognition system when the training data of all the languages is mixed, an underlying assumption is that the sounds represented by the same IPA symbols across the languages are acoustically close too (Schultz and Waibel, 2001). It implies that the training data of the phonemes (IPA representations) shared by more languages is increased and expected to be learned better during the model training.

Some earlier works attempted to measure phonemes similarities for context-independent (monophone) acoustic models (Kohler, 1996). A few later studies on the same lines measured the closeness among phonemes for context-dependent (triphone) acoustic models (Imperl et al., 2000, Le et al., 2006). The closeness has been measured based on the number of confusions between two phonemes p_i and p_j from an HMM model of the phoneme p_i (Imperl et al., 2000). Some studies have also made use of linguistic knowledge to measure the distance between polyphonemes (Le et al., 2006). However, the objective of these approaches has been to cluster the closer phonemes to build a multilingual ASR system rather than studying languages' similarities. A very limited work done could be found in the direction of cross-lingual similarities.

Most of the works on multilingual speech recognition systems combine languages either randomly or based on the number of shared phonemes (based on IPA representations) among the languages. However, various multilingual works have observed that the performance of multilingual systems does not show any improvement over the monolingual models in the case of resource-rich languages. In often cases, multilingual ASR systems perform worse than monolingual models for resource-high languages. Recent work has shown that the improvement in the error rate of a phoneme in a multilingual setup has no clear relation with the number of languages it is being shared with (Żelasko et al., 2020).

Assuming that it is possible to build a universal ASR using the available resources, it is needed to come up with ways to use these resources optimally. It calls for studies to analyse the relationships between the languages (referred to as '*cross-lingual similarities*' in this work).

Motivated by the literature observations that;

- A multilingual speech recognition system does not guarantee a reduction in the speech recognition error rate of a language when compared with a language-dependent ASR system,
- Performance of a multilingual ASR system is degraded for many languages with significant phoneme overlap,
- Language closeness is usually measured based on the number of shared phonemes (IPA representations) to build a multilingual system

- Limited attention has been paid towards defining criteria to choose languages to combine for building a multilingual ASR system,
- There are still a lot of languages with too limited resources to build a modern speech recognition system for them,
- Attempts are actively being made to build larger and larger ASR systems to recognise as many languages as possible.

Some research directions and gaps are identified as follows;

- An underlying assumption in the case of building multilingual systems that the phonemes across the languages with identical IPA representations are also acoustically similar may lead to performance degradation of multilingual setups,
- If the number of shared phonemes (IPA representations) is not a true measure of language closeness, how can it be decided if combining certain languages would help each other in multilingual setups,
- Given that there are still a lot of low-resource languages, it is need of time to devise ways of efficient exploitation of cross-lingual resources to build ASR systems for such languages.

Given the observations from the literature review and identified research gaps and directions, the objective of this research is to;

- Devise data-driven approach to measure cross-lingual acoustic phonetic similarities,
- Study effect of cross-lingual acoustic-phonetic similarities on the multilingual setup,
- Propose ways to improve low-resource speech recognition in light of insights from the aforementioned study and analysis.

1.4.1 The proposed approach

To analyse the acoustic-phonetic similarities among the languages, a data-driven approach is proposed to analyse how a monolingual acoustic model of a source language perceives the speech signal of some target language. As discussed earlier, some earlier studies counted the number of confusions to estimate the similarities between the phonemes (Imperl et al., 2000). However, it is argued that even if a phoneme of a target language is confused with some other phoneme by a source language acoustic model, there is a possibility of patterns in these confusions. If there are significant patterns in these confusions, the languages can still be considered close except that the acoustic realisations of some phonemes of the target language are different in the source language.

Furthermore, if such patterns exist to the extent that they could easily be learnt by a machine learning model, then the resources of a source language can be used to improve the speech recognition of the target language. This approach allows making use of several approaches in multilingual setup that have never been applied for multilingual ASR to the best of authors' knowledge. For example, outputs from diverse monolingual models are fused given a target language speech signal to improve target language ASR performance. As diverse monolingual acoustic models have different output tokens, their outputs cannot be fused straightforwardly. The proposed approach is employed for this kind of model fusion which has not been done before. Furthermore, a long-standing problem of multilingual knowledge distillation is also addressed. A novel data augmentation technique is also presented to improve low-resource speech recognition. Another merit of the proposed mapping approach is that it can be applied on top of any kind of ASR system i.e. hybrid DNN-HMM-based or end-to-end models.

1.5 Organisation of the thesis

The rest of the thesis is organised in two sections i.e. "Background" and "Contributions". Two chapters of the "Background" section cover the literature review on working, challenges, data sets, trends and techniques for building ASR systems in general and multilingual speech recognition systems. The chapters under the section of "Contributions" discuss the motivation, approaches, experimental setup and results of this research.

Chapter 2 discusses the foundations of speech recognition systems along with current trends in building ASR systems. It reviews the literature on different techniques over time for feature extraction and acoustic modelling.

In Chapter 3, developments in multilingual speech recognition are discussed. It covers the review of different approaches towards multilingual and cross-lingual speech recognition. Available multilingual speech data sets and their developments over time are also discussed. Furthermore, current trends in the multilingual speech recognition domain such as massively multilingual and approaches towards low-resource and universal ASR are also discussed.

The "Contributions" section starts from Chapter 4 which contains problem formulation and the initial experiments as an experimental base for the experiments. Furthermore, a detailed analysis is presented to show that the learnable patterns exist in the confusion when a speech signal of a target language is decoded through a source-language acoustic model.

After establishing experimental evidence about learnable cross-lingual mappings in the previous chapter, Chapter 5 covers extensive experimentation and discussion about learning the said mappings. Experimental setup, results and the trends in the results are discussed in detail. This chapter is partially based on our published work (Farooq et al., 2022).

Chapter 6 presents the proposed mapping approach for cross-lingual acoustic-phonetic similarities with evidence from extensive experimentation. Phoneme-based hybrid DNN-HMM models have been trained for different languages to study the phonemic similar ties in them. This chapter is based on the published work (Farooq and Hain, 2022).

In Chapter 7, a novel model fusion technique is proposed in multilingual and cross-lingual setups to improve low-resource speech recognition. The learnable mappings proposed in earlier chapters are exploited for fusing output from diverse monolingual models to recognise the speech of an under-resourced language. This chapter is built upon the published work (Farooq et al., 2022).

The analysis of cross-lingual acoustic-phonetic similarities and model-fusion techniques have been applied on top of phoneme-based hybrid DNN-HMM-based models so far. The study shows that the mapping models can learn mappings on top of phoneme-based hybrid DNN-HMM models. However, the phonemic vocabulary of the languages is a subset of the universal phoneme set which is a limited set and usually the languages have some overlap among them. So, it suggests that it might be easier for a machine learning model to learn the mappings on top of such models as compared to the e2e models. So, in Chapter 8 the proposed approach is extended for e2e models. Mapping models on top of e2e models are further used to devise a novel data augmentation technique. The work done in this chapter has also been published (Farooq and Hain, 2023).

Knowledge distillation (Hinton et al., 2015) has been widely applied for various tasks where a single or ensemble of a few teacher models is used to distil their knowledge to a student model of the same or reduced capacity. However, a limitation of this technique is that the output tokens of a student model must be a proper or improper subset of the teacher model(s) output tokens. It is not usually the case if it is aimed to use ASR systems from different languages to distil knowledge to some other language's model. The reason is that the output tokens of different languages are usually very diverse due to different writing scripts and characters etc. Chapter 9 exploits the proposed approach to overcome this long-standing issue and proves that distilling knowledge from teachers of another language using the proposed approach, help improving speech recognition of a under-resourced language. This chapter is motivated by the collaborative work (Ahmad et al., 2023) and describes our published work (Farooq et al., 2023).

Finally, all the studies, experimentation, outcomes and take-home lessons are summarised in the Chapter 10.

Part I

Background

Chapter 2

Automatic Speech Recognition

In this chapter, foundational and state-of-the-art techniques of building an ASR system are revisited. Previous work and current trends in feature extraction, training and decoding techniques are discussed.

Automatic speech recognition - the conversion of a speech signal into a string of words - is a difficult task due to several cumulative sources of variation. Defining the boundaries of speech units in a speech signal is not straightforward irrespective of speech units (e.g. phonemes, words or sub-words). The task gets more complicated due to phenomena such as varying speech speed, and context-dependencies such as co-articulation and prosody. The complexity level of speech recognition can be factorised into four dimensions (Renals and King, 2010).

1. Vocabulary size: One of the earlier widely deployed speech recognising systems was built with a vocabulary size of only two words i.e. "yes" and "no". However, for a conversation-like human-human or human-computer interaction, much larger vocabularies are needed (Renals and King, 2010).

2. Speaking style: Earlier systems such as the "yes"/"no" recognition system were able to recognise isolated words. Pauses were left between words to let the system know the end of a word. However, a real conversation consists of connected or continuous speech.

3. Speaker characteristics: Speech from each speaker has different characteristics which makes it harder for an ASR system to recognise speech from an unseen speaker. So, usually speaker-dependent ASR system gives better performance than a speaker-independent ASR.

4. Background environment: It is less challenging for an ASR system to learn from clean speech data compared to the data recorded with significant background noise. Furthermore, different recording equipment produces different quality of data such

as telephonic speech is more challenging to train a speech recognition system rather than a microphone-recorded speech.

Given the complex nature of the speech recognition task, it is very hard to build a robust speech recognition system to recognise the speech with minimum error incorporating all the aforementioned variations. However, a recent analysis of sexennial surveys of speech community (Moore and Marxer, 2023) reveals that a lot of speech recognition applications considered nearly impossible in past are now deemed as just a few years away. A survey in the early 2000s stated that the performance of speech recognition systems has been improved by 10% per decade in the last three decades (Deng and Huang, 2004). This is mainly because of the massive improvements seen in the performance due to a combination of algorithmic and modelling improvements and increased computational resources (Renals and King, 2010).

Modern speech recognition systems are based on data-driven methods. Building on fundamental research of the 1970s, these approaches are based on the noisy channel model. In the noisy channel model theory, the human speech production system is considered a noisy channel between the words in the mind of the speaker and the spoken speech waveform. The goal of a speech recogniser is to decode the speech signal to get the correct word sequence. Let X be a sequence of acoustic feature vectors, a speech recognition system aims to predict the most probable sequence of words given the acoustic feature vectors. In terms of probability, it can be written as;

$$W^* = \arg\max_W P(W|X)$$

where W^* is the most probable word sequence among all possible word sequences W given the input utterance X. Using the Bayes theorem, this equation can be rewritten as;

$$W^* = \arg\max_W \frac{P(X|W)P(W)}{P(X)}$$

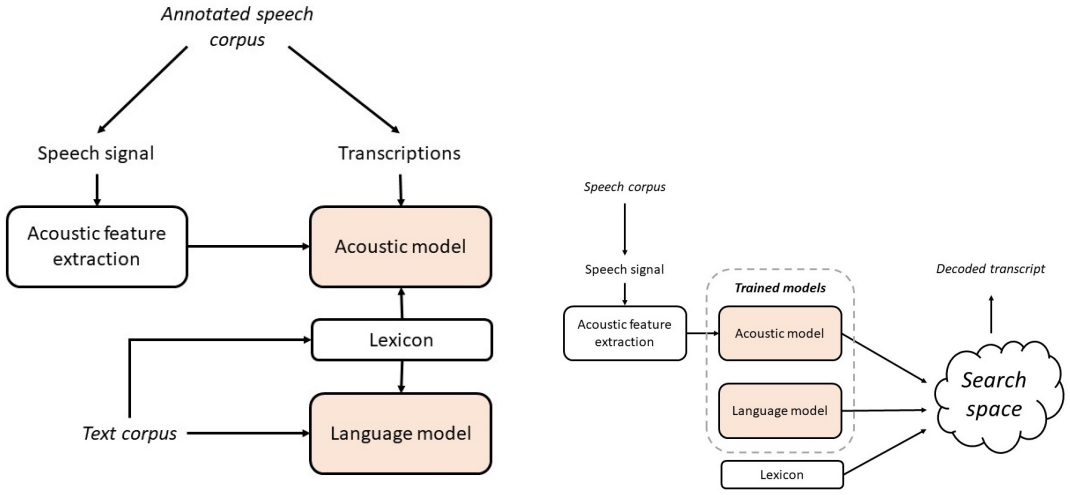
As only the word sequence with the highest probability is needed and the denominator term has the same effect on all the probabilities, the most probable word sequence is the one with the highest numerator value;

$$\frac{P(X|W)P(W)}{P(X)} \propto P(X|W)P(W)$$

So, the most likely word sequence $W*$ is given by;

$$W^* = \arg\max_W P(X|W)P(W) \tag{2.1}$$

$P(X|W)$ is the probability of the acoustic feature vectors given the sequence of words and $P(W)$ is prior for the probability of the sequence of words in the language (and is not conditioned on speech). In Equation 2, $P(X|W)$ is estimated by an *acoustic model*



(A) Schematic overview of training a data-driven automatic speech recognition system. An annotated speech corpus is used to train an acoustic model (along with a lexicon in hybrid GMM-HMM or DNN-HMM systems). Text corpora are used to train a language model.

(B) Schematic overview of decoding a speech utterance into text spoken in it employing a data-driven trained ASR system.

FIGURE 2.1: Schematic overview of (A) training and (B) decoding in an ASR system

whereas $P(W)$ comes from a *language model*. So, generally, a speech recognition system is divided into four interacting modules;

1. ***Acoustic feature extraction:*** A process to extract suitable features from speech waveform to train an acoustic model.
2. ***Acoustic model:*** A machine learning model which is usually trained using transcribed speech corpus. It relates the acoustic features to the output tokens such as phonemes, graphemes or sub-words.
3. ***Language model:*** A statistical or machine learning model which is trained using a large text corpus of the target language. It gives the probability of a sequence of words in the language.
4. ***Pronunciation model:*** Pronunciation dictionary or lexicon is a key component in hierarchical models to specify how the words are made from basic speech units.

The aforementioned four components are shown in Figure 2.1 to depict their use in training (Figure 2.1a) and inference (Figure 2.1b) in a speech recognition system. In the following sections, acoustic feature extraction and modelling techniques are described. Before training a speech recognition system, some pre-processing steps are also performed for both, speech signals and text transcriptions. In Section 2.1, the pre-processing

steps such as framing and acoustic feature extraction etc. are discussed. Then, various acoustic modelling techniques such as GMM-HMMs, hybrid DNN-HMM modelling and end-to-end speech recognition systems are explained in Section 2.2.

2.1 Pre-processing

There are a few important steps to take before training a speech recognition system to pre-process both speech waveforms and transcriptions. Sometimes a speech signal is processed to separate speech and non-speech segments, remove noise or enhance the speech quality before the feature extraction step (Lee et al., 2022). However, only the acoustic feature extraction is discussed in this section as a pre-processing step of the acoustic signal. Transcriptions of the speech signals, on the other hand, also need normalisation and standardisation steps sometimes which are discussed here.

2.1.1 Acoustic feature extraction

The goal of acoustic feature extraction is to keep information from a speech signal which is important to the speech recognition task and discard the information which is not helpful towards speech recognition. A very diverse range of transformations have been seen over time to extract the acoustic features. Some of the widely used are as follows;

- Mel-Frequency Cepstral Coefficients (MFCCs) (Davis and Mermelstein, 1980)
- Perceptual linear predictive (PLP) (Hermansky, 1990) features
- Filterbanks (Chougule et al., 2014)

Feature extraction or pre-processing is an important step to remove irrelevant information from the input. A feature extractor transforms an input speech signal into a sequence of vectors. Various kinds of approaches have been used for feature extraction from a speech signal. However, MFCC is the one most widely used. Filterbank features, speech spectrum and raw waveform have also been used for e2e speech recognition. MFCCs and filterbanks are discussed in detail here. Before delving further into acoustic feature extraction, basic speech processing components such as sampling, sampling frequency and framing etc. are introduced in the following section.

2.1.1.1 Speech processing

A machine learning model is usually trained using a discrete number of training examples. For example, a set of daily observations for a past period such as temperature, precipitation and wind speed on each day can be used to train a weather forecast system. Considering each day as a single training example, the observations on each day i.e. temperature, precipitation and wind speed are the features associated with each example. This is pretty much the case for training a machine learning model for any structured data (in the form of numbers and tables etc). However, language processing

is a difficult problem since language resources such as text and speech etc. are considered unstructured data. Speech is a continuous signal with a lot of spoken phones without any information about their boundaries within the signal. In the case of speech, a discrete number of training examples with known targets are not available, unlike the structured data. So, some pre-processing is needed for segmenting a speech signal into training examples, extracting features and assigning target labels to each example. This is discussed after briefly describing some speech-processing basics.

Speech is an analogue quantity in its nature which needs to be converted into digital form for storage and processing. For this purpose, the signal is sampled at regular intervals. Sampling involves taking discrete values of continuous signals after uniform intervals. The **sampling rate** or **sampling frequency** is the number of samples taken in one second of a signal. For example, a sampling rate of 16KHz means taking 16000 per second of an audio signal. **Nyquist theorem** (Shannon, 1949), which is a fundamental signal processing concept, provides guidelines for proper sampling of a continuous system. It states that the sampling frequency of a signal must be at least twice the highest frequency in the signal for its lossless reconstruction. In the case of audio, the human hearing frequency ranges from 20Hz to 20KHz (Moore, 2014). According to the Nyquist theorem then, it implies that the maximum possible frequency audible to a human might reach 20KHz and audio should be sampled at a minimum of 40 kHz to ensure lossless reconstruction. However, sampling at higher frequency means a large number of samples per second and thus larger file sizes and higher processing time.

Typically speech covers the frequency range of 30 to 10KHz, and most of the energy is in the range of 200 to 3500 Hz (Sobolewski, 2003). Although there is some information in higher frequency bands too, the human ear is not very sensitive to small frequency changes and humans can correct small things such as missing syllables or words etc (Sobolewski, 2003). So, it is not necessarily needed to sample speech at a frequency of 40KHz. In telephony, speech is usually sampled at 8KHz to save the bandwidth. To increase information retention, usually better quality data is recorded at a sampling frequency of 16KHz. However, high-fidelity data such as broadcast and music is usually recorded at even higher frequencies such as 44.1 KHz and 48 KHz. Most of the speech data sets recorded using headsets, hands-free or table microphones etc. are recorded at a sampling frequency of 16 KHz. Data sets consisting of telephonic speech usually come with a sampling frequency of 8KHz. Some data sets recorded with microphones might still be recorded at higher frequencies such as 48KHz.

Going back to the pre-processing of a speech signal for acoustic feature extraction, a speech signal is segmented into fixed-sized overlapping frames. For conventional ASR building, these frames are then aligned with the output tokens. Each frame and the corresponding output token are then used as a training example. Usually for speech recognition, the frame width is set to 20-30 ms and the frame shift is usually 10 ms. Acoustic feature extraction techniques are then applied to these frames. The widely

accepted methods of acoustic feature extraction i.e. MFCCs and filterbanks are discussed here.

2.1.1.2 Mel-frequency cepstral coefficients

The use of Mel-frequency cepstral coefficients was first proposed for speech recognition in 1980 which outperformed the Linear-Frequency Cepstral Coefficients (LFCCs) and linear-prediction cepstral and spectral coefficients (Davis and Mermelstein, 1980). The words *'cepstral'* and *'cepstrum'* have been coined from inverting the words 'spectral' and 'spectrum'.

During the process of LFCCs or MFCCs etc. extraction, the signal is transformed twice i.e. Fourier transform and cosine transform. The Fourier transform provides frequency information of a signal (frequency spectrum of audio), while the second transform (cosine transform) implies spectrum-of-a-spectrum. So, the final output is termed a cepstrum. Since there is a non-linear function of $\log(\cdot)$ between both the transforms, this is a non-linear spectrum-of-a-spectrum.

Human hearing sensitivity is not linear for all the frequencies in the audible spectrum. It is more sensitive to the frequencies in the lower range (Zwicker, 2005). In LFCC extraction, linear frequency bands are used where the spacing between all consecutive frequencies is linear. However, the Mel-frequency scale approximates human hearing perception better and emphasises lower frequencies more than the higher frequencies. A transform function for a linear frequency f_l to Mel frequency f_m is defined as;

$$f_m = 1127 \cdot \ln(1 + \frac{f_l}{700})$$

After applying the Fourier transform, K number of bandpass filters are applied and log-energy of each filter is computed. Filters could be of any shape, triangular filters have been simulated in the original paper (Davis and Mermelstein, 1980) and are still widely used. MFCCs are then computed by taking the cosine transform of these log energies in decibels. i^{th} coefficient is calculated as;

$$MFCC_i = \sum_{k=1}^{K} X_k cos[i(k - \frac{1}{2})\frac{\pi}{K}] \tag{2.2}$$

where X_k is the log-energy of the k^{th} filter. The schematic diagram of extracting MFCC features is shown in Figure 2.2.

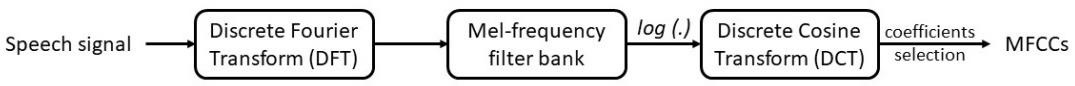


FIGURE 2.2: Schematic overview of steps involved in Mel-frequency cepstral coefficients extraction.

2.1.1.3 Filterbank

Filterbank (or fbank) features have also been employed for the training of speech recognition systems. The process of extracting filterbank is the same as MFCCs except that the energies of all the bandpass filters are concatenated in a vector for each frame. This vector is referred to as the filterbank features. However, filterbank features are highly correlated along its dimensions which is not very suitable for GMMs with diagonal covariance matrices (Li, 2022b). So, these features are not ideal for conventional ASR approaches such as GMM-HMM and are mostly used for training of e2e systems. The last step of discrete cosine transform in MFCCs decorrelates these energies (Li, 2022b). So, MFCCs are more widely used than the filterbank feature.

2.1.2 Text normalisation

Transcriptions of speech utterances also need some pre-processing steps before training a speech recognition system. The nature of pre-processing depends on the requirements of the model being trained. Most of the available speech data is annotated at the word level. Usually, it is human-annotated and sometimes might have been generated in a semi-supervised way (Pratap et al., 2020b).

Conventional HMM-based ASR systems are usually trained on phoneme-level output tokens. It could be trained using graphemes also (Dines and Magimai Doss, 2008, Kanthak and Ney, 2002, Killer et al., 2003) but mostly it is trained on the phonemes level. So, the word-level transcriptions are needed to be converted into phoneme-level transcriptions. A lexicon (or pronunciation model) is used for this purpose. A lexicon is usually a hand-crafted list of words with corresponding phoneme sequences for each word. As defined in Section 1.2, phonemes are the basic speech sounds of a language and are denoted using IPA symbols. Other than language phonemes, some additional representations are also used to indicate some other cues such as silence, unknown, noise, start and end of the utterance etc. The unknown symbol is usually used to annotate an out-of-vocabulary word (which is not present in the lexicon) during training. Further details on how to train an HMM model and perform decoding using these models are discussed in Section 2.2.1.

End-to-end models, on the other hand, are trained on graphemes or sub-word units such as BPE tokens. The transcriptions are transformed into tokens in a tokenisation process. For e2e systems, the tokenisation is followed by a normalisation step. Usually, the annotations of speech data have some unstandardised transcriptions which might cause some confusion for model training. For example, annotation of spoken numbers in multiple forms such as in words and numbers for different instances might adversely affect a learning process. Some other examples include inconsistent handling of contractions, acronyms and cases. In a normalisation step, transcriptions of speech data are normalised using standardised notation and removing punctuation marks etc. The transcriptions are tokenised after the normalisation step. Tokens can be a language's

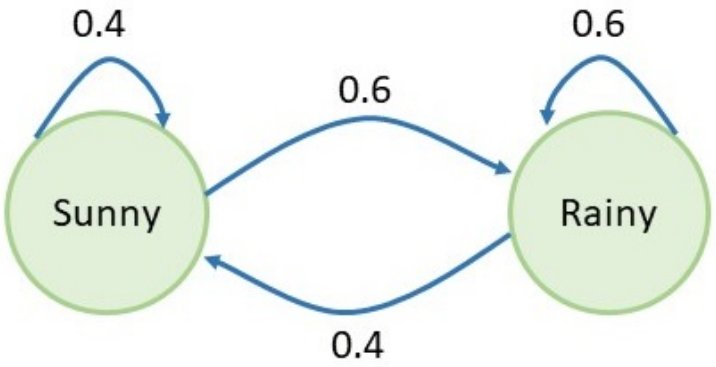


FIGURE 2.3: Example of a Markov chain with two possible states of the weather on a day: sunny and rainy. The arrows show the probability of transition of weather states.

character (graphemes) or sub-word units. Similar to HMMs, some additional tokens are also used such as space, blank symbol, unknown symbol, start and end of sentence.

2.2 Acoustic modelling

Over the past several decades, automatic speech recognition systems have experienced significant advancements, evolving from GMM-HMM models (Rabiner and Juang, 1986), several hybrid DNN-HMM models (Bourlard and Morgan, 1994, Vinyals et al., 2012), e2e speech recognition systems (Graves and Jaitly, 2014) to recent self-supervised models like wav2vec (Baevski et al., 2020), Hubert (Hsu et al., 2021b) and Whisper (Radford et al., 2022) are a few to name. This section discusses the acoustic modelling techniques that are used or relevant to this work. Widely used training and decoding methods of HMM and e2e models are described.

2.2.1 Hidden Markov models

Hidden Markov models have widely been used as machine learning models for speech and language processing (Hinton et al., 2012, Zhu et al., 2005). HMMs are fundamentally based on Markov chains which are a special case of weighted finite-state automaton. A Markov chain or observable Markov model is a stochastic model which informs about the probabilities of a sequence of random variables. These random variables are usually referred to as *states* which can take values of any random process. A principle assumption of Markov chains is that the probability of a given state depends only on the last state and the states before the previous states do not have any impact on the prediction. For a set of states $S = \{s_1, s_2, \cdots, s_{\mathcal{S}}\}$, a Markov chain embodies the Markov assumption that $P(s_i = a|s_{i-1}, s_{i-2}, \cdots, s_1) = P(s_i|s_{i-1})$. A very simple weather prediction Markov chain is shown in Figure 2.3. The states show the observed weather on a day i.e. sunny or rainy. The arrows (in blue colour) show the probability of a weather state given the previous state. For example, the probability of a day being sunny is 0.4 given the previous day had also been sunny and 0.6 otherwise. The prediction probability only

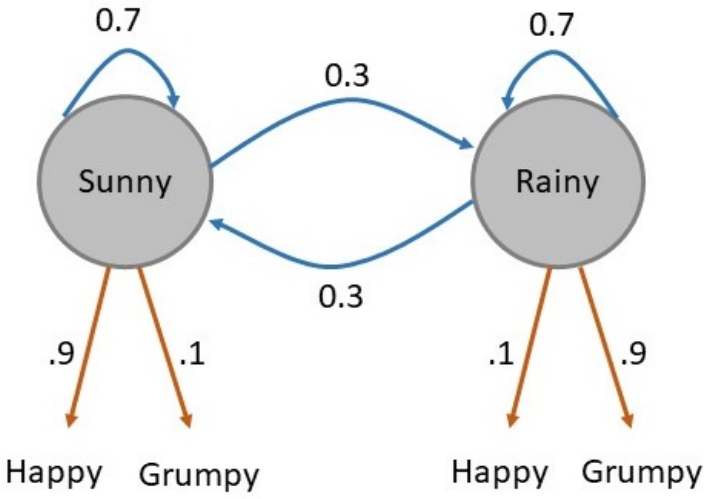


FIGURE 2.4: Example of a hidden Markov chain with two possible states (hidden) of the weather on a day: sunny and rainy. Bob's mood is an observable information which could be happy or grumpy. The arrows between states show the state transition probabilities and the arrows towards mood possibilities (happy and grumpy) show the probability of the mood given the state information.

depends on the last day's weather (previous state) irrespective of observed weather in all the previous states.

Markov chains are useful for predicting the probability of a sequence of observable events. However, in many cases, the possible events of interest might not be directly observable. Let's modify the earlier simple example of weather prediction (Figure 2.3) with hidden information. Alice and Bob live in two different cities and thus Alice cannot observe the weather at Bob's place. However, she knows that Bob's mood heavily depends on the weather in his city. When Bob calls Alice and tells her that he is happy (*observable state*), Alice knows that it is a sunny day in Bob's city (*hidden state*) and vice versa. The probabilities of weather transition in Bob's city and his mood given a weather state are shown in Figure 2.4. From her knowledge, Alice knows that the probability of Bob's being happy is 0.9 given it is a sunny day and 0.1 in the case of a rainy day. In language processing, one example of such a case is Parts-of-Speech (POS) tagging where POS tags are not observed in the world. Rather sequence of words is observed and POS tags are inferred from the observed sequence of words. Similarly, in speech recognition, only the acoustic events (speech signal) are observable to infer "hidden" events (tokens which could be phonemes or graphemes etc). For such problems, Hidden Markov models are used rather than Markov models. In HMMs, the state of the system being modelled is not directly observable rather each state emits observations (as shown for weather prediction example in Figure 2.4).

HMMs have widely been used for building speech recognition systems (Bourlard and Morgan, 1994, Gemello et al., 2007). An HMM model is trained for each token and each

HMM model is a finite state machine. At each time step, the finite state machine is at some hidden state (Li, 2022b). Each HMM has two kinds of states i.e. emitting and non-emitting states. Non-emitting states are useful for joining HMM models of all the tokens together. Let a speech utterance with T observations $O = \{\boldsymbol{o_1}, \boldsymbol{o_2}, \cdots, \boldsymbol{o}_T\}$ and an HMM model of five states $S = \{s_1, s_2, s_3, s_4, s_5\}$ where s_1 and s_5 are non-emitting states. The transition probability from state i to j is given as a transition probability matrix A for which;

$$a_{ij} = P(s_t = j | s_{t-1} = i) \tag{2.3}$$

where $\sum_{j=1}^{\mathcal{S}} a_{ij} = 1$ and $\mathcal{S}$ is the total number of states. The probability of an observation o_t at state i is given as

$$b_i(\boldsymbol{o}_t) = p(\boldsymbol{o}_t | s_t = i) \tag{2.4}$$

Equation 2.4 assumes that the probability of an observation $\boldsymbol{o}_t$ depends only on the state it is being emitted from ($s_t = i$) irrespective of other states and observations. It is the second independence assumption of HMM models (the first one comes from the Markov chain). The emission probability density function at a state is usually modelled by Gaussian Mixture Models (GMMs). A simple approximation could be modelling using a single Gaussian of the form;

$$b_i(\boldsymbol{o}_t) = \mathcal{N}(\boldsymbol{o}_t; \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i) \tag{2.5}$$

$$= \frac{1}{\sqrt{(2\pi)^D |\boldsymbol{\Sigma}_i|}} \exp\left(-\frac{1}{2}(\boldsymbol{o}_t - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{o}_t - \boldsymbol{\mu}_i)\right) \tag{2.6}$$

Practically, a single Gaussian is a poor approximation compared to using GMMs (Li, 2022a). In case of GMMs, $b_i(\boldsymbol{o}_t)$ becomes;

$$b_i(\boldsymbol{o}_t) = \sum_{m=1}^{M_i} c_{im} \mathcal{N}(\boldsymbol{o}_t; \boldsymbol{\mu}_{im}, \boldsymbol{\sigma}_{im}) \tag{2.7}$$

where M_i is the total number of Gaussian components in a Gaussian mixture and c_{im} is the prior of the Gaussian m at i^{th} state.

Along with A (matrix of transition probabilities) and B (matrix of emission probabilities), HMMs have an additional parameter $\pi = \{\pi_1, \pi_2, \cdots, \pi_{\mathcal{S}}\}$ as a probability distribution of states being an initial state where $\mathcal{S}$ is the total number of states and $\sum_{i=1}^{\mathcal{S}} \pi_i = 1$. So, the notation λ with parameters (A, B, π) is used to represent an HMM model.

Inspired by an HMM tutorial of Rabiner (Rabiner and Juang, 1986), HMMs are characterised by three problems;

1. **Likelihood:** Given an HMM model $\lambda(A, B, \pi)$ and a sequence of observations O, how to efficiently compute the probability of observation sequence i.e. $P(O|\lambda)$

2. **Training:** How to find parameters of HMM model λ to maximise the probability of observation sequence $P(O|\lambda)$.

3. **Decoding:** Given a sequence of observations O and an HMM model λ, how to find the optimal sequence of hidden states S.

In the following section, approaches to solving these problems are discussed.

2.2.1.1 Likelihood calculation

In the case of a Markov chain where the state sequence is known, the probability of a given observation sequence $P(O|S)$ can be calculated easily. However, in the case of Hidden Markov models, the state hidden state sequence is not known and the likelihood of the observation sequence is not straightforward to compute. But if the joint probability of the observation sequence and all possible sequence states are known, the likelihood of the observation is simply the sum of probabilities of all possible state combinations i.e.

$$P(O) = \sum_{S'} P(O, S) \tag{2.8}$$

where S' is all possible combinations of hidden state sequences. The Equation 2.8 can be re-written as;

$$P(O) = \sum_{S'} P(O|S)P(S) \tag{2.9}$$

where $P(O|S)$ is the probability of observation sequence given a sequence of the hidden states which is the product of observation probabilities at all time steps given the state.

$$P(O|S) = \prod_{i=1}^{T} p(o_i|s_i) \tag{2.10}$$

And from HMM's second assumption i.e. the probability of a state depends only on the previous state, the probability of a state sequence can be calculated as;

$$P(S) = \prod_{i=1}^{T} p(s_i|s_{i-1}) \tag{2.11}$$

So, Equation 2.9 can be computed using Equation 2.10 and Equation 2.11. However, computing probabilities over all possible hidden state sequences is very expensive as the number of possible sequences grows exponentially (N^T; where N is the total number of states and T is the number of time stamps).

To make the probabilities computation efficient, the **forward algorithm** is used. The forward algorithm reduces the computation complexity to (N^2T) and uses intermediate values for the following calculations. The approach is the same as *dynamic programming*

i.e. values at intermediate steps are stored and then used for computing values in the next steps. In summary, the forward algorithm makes the use of fact that the likelihood of an observation sequence up to timestamp t at state j is equal to the sum of probabilities of all possible paths up to the previous timestamp $(t-1)$ at state i $(\alpha_{t-1}(i))$ times transition probability from previous state i to j (a_{ij}) and the emission probability of observation at time t and state j $(o_t(j))$. This $\alpha_t(j)$ is referred as the "**forward-probability**".

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_t(o_j) \tag{2.12}$$

For the first time step $t = 1$, $\alpha_1(j) = \pi_j b_1(o_j)$ where π_j is the initial probability of state j and a parameter of HMM model λ. Then, probabilities for all possible paths are calculated using Equation 2.12 for each time step t propagating towards the last time step T. Summing all the final probabilities gives the likelihood of the observation sequence $P(O|\lambda)$.

2.2.1.2 Training

In the last section, an estimation of the likelihood of a sequence of observations given an HMM model $\lambda(A, B, \pi)$ has been shown. Now, the question is how do we know the values of A, B and π. Or formally, *how to estimate the initial (π), transition (A) and emission probabilities (B) of an HMM model*?

Before going into the training process, it is important to define another probability term called "**backward-probability**". For the observation $\boldsymbol{o}_t$ at time t and hidden state i, the backward-probability is defined as the likelihood of observing the sequence $o_{t+1}, \cdots, o_T$. It is computed the same as the forward-probability and given as;

$$\beta_t(i) = \sum_{i=1}^{N} \beta_{t+1}(j) a_{ij} b_{t+1}(o_j) \tag{2.13}$$

For training an HMM model, three parameters are needed to be estimated i.e. transition (A) and emission (B) probabilities and initial (π) probabilities.

Transition probabilities If the hidden state sequence is known for a given observation sequence, transition probability from state i to state j can simply be estimated by dividing the total number of transitions from i to j by the number of transitions from state i.

$$\hat{a}_{ij} = \frac{\text{expected \# of transitions from state } i \text{ to } j}{\text{expected \# of transitions from state } i} \tag{2.14}$$

In practice, however, the hidden state sequence is not known and transition probabilities can not be calculated directly. Formally, the numerator in Equation 2.14 can be written as $P(s_t = i, s_{t+1} = j|O, \lambda)$ where O and λ are the observation sequence and HMM model

respectively. It might be confusing to see the HMM model in the conditional probability when it is being trained. HMM models are trained in a back-and-forth backward-forward algorithm which keeps updating the parameters of the HMM model after each iteration. For the first iteration, these parameters are randomly initiated. Assuming that HMM model is given, the joint probability $P(s_t = i, s_{t+1} = j, O)$ can be calculated as;

$$P(s_t = i, s_{t+1} = j, O) = \alpha_t(i) \cdot a_{ij} b_j(o_{t+1}) \cdot \beta_{t+1}(j) \tag{2.15}$$

where $\alpha_t(i)$ is the likelihood of the observation sequence up to t time step at state i and $\beta_{t+1}(j)$ is the observation sequence likelihood after time step $t+1$ given it is at state j at $t+1$. As shown earlier, the transition cost from state i to state j is the product of transition probability from i to j (a_{ij}) and emission probability at state j ($o_j(o_{t+1})$). So, the joint probability $P(s_t = i, s_{t+1} = j, O)$ would be the product of all these probabilities. Since $P(s_t = i, s_{t+1} = j|O, \lambda)$ can be written as

$$P(s_t = i, s_{t+1} = j|O, \lambda) = \mathcal{E}_t(i, j) = \frac{P(s_t = i, s_{t+1} = j, O|\lambda)}{P(O|\lambda)} \tag{2.16}$$

$$\mathcal{E}_t(i, j) = \frac{\alpha_t(i) \cdot a_{ij} b_j(o_{t+1}) \cdot \beta_{t+1}(j)}{\sum_{i=1}^{N} \alpha_t(i) \cdot \beta_t(i)} \tag{2.17}$$

The sequence observation probability is replaced with the product of backward and forward probability for all the states at a given time step t. Equation 2.16 gives the probability of transitions from state i to j at time t. For the total expected number of transitions from state i to j (the numerator in Equation 2.14), $\mathcal{E}_t(i, j)$ is needed to be summed over all the time steps. Similarly, the denominator of Equation 2.14) would be not only the sum over all the time steps but also overall different j states for each time step. So, the Equation 2.16 can be written as;

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \mathcal{E}_t(i, j)}{\sum_{t=1}^{T-1} \sum_{k=1}^{N} \mathcal{E}_t(i, k)} \tag{2.18}$$

Using Equation 2.18, the transition probabilities can be trained in two steps (forward and backwards). In the forward step, the likelihood probabilities (α and β) are estimated. The transition probabilities of the last step are used for these calculations. In the backward step, these transition probabilities are updated.

Emission probabilities Similar to the transition probabilities, emission probabilities are learned using the backward-forward algorithm. The emission probability of a state j can be estimated as;

$$\hat{b}_j = \frac{\text{expected \# of times in state } j \text{ and an emitting observation } o}{\text{expected \# of times in state } j} \tag{2.19}$$

It is important to note that this definition provides a simplified explanation. Typically, $\hat{b}_j$ is modelled using GMMs to capture the distribution of observations more accurately (as described earlier in Equation 2.7). However, for the sake of clarity and simplicity, the current explanation does not incorporate GMMs and will be introduced later in Equation 2.23. As the term $P(s_t = j|O, \lambda)$ is needed to be calculated for the numerator, and the same steps are followed which have been taken for updating transition probabilities.

$$P(s_t = j|O, \lambda) = \gamma_t(j) = \frac{P(s_t = j, O|\lambda)}{P(O|\lambda)} \tag{2.20}$$

The joint probability $P(s_t = j, O)$ in the numerator can be seen as the product of forward and backward probabilities at state j while the denominator has already been formulated in the last section.

$$\gamma_t(j) = \frac{\alpha_t(j)\beta_t(j)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \tag{2.21}$$

As Equation 2.21 gives the probability of being in state j at time t, it needs to be summed over all the time steps to calculate the total probability of being in state j. However, for the probability of being in state j and emitting observation o (the numerator of Equation 2.19), the summation is done over only time steps which omit observation o. So, the Equation 2.19 is used to update omission probabilities in the backward step as follows;

$$\hat{b}_j = \frac{\sum_{t \in T'} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \tag{2.22}$$

where T' is the set of time steps which omit observation o. However, an assumption here is that the observations are discrete from a finite set and a discrete probability density can be used within each state of the model.

However, observations are usually continuous signals in speech processing (MFCCs etc). So, a probability density function is used to model the emission probabilities ensuring that its parameters can be estimated in a consistent way. A finite mixture of the following form is most widely used;

$$\hat{b}_j(\boldsymbol{O}) = \sum_{m=1}^{M} c_{jm} \mathcal{N}(\boldsymbol{O}, \boldsymbol{\mu}_{jm}, \boldsymbol{\sigma}_{jm}) \tag{2.23}$$

where c_{jm} is the mixture coefficient for the m^{th} mixture in j state and $\mathcal{N}$ represents the m^{th} Gaussian distribution in j state with mean vector $\boldsymbol{\mu_{jm}}$ and covariance matrix $\boldsymbol{\sigma_{jm}}$. Though the density function could be any log-concave or elliptically symmetric density, the emission probabilities are widely modelled by Gaussian mixture models. So for continuous observations, Equation 2.21 can be modified for the joint probability of

being in j state at time t with k^{th} mixture of the GMMs accounting for O_t as following;

$$\gamma_t(j,k) = \frac{\alpha_t(j)\beta_t(j)}{\sum_{i=1}^{N}\alpha_t(i)\beta_t(i)} \cdot \frac{c_{jk}\mathcal{N}(\boldsymbol{o}_t;\boldsymbol{\mu}_{jk},\boldsymbol{\sigma}_{jk})}{\sum_{m=1}^{M}c_{jm}\mathcal{N}(\boldsymbol{o}_t;\boldsymbol{\mu}_{jm},\boldsymbol{\sigma}_{jm})} \tag{2.24}$$

where c_{jm} is the mixture coefficient for the m^{th} mixture in j^{th} state and $\mathcal{N}(\boldsymbol{\mu}_{jm},\boldsymbol{\sigma}_{jm})$ is a Gaussian probability density function with mean $\boldsymbol{\mu}_{jm}$ and variance $\boldsymbol{\sigma}_{jm}$. The parameters c_{jk} is re-estimated as;

$$\hat{c}_{jk} = \frac{\sum_{t=1}^{T}\gamma_t(j,k)}{\sum_{t=1}^{T}\sum_{k=1}^{M}\gamma_t(j,k)} \tag{2.25}$$

where the numerator gives the expected number of times being in state j using the k^{th} mixture and the denominator means the expected number of times in the state j. Similarly, the parameters $\boldsymbol{\mu}_{jm}$ and $\boldsymbol{\sigma}_{jm}$ are updated as;

$$\hat{\boldsymbol{\mu}}_{jk} = \frac{\sum_{t=1}^{T}\gamma_t(j,k)\cdot\boldsymbol{O}_t}{\sum_{t=1}^{T}\gamma_t(j,k)} \tag{2.26}$$

$$\hat{\boldsymbol{\sigma}}_{jk} = \frac{\sum_{t=1}^{T}\gamma_t(j,k)\cdot(\boldsymbol{O}_t-\boldsymbol{\mu}_{jk})(\boldsymbol{O}_t-\boldsymbol{\mu}_{jk})'}{\sum_{t=1}^{T}\gamma_t(j,k)} \tag{2.27}$$

where prime denotes the transpose of a matrix.

2.2.1.3 Decoding

The last yet important part is to employ a trained HMM model (λ) to output the most probable hidden state sequence (S) given a sequence of observations ($O = o_1, o_2, \cdots, o_T$). The Viterbi algorithm proposed by Andrew Viterbi (Viterbi, 1967) is used in the decoding. The process of the Viterbi algorithm is pretty similar to computing forward probabilities in the Section 2.2.1.2. The only difference is that the Viterbi algorithm does not sum all previous paths rather only keeps the most probable one for each state. Formally, the likelihood of state j at time t is given as;

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i) * a_{ij}b_j(o_t) \tag{2.28}$$

The most likely hidden state sequence is the one which gives the highest values of v at the last time step ($v_{t=T}$). To output the most probable hidden state sequence, the most probable state sequence is also stored at each state to backtrack after the final output.

$$s'_t(j) = \arg\max_{i=1}^{N} v_{t-1}(i) * a_{ij}b_j(o_t) \tag{2.29}$$

At the end of the algorithm, $\max_{j=1}^{N} v_T(j)$ is the likelihood of the most probable path and $\arg\max_{j=1}^{N} v_T(j)$ gives the most probable hidden state sequence.

2.2.2 Hybrid DNN-HMM models

As discussed in the last section, the emission probabilities of observations are modelled by GMMs. In hybrid DNN-HMM models, the problem is regarded as discriminative rather than generative. A DNN model is trained to predict probability $p(s|o_t)$ of a state s given an observation o_t at time t. It is contrary to GMMs which estimate the probability of an observation given a state $p(o_t|s)$. In the case of DNNs, the pseudo-log-likelihood of state s given observation o_t is used by the recogniser;

$$\log p(o_t|s) = \log p(s|o_t) - \log P(s)$$

where $P(s)$ is the prior probability of state s which is calculated from the training data.

The number of output classes of a DNN is theoretically equal to the total number of possible HMM states which is $Q^n \times S$ where Q is the total number of unique tokens (phonemes or graphemes etc.), n is the context width (monophone and biphone etc.) and S is the number of states per HMM model. However, the number of states increases significantly by changing the context width or number of states of the HMM models. In practice, various HMM states are clustered to train a DNN model. Given an input observation, a trained DNN model predicts the posterior probabilities of the states. These probabilities are used as emission probabilities of an HMM model. The rest of the process remains the same.

2.2.2.1 Training

Training of the neural network of hybrid DNN-HMM models usually requires a pre-trained GMM-HMM model. Transcriptions are force-aligned using a pre-trained GMM-HMM model. A neural network is trained using speech features (e.g. MFCCs) as input and forced-aligned transcriptions as the targets for each frame. These neural networks can be trained using different objective functions such as Cross-Entropy (CE), Maximum Mutual Information (MMI) (Bahl et al., 1986), Minimum Phone Error (MPE) (Povey, 2003) and state-level minimum Bayes risk (sMBR) (Kaiser et al., 2002) etc. However, only the cross-entropy and maximum mutual information objective functions are discussed here as most of the work around hybrid DNN-HMM models in this thesis uses only these two functions.

Cross-entropy objective function Cross-entropy is a widely used objective for binary or multi-class classification tasks which computes the negative log posteriors.

$$\mathcal{F}_{CE} = -\sum_{T} \log \hat{y}_t(s') \tag{2.30}$$

where T is the number of frames in an utterance and $\hat{y}_t(s')$ is the probability of state s' which is the reference state label at time t. Cross-entropy, however, is a frame-level loss function and does not take context into account whereas speech is a continuous signal

and the output at a frame might fairly depend on its context.

Maximum mutual information objective function Maximum mutual information maximises the mutual information between observations and output word sequences (Veselý et al., 2013). Given a sequence of observation $\boldsymbol{O} = \{\boldsymbol{o}_1, \boldsymbol{o}_2, \cdots, \boldsymbol{o}_T\}$ and $\mathcal{W}$ as the reference word sequence for an utterance, the MMI criterion is;

$$\mathcal{F}_{MMI} = \log \frac{P(\boldsymbol{O}|\boldsymbol{S'}) \cdot P(\mathcal{W})}{\sum_W P(\boldsymbol{O}|\boldsymbol{S}) \cdot P(W)} \tag{2.31}$$

where the denominator is a summation over all possible word sequences in the decoded speech lattice for the utterance and $\boldsymbol{S}$ is the sequence of states associated with the sequence of words (W). $\boldsymbol{S'}$ is the state sequence associated with the reference word sequence $\mathcal{W}$. As MMI is a sequence-level objective function, a DNN trained with this objective proves to perform better than CE-trained DNN models. However, the requirement of summation over all possible word sequences in the lattice in the denominator makes it computationally very expensive. So, **Lattice-Free Maximum Mutual Information (LF-MMI)** has been introduced (Povey et al., 2016) where a phoneme-based graph is used rather than a word-based graph which reduces the possible combinations and thus the computational complexity. Furthermore, the frame rate in the decoding is reduced three times which further improves the computation speed.

2.2.3 Kullback-Leibler HMM (KL-HMM) Models

Kullback-Leibler HMMs (Aradilla et al., 2007) are a type of hybrid HMM model proposed for acoustic modelling. They are described here briefly. As mentioned in the previous section, the output posterior distributions of a DNN are used as emission probabilities of the tied HMM states in a hybrid DNN-HMM model. Thus, the outputs of a DNN are tied to the HMM states, which typically model context-dependent sub-word units such as triphones. One limitation of hybrid DNN-HMM models is that if we want to consider a different type of sub-word unit, such as biphones, the DNN structure needs to be modified. This limitation is addressed by KL-HMMs (Aradilla et al., 2007), in which the HMM states are not tied to specific DNN outputs but are characterised by their target posterior distributions. Therefore, the HMM states can represent different kinds of sub-word units even if the DNN output remains unchanged.

In KL-HMM, the posterior probabilities of acoustic states (e.g., triphones) are estimated using a DNN and used directly as feature observations. Let z_t denote the acoustic state posterior feature vector estimated at time frame t:

$$\begin{aligned} \boldsymbol{z}_t &= [z_t^1, \ldots, z_t^d, \ldots, z_t^D] \\ &= [P(a_1 \mid x_t), \ldots, P(a_d \mid x_t), \ldots, P(a_D \mid x_t)] \end{aligned}$$

where x_t is the acoustic feature at time frame t, $\{a_1, a_2, \ldots, a_D\}$ is the set of acoustic states, D is the number of acoustic states, and $P(a_d \mid x_t)$ denotes the posterior

probability of acoustic state a_d given x_t.

Let $S = \{s_1, \ldots, s_i, \ldots, s_{\mathcal{S}}\}$ be the set of HMM states, which can be different context-dependent sub-word units (from those of DNN outputs). Each HMM state s_i in the KL-HMM system is parameterised by a categorical distribution:

$$y_i = [y_i^1, \ldots, y_i^d, \ldots, y_i^D]$$

where $0 \leq y_i^d \leq 1$, $\sum_{d=1}^{D} y_i^d = 1$, and $y_i^d = p(a_d \mid s_i)$. Therefore, KL-HMM can be seen as a probabilistic modelling approach where the relationship between acoustic states modelled by the DNN and the output states modelled by KL-HMM is probabilistic.

The local score at each HMM state is the KL divergence between the acoustic state posterior feature and the state distribution:

$$J(y_i, z_t) = \sum_{d=1}^{D} y_i^d \log\left(\frac{y_i^d}{z_t^d}\right)$$

Training: In the KL-HMM approach, it is assumed that an already trained DNN (similar to hybrid DNN-HMM models) is available. The cost function of a KL-HMM model is:

$$\mathcal{J} = \min \sum_{t=1}^{T} J(y_i, z_t) \tag{2.32}$$

where y_i is the target posterior distribution that characterises the i-th HMM state. Since z_t is extracted from the pre-trained DNN model, the only parameters left to estimate are the HMM states' target posterior distributions $\{y_i\}_{i=1}^{\mathcal{S}}$. The KL-HMM parameters $\{y_i\}_{i=1}^{\mathcal{S}}$ are estimated using the Viterbi expectation-maximisation (Viterbi-EM) algorithm, which minimises the cost function in (2.32). During testing, decoding is performed using a standard Viterbi decoder, with the log-likelihood-based score in the Viterbi decoding replaced by the KL-divergence-based local score $-J(y_i, z_t)$.

2.2.4 Neural networks for speech recognition

Speech recognition has a long history of utilising neural networks, often used to model emission probabilities of HMMs (Bourlard and Morgan, 1994, Zhu et al., 2005). Basic Feed-Forward Neural Network (FFN) to Recurrent Neural Network (RNN) have been used for hybrid DNN-HMM models for speech recognition (Bourlard and Morgan, 1994, Vinyals et al., 2012). The advanced works explore using neural networks for end-to-end speech recognition (Graves et al., 2006) rather than combining them with HMMs. Previous works have explored various types and architectures of feed-forward, recurrent, convolutional and transformer models for the task of speech recognition. This section briefly describes the neural networks that are used for speech modelling, their training, activation and objective functions which are widely used to train them. Neural networks are usually trained using back-propagation (Rumelhart et al., 1986) that is briefly

described in the following section.

2.2.4.1 Feed-Forward Neural Networks

Feed-forward neural networks are fundamental artificial neural networks where information travels in only one direction i.e. from the input layer to the output layer through one or more hidden layers. In their compact form, feed-forward neural networks do not contain cycles or loops. Each layer in a feed-forward network only receives input from the previous layer and passes output to the next layer, creating a straightforward, acyclic structure. A simple FFN with only one hidden layer is shown in Figure 2.5. This is also referred to as Multi-Layer Perceptron (MLP), though an MLP can have more than one hidden layer. The first and last layers (shaded in light yellow and green colours) are the input and out layers respectively. The input data (or feature vector) $\boldsymbol{X} \in \mathbb{R}^{1\times T}$ is fed through the input layer. The output of the first hidden layer is computed as;

$$\boldsymbol{H_1} = \sigma(\boldsymbol{X}\boldsymbol{W_1} + \boldsymbol{b_1})$$

where $\boldsymbol{W_1}$ is a weight matrix ($\boldsymbol{W_1} \in \mathbb{R}^{T\times N}$ where N is the number of neurons in the first layer), $\boldsymbol{b_1}$ is a bias vector ($\boldsymbol{b} \in \mathbb{R}^{1\times N}$) and $\sigma(\cdot)$ is an activation function. A bias vector acts to add an offset in the weighted sum of input values. It lets the neurons get some value even if the weighted sum of the inputs is zero which implies that it allows the neurons to have some impact even if the input impact is zero. The output of the

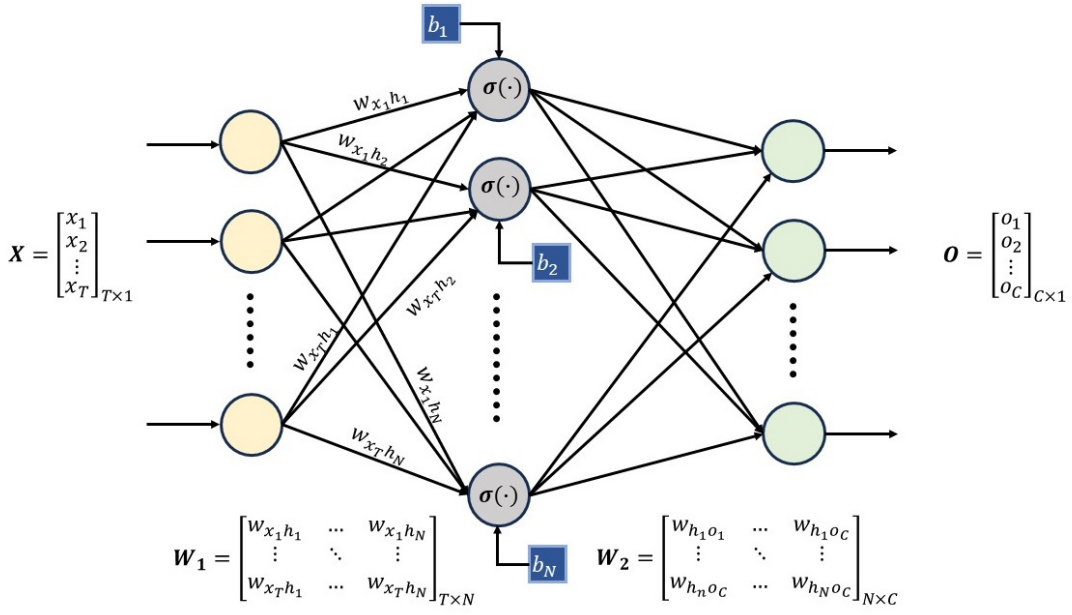


FIGURE 2.5: An example feed-forward neural network with 3 layers: input, hidden, and output. The input layer receives the input data, which is then processed by the hidden layers. The output layer produces the final output of the network. Hidden parameters are computed by multiplying input features X with a weight matrix W_1 and adding a bias B_1. An activation function $\sigma(\cdot)$ is applied to all output parameters. The same operation is carried out on hidden layer parameters to get outputs O using a different weights matrix W_2 and bias vector B_2.

next hidden layers can be computed in the same way. However, each layer would have different weight matrices and bias vectors. Generally, the output of a j^{th} hidden layer can be described as;

$$\boldsymbol{H_j} = \sigma(\boldsymbol{H}_{j-1} \cdot \boldsymbol{W_j} + \boldsymbol{b_j})$$

Each layer can have a different number of neurons in it. The dimension of the last (output) layer is chosen depending on the task. For example, for a binary classification task, the final layer would have only two neurons. In the case of a phoneme-based speech recogniser, the output layer might have the same number of neurons as the number of phonemes.

For training of a neural network, all the weights and bias matrices are usually randomly initialised which are then learnt during training. Neural networks are usually trained using the back-propagation (Rumelhart et al., 1986) method where learning is done by back-propagating the errors.

Training Given an input example, the output of a neural network is computed as outlined above. The error between the neural network output and the ground truth is computed. The error computation function is usually selected based on the task. For example, for a simple classification task, a cross-entropy function is used and Mean-Squared Error (MSE) is computed for a regression task. These functions are also referred to as loss functions or objective functions. Various objective functions used to train speech models are discussed in the following sections. Having $\hat{Y}$ output from a neural network and the ground-truth output Y for an input X, the error is calculated using a function $g(\cdot)$.

$$E = g(Y, \hat{Y})$$

This error is back-propagated to compute the gradient of the loss with respect to the weights and biases at each layer. In machine learning, the gradient refers to a vector containing partial derivatives of a scalar-valued function with respect to its input variables. So, $\frac{\partial E}{\partial X} = \frac{\partial}{\partial X} g(Y, \hat{Y})$ are needed to be calculated. However, $\hat{Y}$ is the final output of the neural network through several non-linear hidden layers and thus cannot be differentiated with respect to (w.r.t) input X straightforwardly. So, the chain rule is used to compute the gradient of the loss;

$$\frac{d}{dx}[f(g(x))] = f'(g(x)) \cdot g'(x)$$

where $g'(x)$ is the derivative of g w.r.t x and $f'(g(x))$ is the derivative of function f w.r.t its input evaluated at $g(x)$. The chain rule can be extended to any number of functions and thus used to compute the gradient of the loss w.r.t the weights and biases of the layers of a neural network. After calculating the gradient, the weights and biases are

updated as;

$$\boldsymbol{W}_j^* = \boldsymbol{W}_j - \eta \cdot \frac{\partial E}{\partial \boldsymbol{W}_j}$$

where η is a hyper-parameter known as **learning rate** and defines the size of the step to be taken to optimise the objective function. Usually, the weights are not updated for each input example for the sake of training stability rather the gradients are calculated and back-propagated after a forward pass on a batch of input examples (referred to as **batch size**).

Time-Delay Neural Network A Time-Delayed Neural Network (TDNN) (Waibel et al., 1989) is an important type of feed-forward neural networks. It introduces the concept of time delays in the connections between neurons, allowing the network to capture temporal dependencies in the input data. This makes TDNNs well-suited for applications in sequential or time-series data processing such as speech and natural language processing. In conventional deep neural networks, the initial layer learns an affine transform of the entire temporal context. In TDNN architecture, however, the initial layer learns transforms in a narrow context and the higher layers learn hidden activations from the wider context. In TDNN, each layer operates at a different temporal context which increases towards the deeper layers. A simple TDNN architecture is shown in Figure 2.6.

In Figure 2.6, a TDNN with two hidden layers is shown. Each neuron in the first hidden layer is computed using features at the current time frame and one left and right frame. For the example here, only one left and right contextual frame are used though it could be any context. For the second layer, a context of two left and right frames is used to

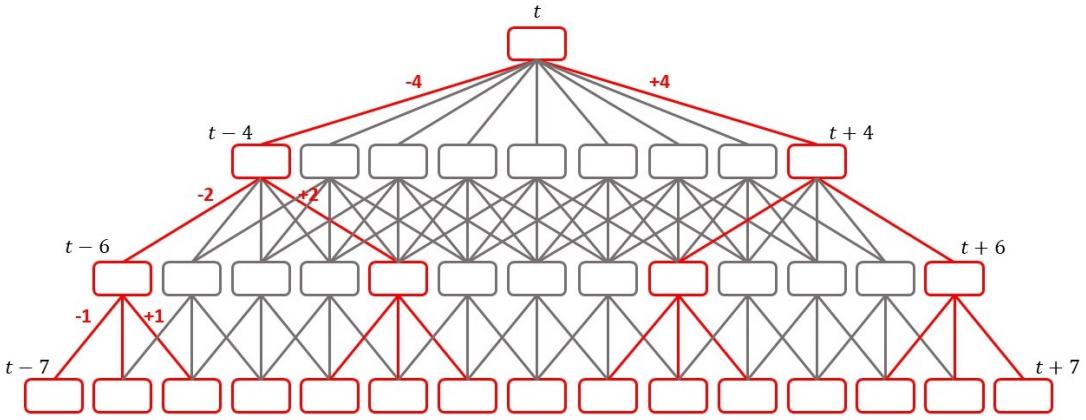


FIGURE 2.6: An example TDNN model with (red and grey lines) and without sub-sampling (only red lines) approach is shown for a given time step t. Contextual frames are considered at each time step (context width is 7 in this figure). For computing parameters of the second layer, features of one left and right contextual frame are also used. This temporal context increases towards the higher layers i.e. last layer uses 4 left and right frames. In the sub-sampling approach, only the extreme neurons of each frame from the last layer are used to compute the parameters of the next layer.

TABLE 2.1: Temporal context captured by different layers of TDNN of Figure 2.6. The temporal context considered by different layers is shown for TDNN without sub-sampling (Waibel et al., 1989) and with sub-sampling (Peddinti et al., 2015) approaches.

Layer	Input Context	
	w/o sub-sampling	w/ sub-sampling
Layer 1	[-1,1]	[-1,1]
Layer 2	[-2,2]	{-2,2}
Output	[-4,4]	{-4,4}

compute the value of each neuron. The final layer outputs the class probabilities by carrying out the weighted sum of all the neurons of the previous layer. As it is evident that a lot of computation is needed for a TDNN, a variant of it with sub-sampling has also been proposed (Peddinti et al., 2015) (only red lines in the Figure 2.6). With sub-sampling, rather than computing neurons of second layers by taking all five frames into account (current time frame and two left and right contextual frames), only extreme neurons of previous layers are used for computation. It reduces the computation of a TDNN significantly. The input context without sub-sampling (red and grey lines in Figure 2.6) and with sub-sampling (only red lines) is shown in Table 2.1.

As the transformations within the TDNN architecture are tied across different time steps, they can be seen as a precursor to Convolutional Neural Network (CNN). In the back-propagation process of a TDNN, the tying of these transformations results in the gradient accumulation over all the time steps of the input temporal context for the lower layers of the network. Consequently, this makes the lower layers of a TDNN learn translation-invariant feature transformations (Waibel et al., 1989).

The time delays in time-delay neural networks are typically fixed and predefined. Each neuron in each layer of a TDNN processes information from a specific time window. This limits the ability of TDNNs to capture long-term temporal dependencies. Sequence (or recurrent) neural models such as Gated Recurrent Unit (GRU) or Long Short-Term Memory (LSTM) are alternative architectures that can handle long-range dependencies in a better way.

In the following two sections, activation and objective functions are described which are an important part of all kind of neural networks and their training process. Their use is not limited to only the feed-forward neural networks.

Activation functions Activation functions play a crucial part in the training of neural networks by introducing non-linearities into the computation. Without using activation functions, a neural network would be a linear model regardless of the number of layers etc. A lot of problems are much more complex than to be solved by a linear model. The introduction of activation functions makes a model non-linear and able to learn complex patterns and relationships in the data. Furthermore, these functions also help to stabilise the training of neural networks by preventing neurons from saturating. Saturation occurs

when the output of a neuron is pushed to extreme values, making it difficult for the network to learn effectively.

Various activation functions are used based on the complexity of the network and limitations of the activation functions etc. Sigmoid, hyperbolic tan (*tanh*), Rectified Linear Unit (ReLU) (Nair and Hinton, 2010), Leaky Relu (Xu et al., 2015a) and softmax are a few to name. However, some of the most widely used are described here and plotted in Figure 2.7 along with their derivatives.

- **Sigmoid:** The sigmoid function takes a real value input and produces outputs within the interval of 0 to 1. Mathematically, it is described as;

$$\sigma_{sigmoid}(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{x}}}$$

So, as the input gets smaller (more negative) the output moves close to 0. The sigmoid function outputs values closer to 1 as the input gets larger and larger. This is a widely used activation function as it is differentiable and provides a smooth gradient (no jumps in the output of its derivative).

$$\frac{\partial \sigma_{sigmoid}(\boldsymbol{x})}{\partial \boldsymbol{x}} = \sigma_{sigmoid}(\boldsymbol{x}) \cdot (1 - \sigma_{sigmoid}(\boldsymbol{x}))$$

However, as the output range of the sigmoid function is from 0 to 1, it pulls all the inputs to a very small range which makes it impossible for the following layers to discriminate among the diverse input examples (very negative and very positive etc). Furthermore, as evident from the Figure 2.7 the gradient of the sigmoid function has very small output values for the inputs outside the range of $[-3, 3]$. As the gradient value becomes very small, a neural network suffers from **vanishing gradient problem** and is not able to learn.

- **Hyperbolic tan (*tanh*):** The *tanh* function is very similar to the sigmoid function except that it outputs in the range of $[-1, 1]$ rather than 0 to 1.

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

The *tanh* activation function produces output values that are centred around zero, allowing for a straightforward mapping of output values to strongly negative, neutral, or strongly positive categories. Usually employed in the hidden layers of a neural network, the *tanh* activation function has values ranging between -1 and 1. So, the mean of a hidden layer tends to be close to zero, which helps in the centring of data and facilitating the learning process for the subsequent layers. However, the derivative of *tanh* also suffers the vanishing gradient problem (as shown in Figure 2.7).

$$\frac{\partial \sigma_{tanh}(\boldsymbol{x})}{\partial \boldsymbol{x}} = 1 - \sigma^2_{tanh}(\boldsymbol{x})$$

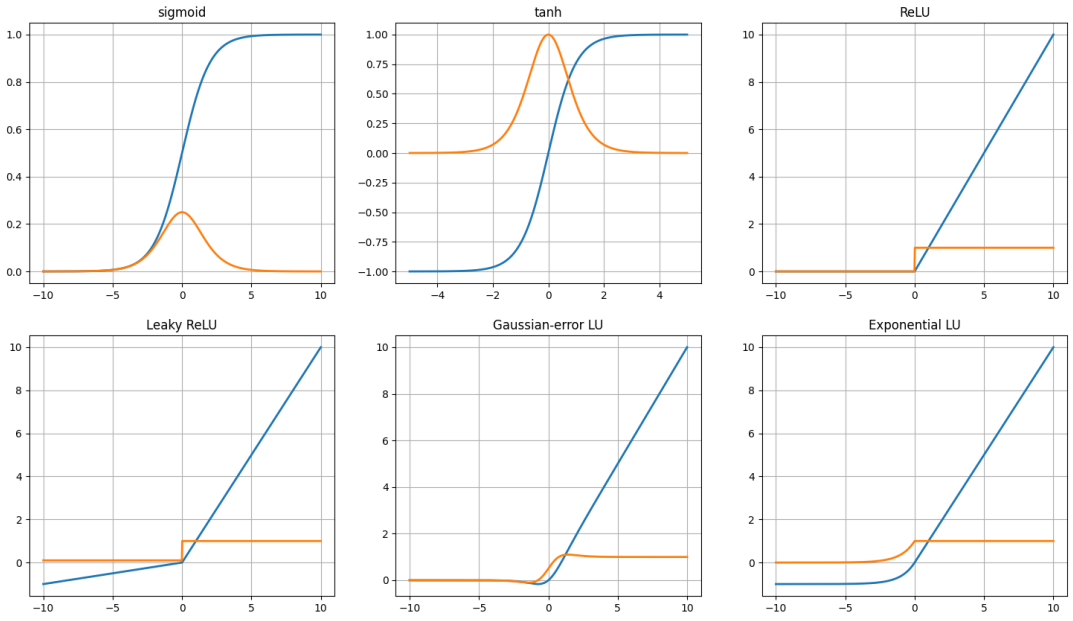


FIGURE 2.7: Different frequently used activation functions (blue lines) along with their gradients (orange lines).

- **Rectified Linear Unit:** The ReLU is a linear activation function. Though it gives an impression of linearity, ReLU does not activate all the neurons at the same time. Only the neurons with linear transformation greater than 0 are activated. Being a simple differentiable linear function, ReLU allows back-propagation and makes it computationally very efficient.

$$\sigma_{ReLU}(\boldsymbol{x}) = max(0, \boldsymbol{x})$$

The gradient of a ReLU function is constant 1 for input values greater than 0 while it gets value 0 for input less than 0. Although ReLU is computationally efficient and accelerates the convergence of gradient descent because of its linear and non-saturating property, ReLU suffers from **dying ReLU** problem (Lu, 2020). Since the gradient of input values less than 0 is 0, the weights for such neurons would not be updated in the back-propagation process. This might create dead neurons that never get activated. Since any negative input values instantly become zero, it decreases a model's capacity to properly fit or train on the data.

- **Leaky ReLU:** Dying ReLU problem is addressed by the leaky ReLU activation function as it has a small positive slope in the negative area too (shown in Figure 2.7).

$$\sigma_{L\_ReLU}(\boldsymbol{x}) = max(0.1\boldsymbol{x}, \boldsymbol{x})$$

Since a derivative exists for negative regions as well, no dead neurons get generated during the backpropagation. However, the derivative of small negative values

would be very small and make learning of the parameters slower.

Despite having a positive slope in the negative region, leaky ReLU is not always guaranteed to solve the problem of dying ReLU because of a constant multiplying factor of 0.1. A variant to overcome this issue is **Parametric ReLU** where the multiplying factor is learnt during the backpropagation.

$$\sigma_{P\_ReLU}(\boldsymbol{x}) = max(a\boldsymbol{x}, \boldsymbol{x})$$

where a is a learnable parameter.

- **Exponential Linear Unit (ELU):** ELU is another variant that modifies the slope in the negative region. A *log* curve is used to define the output in the negative region.

$$\sigma_{ELU}(\boldsymbol{x}) = \begin{cases} \boldsymbol{x}, & \text{for } x \geq 0 \\ \alpha(e^{\boldsymbol{x}} - 1), & \text{for} x < 0 \end{cases}$$

  Unlike ReLU, ELU becomes smooth gradually until its output reaches $-\alpha$, providing a more continuous transition. Additionally, ELU effectively addresses the dead ReLU problem by introducing a logarithmic curve for negative input values. However, ELU is a computationally expensive function as it involves exponential operation.

- **Softmax:** The sigmoid function outputs values in the range of 0 to 1 but does not constrain them in any other way. The softmax function, on the other hand, outputs values in the range of $[0, 1]$ such that the sum of all the outputs equals 1. Softmax is usually used as an activation function at the last layer to get the probabilities of the output classes. For a vector of K entries, the i^{th} element of output vector is given as;

$$\sigma_{softamx}(\boldsymbol{x})_i = \frac{e^{(\boldsymbol{x})_i}}{\sum_{k=1}^{K} e^{(\boldsymbol{x})_k}}$$

Objective functions An objective function (also referred to as cost or loss function) is used to quantify the error between predicted output from a neural network and the target labels. The objective of neural network training is to adjust weights and biases to minimise the objective function. Various objective functions such as cross-entropy, mean squared error, Kullback-Leibler (KL) divergence and Euclidean distance (L2) loss etc. are used depending on the task. Limiting the scope of this section to the work in this thesis, only the KL-divergence and cross-entropy losses are discussed here.

- **Kullback-Leibler divergence:** The KL-divergence $D_{KL}(P||Q)$ is a type of statistical distance to measure the divergence of a probability distribution P from a second, expected probability distribution Q. In the context of information theory

and statistics, KL divergence quantifies the difference between two probability distributions. For two probability distributions $P(x)$ and $Q(x)$ over a discrete random variable x, the KL divergence from P to Q is defined as;

$$D_{KL}(P||Q) = \sum_{X} P(x) \log \frac{P(x)}{Q(x)} \tag{2.33}$$

Although KL-divergence measures how two distributions are different, it is not a distance 'metric' and it is evident from the above equation that KL-divergence between two distributions is not symmetric. The value of KL-divergence between two distributions is always non-negative. In machine learning, KL divergence is usually used in variational methods and certain types of probabilistic models, such as Variational Auto-Encoder (VAE). It is also used in information retrieval and other fields to measure the difference between probability distributions or models.

- **Cross-Entropy:** Cross-entropy loss is the most widely used objective function in classification tasks. In information theory, **entropy** is the measure of the minimum number of bits required to transmit a randomly selected event from a probability distribution. A smaller number of bits are required for a skewed distribution and thus the entropy is lower and vice versa. Entropy $H(X)$ for a random variable X with the probability of its events as $P(x)$ can be calculated as;

$$H(X) = -\sum_{X} P(x) \log P(x)$$

Built upon the idea of entropy, cross-entropy calculates the (additional) number of bits required to transmit an average event from a distribution P when the optimal code is used for some other distribution Q.

$$H(P, Q) = -\sum_{X} P(x) \log Q(x)$$

Writing in terms of a loss function to train a machine learning model, P and Q can be thought of as target and the approximation of the target distributions respectively.

$$\mathcal{F}_{CE} = -\sum_{X} P(x) \log Q(x)$$

Cross-entropy can be deemed as a reduced form of KL-divergence loss. Equation 2.33 of KL-divergence loss can be written as;

$$D_{KL}(P||Q) = \sum_{X} (P(x) \log P(x) - P(x) \log Q(x))$$

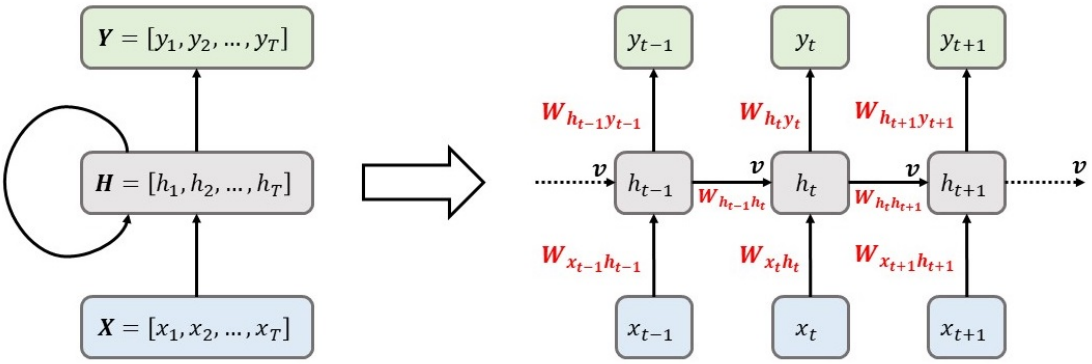


FIGURE 2.8: An example one-unit recurrent neural network. From bottom to top: input state, hidden state and the output state. The hidden state at each time step h_t is computed using the input at the given state x_t and the hidden state at the previous time step h_{t-1}.

Since $P(x)$ is the target probability distribution and has a value of 1 for the target class and 0 for the rest, the term $P(x) \log P(x)$ is always 0 because either $P(x) = 0$ or $\log P(x) = \log(1) = 0$. So, the above equation is reduced to the CE equation.

2.2.4.2 Recurrent neural networks

Recurrent neural networks are designed to process sequential or time-series data. Unlike feed-forward neural networks, RNNs have cyclic connections in their compact form which allows them to maintain a hidden state to memorise information about the previous inputs in the sequence. RNNs have been proven to outperform conventional neural networks for the speech recognition task (Graves et al., 2013b).

Similar to any other neural network, an RNN has input, output and one or more hidden layers. A simple RNN with one hidden layer is shown in Figure 2.8. RNNs are particularly designed for sequential data, allowing them to process one element of an input sequence at a time, maintaining an internal state to capture context from previous inputs. The key feature of RNNs is the presence of recurrent connections, which allow previous information to be used across different time steps. Each neuron in the network receives input not only from the current time step but also from its own output at the previous time step. This ability to retain memory makes RNNs particularly effective in tasks where context and temporal dependencies are crucial.

For a simple RNN with one hidden layer shown in Figure 2.8, the output of the hidden layer is described as;

$$\boldsymbol{h}_t = \sigma(\boldsymbol{x}_t \cdot \boldsymbol{W}_{x_t h_t} + \boldsymbol{h}_{t-1} \cdot \boldsymbol{W}_{h_{t-1} h_t} + \boldsymbol{b}_{x_t h_t})$$

where $\boldsymbol{W}$ are the weight matrices and b is the bias vector. $\boldsymbol{W}_{x_t h_t} \in \mathbb{R}^{F \times D}$ are the weights for the input to the hidden layer, $\boldsymbol{W}_{h_{t-1} h_t} \in \mathbb{R}^{D \times D}$ are the weights between the

neurons of the hidden layers and $\boldsymbol{b}_{x_t h_t} \in \mathbb{R}^{1 \times D}$ is the bias vector where D and F are the dimensions of feature vector and hidden representation vector respectively. Then the output of the RNN at timestep t is given as;

$$\boldsymbol{y}_t = \boldsymbol{h}_t \cdot \boldsymbol{W}_{h_t y_t} + \boldsymbol{b}_{h_t y_t}$$

where $\boldsymbol{W}_{h_t y_t} \in \mathbb{R}^{D \times C}$ is weight matrix from hidden to the output layer and C is the number of output classes.

RNNs are trained using Back-Propagation in time (BPPT) (Werbos, 1988, Williams and Zipser, 1995). In BPTT, the recurrent connections of an RNN are unfolded over time, creating a chain of connected instances of the network. As described above, the cell state and output for each time step are computed sequentially. The error is computed by comparing the output of each time step to the target output at the corresponding time step. In the backward pass, gradients of the error with respect to model parameters are calculated starting from the last time step and moving backwards to the start of the sequence. The gradients at each time step are then accumulated over the entire sequence. Although training of RNNs with BPPT addresses the challenges associated with the processing of sequential data through conventional neural networks, it poses its own challenges. RNNs can become computationally unstable and unable to hold sequential information for longer sequences which can impact their performance for applications such as machine translation and speech recognition etc. In the case of longer sequences, RNNs suffer vanishing or exploding gradients problems with BPPT training (Hochreiter and Schmidhuber, 1997). If the gradient becomes too small or too large, it becomes harder for a network to learn the weights as gradients explode or diminish exponentially. This issue is addressed by Long Short-Term Memory (LSTM) RNN architecture which can learn for a sequence of up to 1000 time steps without having gradient vanishing or exploding problems (Hochreiter and Schmidhuber, 1997). A gated recurrent unit has also been introduced which addresses the same problem but is more efficient than the LSTMs. In the following section, these two RNN architectures are described briefly.

Long Short-Term Memory In LSTM RNNs, each neuron (commonly referred to as a cell) also has an internal state. So in LSTMs, the internal state is also used along with input at the current time step and output from the previous time step to calculate the output at the current time step. The internal state or cell state carries the long-term information while the hidden state encapsulates the relevant information to be passed to the next time step or the output layer. The flow of the available information (i.e. input at the current time step, output from the previous time step and the cell state) is controlled by the 'gating' mechanism. An LSTM cell is shown in Figure 2.9.

LSTM's gating mechanism consists of three gates made of *sigmoid* function to control the flow of information into and out of a cell. The three gates are referred to as input,

output and forget gates.

- **Forget gate** determines what information from the previous cell state should be discarded. This allows the network to learn when to "forget" irrelevant information, preventing the accumulation of unnecessary information that can lead to vanishing gradients. Represented as $\boldsymbol{f}_t$ in the Figure 2.9, forget gate is described by the previous hidden state and the current time step input;

$$\boldsymbol{f}_t = \sigma(\boldsymbol{x}_t \boldsymbol{U}^f + \boldsymbol{h}_{t-1} \boldsymbol{W}^f)$$

  where U and W are the weight matrices which are different at each time step (though no subscript is used to denote that here).

- **Input gate** decides what new information should be added to the cell state. The input gate $\boldsymbol{i}_t$ is given as;

$$\boldsymbol{i}_t = \sigma(\boldsymbol{x}_t \boldsymbol{U}^i + \boldsymbol{h}_{t-1} \boldsymbol{W}^i)$$

- **Output gate** determines what information from the current cell state should be output as the hidden state. $\boldsymbol{o}_t$ in Figure 2.9 is computed as;

$$\boldsymbol{o}_t = \sigma(\boldsymbol{x}_t \boldsymbol{U}^o + \boldsymbol{h}_{t-1} \boldsymbol{W}^o)$$

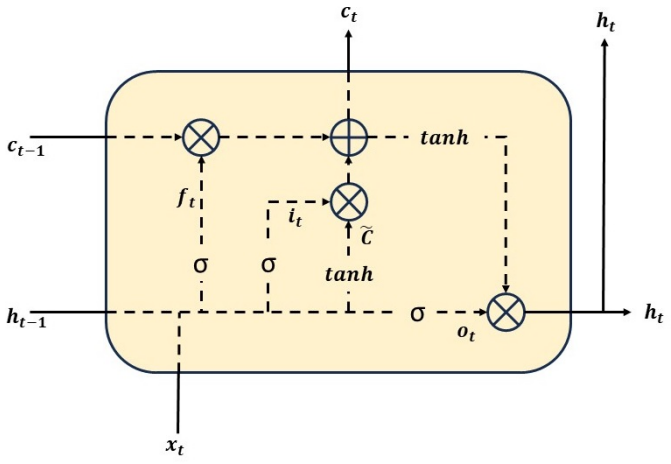


FIGURE 2.9: Illustration of the internal structure of an LSTM cell. The cell consists of three gates: the forget gate ($\boldsymbol{f}_t$), the input gate ($\boldsymbol{i}_t$), and the output gate ($\boldsymbol{o}_t$). The forget gate decides which information to discard from the cell state ($\boldsymbol{c}_{t-1}$), while the input gate determines which new information ($\tilde{\boldsymbol{C}}$) to store in the cell state. The output gate selects which part of the cell state to output as the hidden state ($\boldsymbol{h}_t$). The cell state ($\boldsymbol{c}_t$) and hidden state ($\boldsymbol{h}_t$) are passed to the next time step of the LSTM cell.

The current cell state of an LSTM cell is given as;

$$\boldsymbol{C}_t = \sigma(\boldsymbol{f}_t \boldsymbol{C}_{t-1} + \boldsymbol{i}_t \tilde{\boldsymbol{C}}) \tag{2.34}$$

where $\tilde{C}$ (as shown in Figure 2.9) is given as;

$$\tilde{\boldsymbol{C}}_t = tanh(\boldsymbol{x}_t \boldsymbol{U}^g + \boldsymbol{h}_{t-1} \boldsymbol{U}^g)$$

So, Equation 2.34 can be re-written as;

$$\boldsymbol{C}_t = \sigma(\boldsymbol{f}_t \boldsymbol{C}_{t-1} + \boldsymbol{i}_t (tanh(\boldsymbol{x}_t \boldsymbol{U}^g + \boldsymbol{h}_{t-1} \boldsymbol{W}^g)))$$

It is evident from this equation that the update in the cell state is controlled by the input gate ($\boldsymbol{i}_t$) which controls the flow of new information into the cell state. The hidden state and the output of a cell are then the product of the activated current cell state and the output gate.

$$\boldsymbol{h}_t = tanh(\tilde{\boldsymbol{C}}_t) \boldsymbol{o}_t$$

The control of information flow mechanisms in LSTMs helps to alleviate the problems of traditional RNNs. LSTM networks can be uni-directional or bi-directional (BLSTMs) (Graves and Schmidhuber, 2005, Graves et al., 2013b). BLSTMs allow the flow of information in both directions by adding an additional layer to handle the reverse flow of information. Outputs of the forward and backward layers can be combined by accumulation or concatenation etc. BLSTMs perform better than uni-directional LSTMs for various tasks including speech recognition (Graves et al., 2013a).

Gated recurrent unit The gated recurrent units (Cho et al., 2014) have been proposed as an efficient alternative to LSTMs and consist of only two gates i.e. update gate and the reset gate. A GRU cell is shown in Figure 2.10.

- **Update gate** in a GRU determines how much of the previous hidden state should be retained and how much of the new candidate hidden state should be incorporated. It takes into account the current input and the previous hidden state. The value of update gate $\boldsymbol{z}_t$ (as shown in Figure 2.10) is computed as;

$$\boldsymbol{z}_t = \sigma(\boldsymbol{x}_t \boldsymbol{U}^z + \boldsymbol{h}_{t-1} W^z)$$

- **Reset gate** controls the extent to which the previous hidden state should be ignored when computing the new candidate hidden state. It helps the model to selectively forget irrelevant information. The reset gate is described as;

$$\boldsymbol{r}_t = \sigma(\boldsymbol{x}_t \boldsymbol{U}^r + \boldsymbol{h}_{t-1} W^r)$$

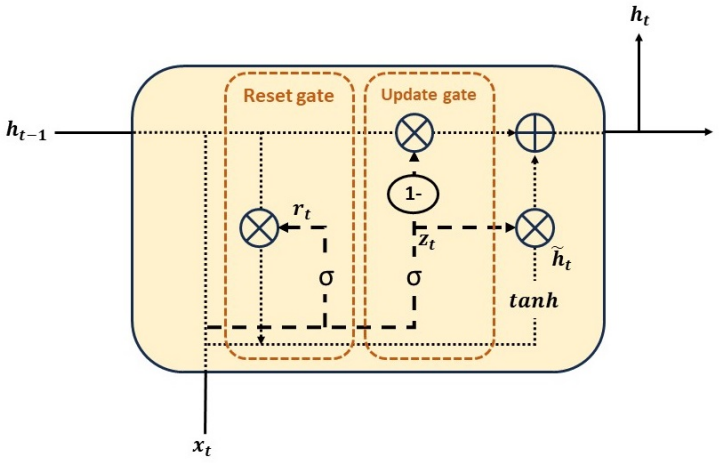


FIGURE 2.10: A gated-recurrent unit cell with three main components: an update gate ($\boldsymbol{z}_t$), a reset gate ($\boldsymbol{r}_t$), and a candidate hidden state ($\tilde{\boldsymbol{h}}_t$). The input data (input at current time $\boldsymbol{x}_t$ and last hidden state ($\boldsymbol{h}_{t-1}$) flow into the GRU cell, which then processes the data and produces an output ($\boldsymbol{h}_t$). The update gate determines whether the hidden state ($\boldsymbol{h}_{t-1}$) is to be updated with a new hidden state ($\tilde{\boldsymbol{h}}_t$), while the reset gate decides whether the previous hidden state should be disregarded.

The actual activation of an RNN node is then computed by;

$$\boldsymbol{h}_t = \boldsymbol{z}_t \boldsymbol{h}_{t-1} + (1 - \boldsymbol{z}_t)\tilde{\boldsymbol{h}}_{t-1}$$

where

$$\tilde{\boldsymbol{h}}_{t-1} = tanh(\boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{U}(\boldsymbol{r} \odot \boldsymbol{h}_{t-1}))$$

where $\odot$ is the Hadamard (or element-wise) product of two vectors. It is evident from the last equation that if the value of the reset gate is very small, the last hidden state is forced to ignore and the output of the current state mostly depends on the input only. The update gate, on the other hand, controls how much information from the previous hidden state would go into the current hidden state. If the value of the update gate is very small, current hidden state $\boldsymbol{h}_t$ gets more information from previous state $(1 - \boldsymbol{z}_t)\tilde{\boldsymbol{h}}_{t-1}$. This acts similarly to the memory cell of an LSTM which carries the information of the long-context (Cho et al., 2014). Each hidden unit of an RNN has separate update and reset gates and thus learns to capture dependencies over different time scales.

Since the GRU implementation is simpler than an LSTM cell, it results in fewer parameters and is thus efficient to train. Even with fewer parameters, they can outperform the more complex LSTM models for speech recognition task (Ravanelli et al., 2018). Similar to LSTMs, GRUs can also be implemented in uni-directional or bi-directional configurations (Bhuvaneswari et al., 2019).

There are various other popular architectures such as RNN-Transducer (RNN-T) which are being widely used for speech recognition. However, those are beyond the scope of

this thesis and so not discussed here.

Objective functions Earlier, cross-entropy and KL-divergence objective functions have been discussed which are used for frame-based speech recognition systems. Since sequence models (i.e. RNNs) prove to perform better than conventional neural networks because they are specifically designed to handle sequential information, sequence-based objective functions have also been introduced such as **Connectionist Temporal Classification (CTC)** objective function for speech recognition (Graves et al., 2006). CTC allows the training of recurrent neural networks without the need for aligned input-output pairs, making it particularly useful for problems where the alignments between input and output sequences are not known during training. CTC objective is usually used to train a sequence model which will be explained in detail in the Section 2.2.5.2.

2.2.4.3 Convolutional neural networks

Motivated by an idea of a neurophysiology discovery that neurons in a cat's visual cortex were sensitive to specific local patterns (Hubel and Wiesel, 1962), convolutional neural networks have been initially used for image processing (Lecun et al., 1998). Convolutional neural networks employ shared weights and local receptive fields to capture temporal and spatial information. CNNs are designed for grid-like data structures such as images while preserving the spatial hierarchies and local patterns in data.

Usually, CNN models consist of blocks of convolution layers followed by pooling layers. A **convolution layer** is designed to perform convolution operations on 2D input data. A small filter (known as a kernel) is moved over the input to compute a dot product at each position. This results in a feature map that captures local patterns in the input. A kernel or filter is a small learnable matrix that is applied to the local regions of an input. The weight parameters of the filters are shared across different spatial locations. This parameter sharing reduces the number of parameters in the network. A pooling layer is used to reduce the dimensionality and information aggregation on the output of a CNN layer and capture the most essential information from the input. They generally do not have any learnable parameters. Average pooling and max pooling are the two most commonly used pooling types. In max pooling, the maximum value from the input window is selected while average pooling takes the average of the input window. Pooling could be applied on the output of a single filter to further reduce the dimensionality or across the different channels of a CNN layer (output from different kernels).

In speech recognition, CNN layers are used as the initial layers to the sequence models where input is given in 2D (Abdel-Hamid et al., 2014). Features of all the frames of an audio are concatenated to form a 2D input which is then fed to CNN layers. An example is shown in Figure 2.11 where the dimensionality of a speech signal can be reduced by capturing information in the low-dimension using CNN layer(s). In the shown figure, a fully connected neural network is used after flattening the output of the final pooling layer. However, any other model such as RNNs can also be used (Hori et al., 2017).

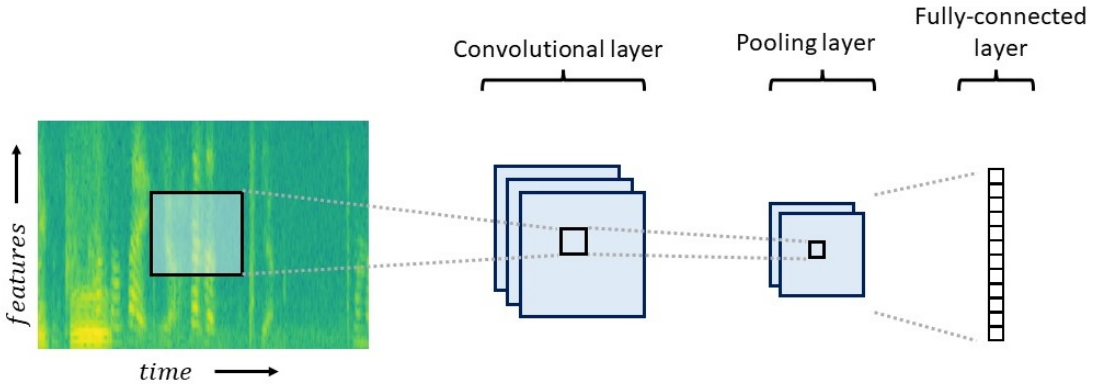


FIGURE 2.11: Illustration of a high-level CNN model with a convolutional, pooling and a fully connected layer. A small kernel of the convolution layer performs convolution operation over the input image to capture spatial information in a low-dimension output. A pooling layer is used to perform pooling over the output of a convolution layer to further reduce the dimensionality. The last fully connected layer is usually used to output probabilities of a finite number of classes for the downstream task.

2.2.5 End-to-end models

In this section, various end-to-end speech recognition systems and their training are discussed. End-to-end ASR systems can be trained in various ways using different architectures. However, the scope of this section is only limited to the architectures and regimes which are used later in this work.

2.2.5.1 Encoder-Decoder models

A technique proposed to use two RNN models for machine translation has proven to outperform a model of single RNN (Cho et al., 2014). One RNN is used as an encoder which encodes the input data in the hidden layers. The hidden layer representation is then fed to another RNN which is referred to as a decoder. The decoder outputs the target labels. With its success in machine translation and several other sequence tasks such as image captioning (Xu et al., 2015b), encoder-decoder models have been employed for speech recognition and proven to perform better than frame-based ASR systems (Chorowski et al., 2014, Lu et al., 2016).

In encoder-decoder models, the encoder processes the variable-sized input sequence and captures the information into a fixed-size representation. The decoder takes the last encoder state and generates the output sequence step by step. At each step, it produces the next element in the sequence based on the last decoder state and the previously generated output tokens (autoregressive) or using only the last decoder state (non-autoregressive) (Gu et al., 2018). As the decoder predicts output using representations from the encoder, the output sequence length does not depend on the input sequence and thus no alignments are required. Only encoder-decoder models are discussed here which are based on recurrent neural networks, however, transformers-based encoder-decoder models have also been used for speech recognition (Chang et al., 2020). In

RNN-based encoder-decoder models, both encoder and decoder consist of one or more RNN (such as GRU or LSTM etc.) layers. At each time step, the decoder gets the hidden representation and output from the last time step to output the token i.e.;

$$\boldsymbol{h} = RNN(\boldsymbol{x})$$
$$y_n = RNN(y_{n-1}, s_{n-1})$$

where y_n is the output at n^{th} output step and s_{n-1} is the decoder state from the last output step. For $n = 1$, s_{n-1} is the last representation from the encoder $\boldsymbol{h}$.

A variant of the encoder-decoder model where the decoder is replaced by an attentional decoder proves to perform better (Chan et al., 2016). In such a model, the output at each time step also takes a context vector into account. A context vector is a weighted vector of all the hidden representations of the encoder. At each time step, an attention vector is trained to learn how much the encoder's hidden representations are to be attended.

$$\boldsymbol{h} = RNN(\boldsymbol{x})$$
$$c_n = AttentionContext(s_n, \boldsymbol{h})$$
$$s_n = RNN(s_{n-1}, y_{n-1}, c_{n-1}$$
$$y_n = MLP(s_n, c_n)$$

At each decoder timestep, *AttentionContext* generates a context vector c_n taking the decoder state from the last time step and all the encoder representations. The last decoder state s_{n-1}, previously generated output token y_{n-1} and the last context vector c_{n-1} are given as input to the decoder RNN which generates decoder state s_n at current time step. This current decoder state s_n and the context vector c_n are fed to a multi-layer perceptron (or a fully-connected) layer with a softmax activation function to output the probabilities of the output tokens at each time step.

AttentionContex computes scalar energy ($e_{i,n}$) for each encoder timestep i at each decoder timestep n taking h_i and s_n as the inputs where h_i is the encoder output at i^{th} timestep (Chan et al., 2016). Then, c_n is the encoder outputs $\boldsymbol{h}$ weighted by corresponding normalised scalar energy.

$$e_{i,n} = (\phi(s_n), \Phi(h_i))$$
$$\alpha_{i,n} = \frac{e_{i,n}}{\sum_i e_{i,n}}$$
$$c_n = \sum_i \alpha_{i,n} h_i$$

where ϕ and Φ are multi-layer perceptron networks (Chan et al., 2016).

2.2.5.2 CTC training

As discussed earlier in Section 2.2.5.1 RNNs do not need alignments for ASR training. Rather, they have been trained to output a series of tokens in a way of independent label classification (Graves et al., 2006). This implies that the training data must be pre-segmented and the output needs to be post-processed to have the final output sequence. However, in CTC training (Graves et al., 2006), RNNs can be directly trained with sequence labelling (without pre-segmenting).

For CTC training of a model, the possible output tokens L are the tokens for the task L' (such as graphemes of a language for a grapheme-based ASR system) and one additional blank symbol (let's denote it with ϵ; $L = L' \cup \{\epsilon\}$). The task of the CTC training is to minimise the 'label error rate' which is an edit distance measure between the ground-truth labels sequence and the network output label sequence. The decoder RNN outputs the probabilities of the token symbols from L' or ϵ at each timestep where the probability of ϵ implies no symbol or a 'blank'. Let y_k^t be the probability of k symbol at time t which defines a distribution over the set L^T of length T sequences:

$$p(\pi|\boldsymbol{x}) = \prod_{t=1}^{T} y_{\pi_t}^t \quad \forall\ \pi \in L^T$$

where $\boldsymbol{x}$ is the input signal and π is the elements in L^T (referred to as paths). An implicit assumption here is that the network outputs are conditionally independent as no previous output is used for the prediction of future outputs (Graves et al., 2006).

Having the output sequences from the network for a given signal $\boldsymbol{x}$, a many-to-one mapping step ($\mathcal{B}(.)$) is done in which the repeated and blank symbols are removed. For example, $\mathcal{B}(aa\epsilon ab) = \mathcal{B}(a\epsilon\epsilon ab) = aab$. So, two different paths from network output can lead to the same output sequence. Finally, the probability of a labelling sequence $\boldsymbol{l} \in L^{\leq T}$ is calculated as the sum of probabilities of all the probable paths for $\boldsymbol{l}$.

$$p(\boldsymbol{l}|\boldsymbol{x}) = \sum_{\pi \in \mathcal{B}^{-1}(l)} p(\pi|\boldsymbol{x})$$

The final output sequence can be worked out as the most probable sequence from the above equation (*best-path*) or the backward-forward algorithm (*prefix-search* as described in detail in (Graves et al., 2006)). The backward-forward algorithm is similar to the one explained in Section 2.2.1.2 for hidden Markov models. The probabilities of all the paths are calculated in the same way in the forward path. For the most probable path at the end of the forward pass, corresponding labels are given as output in the backward pass.

ASR systems trained with CTC training outperform frame-based ASR systems (Graves et al., 2006) and are the most widely used training scheme for most of the ASR systems (Amodei et al., 2015, Graves and Jaitly, 2014). However, the Markov assumption that CTC relies on, regarding conditional independence between output symbols, can hinder

its ability to capture complex dependencies among those symbols. So, some variants have been introduced to exploit CTC further. The hybrid CTC/attention loss (Watanabe et al., 2017) is briefly discussed here as that is used in some models trained in this work.

Hybrid CTC/Attention architecture The objective of hybrid CTC/attention training (Watanabe et al., 2017) is to leverage the benefits of both CTC and attention models. CTC is effective for tasks where the alignments are mostly monotonic while the attention mechanism does not make Markov independence assumption. In the hybrid CTC/attention training benefits of both, CTC and attention are leveraged to train a model. Hybrid CTC/attention architecture usually consists of an RNN encoder, attentional decoder and a feed-forward neural network. This is very similar to the architecture already explained for the encoder-decoder model. The difference is that a CTC loss is also applied on the top of the encoder while frame-wise classification loss is applied on the top of the decoder. One or a few feed-forward layers are applied on the top of the encoder layers to apply CTC loss. So, the encoder is trained over the gradients calculated from both, CTC and classification loss, while the decoder is trained using only one (classification) loss. The speech recognition systems trained with hybrid CTC/attention objective have been proven to perform better than only CTC or frame-wise classification models (Hou et al., 2020, Watanabe et al., 2017).

Chapter 3

Multilingual Speech Recognition

In the past few decades, automatic speech recognition systems have seen a lot of developments from HMM models (Rabiner and Juang, 1986) to large self-supervised models such as wav2vec (Baevski et al., 2020), Hubert (Hsu et al., 2021b) and Whisper (Radford et al., 2022) are a few to name. It is no longer just a research topic, rather speech recognition systems have been deployed in a variety of applications such as voice assistants (Sri, Google Assistant and Cortana etc). However, a few languages have been predominantly focused on for building speech recognition systems. More than 7000 languages are being spoken around the globe. So from the application perspective, the need of the time is to build ASR systems that could support multiple input and output languages. However, a lot of languages do not have sufficient available data resources for building their speech recognition systems. In this chapter, the efforts made towards building multilingual speech recognition systems and ways to exploit cross-lingual resources to train ASR systems for under-resourced languages are revisited. The work in this thesis mostly revolves around acoustic modelling, so the acoustic modelling is mainly focused on while revisiting the literature.

Earlier works on speech recognition have been limited only to the English language (Gauvain et al., 1994). However, the advancements in the domain attracted researchers to build ASR systems for other languages too (Glass et al., 1995, Lamel et al., 1995). The implementation of core speech recognition systems is independent of the target language and all the core components remain the same for speech recognition systems of different languages (Schultz and Kirchhoff, 2006b). Many works have shown that the speech recogniser trained for a language can be ported to other languages (Glass et al., 1995, Lamel et al., 1995). Lamel et al. (Lamel et al., 1995) ported the GMM-HMM-based speech recognition systems of American English and French to British English and German languages. Porting of a speech recogniser to another language needs to change certain system components or parameters such as; the change of phonemes set requires to train HMM models for new phonemes (Lamel et al., 1995). Even for the same phonemes, new language-dependent knowledge is needed to be incorporated. The success of porting

speech technology to other languages has given rise to the idea of sharing knowledge across languages. Multilingual ASR has been trained in the sense that the HMM models of shared phonemes have been trained using training data of all the sharing languages (Kohler, 1996). As described earlier in Section 1.3.2, multilingual speech recognition systems gained attention because of their many-fold advantages (Schultz and Kirchhoff, 2006b);

- A multilingual speech recognition is capable of recognising input speech from several multiple languages. If the input language is from a close set, a multilingual ASR system performs implicit language identification and converts speech input to the text in the corresponding language (Zhou et al., 2022). From an application perspective, ideally, a universal ASR (a speech recognition system which can recognise speech from any language of the world) is needed.
- A single model is easy to maintain and deploy for multiple languages rather than having a separate speech recognition system for each language (Schultz and Kirchhoff, 2006b).
- Code-switching is a common phenomenon of spoken language by bilingual speakers (Farooq et al., 2020). A multilingual ASR enables recognition of code-switched speech whereas a monolingual ASR can not handle code-switched speech (Schultz and Kirchhoff, 2006b).
- Another advantage is about exploiting multilingual ASR systems or resources to improve speech recognition of low-resource languages (Kannan et al., 2019, Kovac, 2005, Li et al., 2014a, Lin et al., 2009). It is quite challenging to train a good ASR using very limited data resources. Various approaches have been adopted to improve the speech recognition of languages with limited data exploiting data or speech recognition systems of other languages (Chen and Mak, 2015, Gaur et al., 2021, Heigold et al., 2013, Sailor and Hain, 2020).

However, multilingual speech recognition poses its challenges. In the following sections, challenges associated with multilingual speech recognition are briefly discussed. The work done for the aforementioned aspects of multilingual speech recognition is revisited after that.

3.1 Challenges for multilingual speech recognition

Limited attention has been paid to multilingual speech recognition research until the last few years. Multilingual speech recognition is more challenging because of data requirements and language peculiarities. A few challenges associated with multilingual data and languages are discussed here.

3.1.1 Data collection

Machine learning-based speech recognition approaches are data-driven and require a reasonable amount of data to train a speech recognition system of a language. Lexicon or pronunciation modelling has been among the core components of GMM-HMM-based speech recognition system which is usually designed by linguists and requires language knowledge. So, language knowledge is also needed along with the speech data.

Lack of resources A large amount of speech data of the target language is required to train a speech recognition system. As the earlier research on speech recognition has been mainly focused on the English language, efforts have been made to collect English speech data such as TIMIT (Fisher et al., 1986), Wall Street Journal (WSJ) (Paul and Baker, 1992) and Switchboard (SWBD) (Godfrey et al., 1992) etc. In case of porting an ASR system to another language, the speech data of the target language is required which needs very expensive and laborious efforts. The problem is exacerbated by the increasing number of languages. So, sufficient data resources of multiple languages are needed to train a multilingual speech recognition system. However, data of very few languages, that are spoken by a large proportion of the population, have been collected during the earlier works (Hess et al., 1995, Larnel et al., 1991, Paul and Baker, 1992, Robinson et al., 1995). Furthermore, linguistic knowledge such as phonemes and words-to-phonemes maps are also needed. This makes language research a prerequisite for building an ASR system.

Uniformity in data For a fair learning of each language in a multilingual speech recognition system, the quality of data of all the languages should ideally be uniform in several aspects such as duration, speakers, speaking style, data domain, recording equipment and environment etc. Data recorded with certain factors makes learning more challenging than others. For example, read speech data is easier to understand (and thus less challenging to learn by machine) as compared to spontaneous conversational speech. Similarly, speech with background noise in an open environment makes the recognition task harder than the speech recorded in a very clean environment. In a multilingual setup, mixing data from different languages that were recorded under varying conditions introduces a bias in the learning process (Schultz and Kirchhoff, 2006b), which is not good for a fair comparison of the different languages.

3.1.2 Language units

The objective of multilingual speech recognition has been to share the acoustic data across multiple languages (Lin et al., 2009). A common phonetic alphabet system for all the languages such as IPA (Association, 1999) has been defined. Speech data of the languages can be transcribed on words, phonemic or phonetic level. For phoneme-based ASR systems, utterances are either transcribed on a phonemic level or are converted to phonemic transcriptions using a pronunciation dictionary if annotated on a word level. In IPA-based phonemic transcriptions, there is an implicit assumption that the phonemes

represented by the same IPA representations across the languages are acoustically similar too (Schultz and Kirchhoff, 2006b). However, phonemes of different languages might have different underlying acoustic realisations (phones). Given pronunciation dictionaries of different languages in a uniform phonemic representation (i.e. IPA), lexicons from various languages can be joined together for multilingual ASR training (Schultz and Kirchhoff, 2006c). The number of total phonemes becomes the union of individual phoneme sets of all the languages for the multilingual ASR which is always a subset of the universal phoneme set. A universal phoneme set is a set consisting of phonemes from all the languages and consists of a limited number (The Phoible[1] (Moran and McCloy, 2019) database of 2186 languages contains 3183 phonemes to-date).

In the case of e2e of speech recognition systems, most of the languages have very diverse writing systems. A significant number of languages do not even have a written form (Schultz and Kirchhoff, 2006c). For a multilingual ASR, the number of output tokens would be the union of tokens of all the individual languages which amounts to a significantly large number of classes if the participating languages have different writing scripts. It makes it difficult for a model to learn even if the languages are acoustically close (Schultz and Kirchhoff, 2006b).

The aforementioned challenges in building multilingual speech recognition raised slow progress in this domain until the recent past. Recently, a significant increase has been seen in multilingual data resources (Ardila et al., 2020, Pratap et al., 2020b) and efforts have been made to build massively large multilingual speech recognition (Li et al., 2022a, Pratap et al., 2020a, Tjandra et al., 2022) systems as well as exploiting these resources to improve ASR performances for low-resource languages (Hsu et al., 2020, Shetty and Sagaya Mary N.J., 2020).

3.2 Massively multilingual speech recognition systems

It has been a long-standing goal of speech recognition research to develop a single model which is capable of recognising speech from multiple languages (Pratap et al., 2020a). Until the late past decade, the research on multilingual speech recognition has been limited to a set of a few languages (around 5-10 mostly) (Burget et al., 2010, Kannan et al., 2019, Madikeri et al., 2020). Furthermore, the languages of interest have been among the very rich-resource languages for a very long time such as English, German and French etc (Burget et al., 2010, Lin et al., 2009). The potential reason is the unavailability of data and limited computational resources. Multilingual data sets are discussed in detail in Section 3.5, a very few multilingual speech data sets have been available until the recent past.

Recently, some projects such as CommonVoice (Ardila et al., 2020) and Babel (Gales et al., 2014) have stepped towards collecting speech data of various languages. Though the GlobalPhone corpus consists of speech from multiple languages, it has not been a

[1] phoible.org [Last accessed: 14th January 2024]

freely available resource. CommonVoice, on the other hand, consists of the actively collected data sets. Resources for both existing and additional languages have been actively collected and made freely available to the public. Furthermore, the wider use of modern technology has enabled various companies to have in-house public data to build their speech recognition systems (Li et al., 2022a, Pratap et al., 2020a).

Hou et al. (Hou et al., 2020) have built a large-scale single e2e multilingual model which has been trained on around 5000 hours of speech data from 42 languages. The data for the participating languages have been sourced from different available corpora i.e. CommonVoice, Voxforge and Babel etc. The speech recognition system has been trained using a hybrid CTC/attention objective which achieved an average word error rate of 52.8%. A language tag has been appended at the start of each transcription. Language identification performance is measured in terms of the correct tag prediction by the model. The system achieved a language identification accuracy of 93.5%. The trained model has been used to bootstrap a model of 14 unseen low-resource languages. The low-resource multilingual ASR system has seen a reduction in average WER from 28.1% to 11.4% when bootstrapped from the large-scale multilingual model. Since the data of different languages have been included from different corpora, the data may have been recorded in non-uniform conditions such as recording equipment and environment. So the training and thus the performance of the model may have been influenced by these factors.

The work done by Pratap et al. (Pratap et al., 2020a) discusses building a massive multilingual model of up to 1 billion parameters. The model has been trained using about 16,000 hours of speech data from 51 languages. The speech data is Facebook's (now Meta) in-house data and is not publicly available. The participating languages range from rich-resource to low-resource languages. Data (in terms of hours) varies from 100 to 1100 hours for low to high-resource languages. Various efforts have been made to avoid over-fitting and catastrophic forgetting. An average relative WER improvement of up to 28.8% has been observed in comparison with monolingual models of the languages. Results show that the resource-rich languages show degradation, no or very-low improvement in multilingual models when compared with the performance of their monolingual models. This work has recently been extended to build a bigger model. An RNN-T model has been trained for 70 languages using 150,000 hours of their in-house data set (Tjandra et al., 2022).

Catastrophic forgetting is a problem in machine learning when a model starts forgetting previously learnt information while learning new tasks (McCloskey and Cohen, 1989). Li et al. (Li et al., 2022a) proposed a lifelong learning solution to build a massively large multilingual ASR system avoiding the problem of catastrophic forgetting. The final model is trained for 67 languages using 670,000 hours of in-house data from Google. The model is gradually trained for different sets of languages and does not see degradation in the performance of languages it learnt as the first set of tasks. This process is expected

to speed up the development of a language-universal speech recognition system (which could recognise speech from ideally any language of the world) (Li et al., 2022a).

With the recent trends towards self-supervised modelling (Baevski et al., 2020, Hsu et al., 2021b), various multilingual SSL models have been trained (Babu et al., 2022, Conneau et al., 2021). Wav2vec2.0 model has been trained using 56,000 hours of unsupervised speech data of 53 languages (XLSR-53 from Babel, MLS and CommonVoice corpora) (Conneau et al., 2021). The model consists of nearly 317 million parameters. This has later been extended to build a bigger model of (up to 2 billion parameters) which have been trained on 128 languages (Babu et al., 2022). It also included data from VoxPopuli and VoxLingua data resources to have 436,000 hours of unsupervised speech to train the model (XLS-R).

3.3 Improving low-resource speech recognition

One main objective of multilingual speech recognition is to share language resources to improve speech recognition of low-resource languages (Lin et al., 2009). Many different approaches have been made towards solving this problem which are discussed here.

3.3.1 A single multilingual model

Many studies have shown that multilingual speech recognition systems reduce speech recognition error rates for low-resource languages when compared with the monolingual ASR systems (Li et al., 2014a). A multilingual model is trained by mixing training data from all the participating languages before training. However, various architectures and training procedures have been explored to train a multilingual model. For example, a conventional approach could be where all the layers are shared among all languages and the output tokens are the union of output tokens of all individual languages. A multilingual ASR system has also been trained with multitask machine learning (Caruana, 1997, Kovac, 2005). One approach with Multi-Task Learning (MTL) is to build a model with some initial shared layers among all the languages followed by language-dependent layers (or output layer) (Heigold et al., 2013). Another approach is to have speech recognition as the primary task and language identification as the auxiliary task in an MTL-trained model (Sailor and Hain, 2020). Many other approaches have been experimented with multi-task learning approaches such as recognising phonemes and graphemes as parallel tasks (Chen and Mak, 2015).

Kannan et al. (Kannan et al., 2019) have trained an RNN-T-based multilingual speech recognition system. However, the authors have used Learning Hidden Unit Contributions (LHUC) (Swietojanski and Renals, 2014b) like language-dependent adapter layers before each layer in the encoder of RNN-T. These adapter layers are expected to learn weights to assign each layer based on the input language. In a similar approach, a mixture-of-experts (Shazeer et al., 2017) has been employed in a multilingual speech recogniser (Gaur et al., 2021). In this approach, each layer of the RNN-T encoder learns language-independent and language-dependent parameters, and the way to activate them.

So far, training of a multilingual model in different ways has been discussed. Many other approaches have been adopted where multilingual or cross-lingual resources have been exploited to improve low-resource speech recognition such as using the multilingual ASR system as a feature extractor, transfer learning and data augmentation. These approaches are revisited briefly now.

3.3.2 Feature extraction and transfer learning

Previous work has shown that using bottleneck features from a pre-trained multilingual speech recognition system to train a low-resource ASR system helps in improving the performance rather than using conventional MFCC or PLP etc. features (Grézl et al., 2014, Heigold et al., 2013, Thomas et al., 2012, Veselý et al., 2012). For this purpose, usually a multilingual model is trained which consists of initial shared layers among the languages followed by a low-dimension layer and then language-dependent layers. The intuition is that the layer before the language-dependent layers would learn language-independent representations. The schematic of such a model is shown in Figure 3.1. Once the model is trained, the target language data is passed through this model and the representations from the bottleneck layers are extracted as the features to train another model. These representations help improve performance regardless of whether the target language has been among the pre-training languages for the multilingual model (Heigold et al., 2013) or not (Heigold et al., 2013, Thomas et al., 2012, Veselý et al., 2012). It means bottleneck features can be extracted using multilingual or cross-lingual resources.

Thomas et al. (Thomas et al., 2012) have combined both approaches, bottleneck features and transfer learning, to improve low-resource speech recognition. A multi-layer

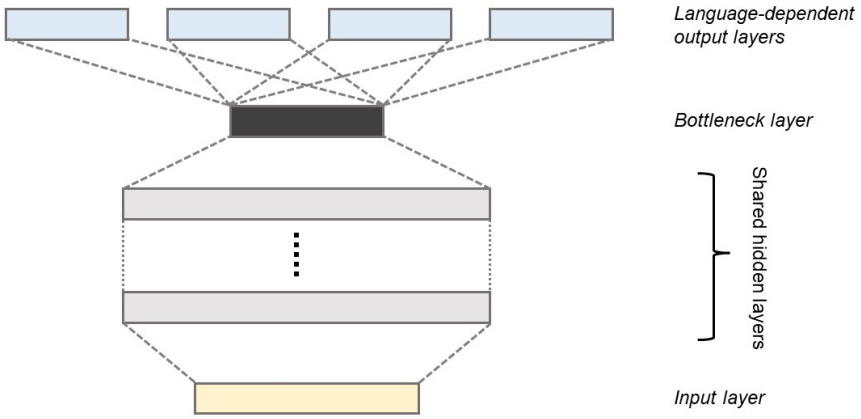


FIGURE 3.1: Illustration of a multi-task multilingual speech recognition system to extract bottleneck features. The neural network consists of an input, several shared hidden layers, a low-dimension hidden layer (bottleneck layer) and multiple output layers (one for each language usually). Once the model is trained using data from multiple languages, speech data of a target language is passed through this model and the representations from the bottleneck layer are extracted as the features to train another model.

perceptron is trained for multilingual speech recognition using speech data from multiple languages. Except for the output layer, the layers of this trained model are used to bootstrap a model for a low-resource language. The final layer for the new model is bootstrapped from a single-layer MLP. This single-layer MLP is trained using bottleneck features of the target (low-resource) language from a pretrained multilingual model. This approach proves to perform better for a low-resource language when compared with randomly initialised models and using conventional features such as PLP. Various other studies have also shown the advantage of using features extracted from a multilingual ASR system to train a monolingual speech recognition system (Tüske et al., 2013).

Similarly, studies have been done to show that a bootstrapped model to train an ASR system for a low-resource language outperforms a randomly initialised model (Hou et al., 2020, Huang et al., 2013, Vu and Schultz, 2013). A multilingual speech recognition system is trained using data from multiple languages. The layers of this model are then used as initialisation for the layers of a low-resource speech recognition system. The target language-dependent output layer is usually added on top of these layers. Either the parameters of the whole model are updated (Huang et al., 2013) during the training (also referred to as fine-tuning) or only the output layer is updated while the rest of the layers are kept unchanged (Huang et al., 2013, Vu and Schultz, 2013). The approach not only works for the seen languages in multilingual training but it has also been proven to perform well for unseen (cross-lingual) target languages (Huang et al., 2013, Vu and Schultz, 2013).

3.3.3 Multilingual knowledge distillation

Recently, efforts have been made towards multilingual knowledge distillation with an aim to utilise multilingual resources to improve low-resource speech recognition, where multilingual models are used to train language-specific student ASR models (Leal et al., 2021). Student-teacher training or knowledge distillation (Hinton et al., 2015) has been widely used to transfer knowledge from either single or multiple teacher models (Huang et al., 2023) to a student model. This technique involves transferring a teacher's knowledge to a student model either at the output layer (Hinton et al., 2015) or at intermediate stages (Romero et al., 2015) and has been applied to tasks such as model compression (Huang et al., 2023, Kim et al., 2019) and domain generalisation (Fang et al., 2021, Kim et al., 2021, Wang et al., 2021). The student model is trained with the combined objective of minimising the KL-divergence loss for the prediction of the teacher's posteriors (soft labels) and a classification loss with the original training labels (hard labels).

Since KL-divergence loss is used as KD loss between a teacher's soft labels and the student model posteriors (Hinton et al., 2015), the output classes of the student model should be a subset of the teacher model. This is because KL divergence loss is calculated between two posterior distributions and makes sense only if the posterior distributions represent the probabilities of the same classes from two different models. However, multilingual knowledge distillation studies have used teacher models where the student

model output classes are an improper subset of the teacher model classes (Leal et al., 2021). In such cases, to train a student model for a specific language, the posterior distribution of a multilingual model is used, dropping the probabilities of the classes that do not belong to the target language. The updated posterior distribution is then normalised before calculating the KL divergence loss. Some studies have used multiple multilingual teacher models to distil knowledge for training a monolingual student model (Shen et al., 2023). Nonetheless, this still requires a teacher model to cover all the student classes, despite the diverse character sets and writing scripts of many languages.

Tan et al. (Tan et al., 2019) proposed training a multilingual machine translation system by distilling data from monolingual teacher models. The posteriors of pre-trained monolingual machine translation models are used to match the posteriors of the multilingual student model. During each training step, a mini-batch of only one bilingual pair is sampled for student training, and the posteriors from the corresponding teacher model are used to calculate the KL divergence loss. (Cui et al., 2017) et al. trained student models for low-resource languages by distilling knowledge from teacher models, which were trained on the same languages but used different neural network structures. Additionally, the teacher model was trained using features extracted from a multilingual speech recognition system. Xu et al. (Xu et al., 2019) used both multilingual and language-specific monolingual ASR models jointly to train a student model for a specific language.

In a recent study (Fukuda and Thomas, 2021), output posterior distributions from individual monolingual acoustic models were used to train a multilingual ASR model. The student multilingual model includes some shared and a few language-specific layers. The goal of this research is to train a multilingual model with some language-agnostic shared layers, which can be used to initialise a model for an unseen language. Leal et al. (Leal et al., 2021) trained a monolingual student model using multiple cross-lingual teacher models, ensuring the languages shared the same set of tokens.

3.4 Challenges in multilingual speech recognition research

3.4.1 Cross-lingual acoustic-phonetic similarities

In the last two sections, several approaches towards multilingual speech recognition have been presented. The objective of different studies has been to improve the performance of multilingual speech recognition systems. Multilingual setups show improvements for low-resource languages in literature (Abate et al., 2020, Tachbelie et al., 2020a) but no significant reduction in WER is seen for rich-resource languages (Conneau et al., 2021, Feng et al., 2021, Gaur et al., 2021, Hou et al., 2020, Pratap et al., 2020a, Żelasko et al., 2020). Since most of the multilingual models, which include rich-resource languages, are trained on unbalanced data and so the performance degradation of rich languages is attributed to increased confusion for these languages in the multilingual

setup (Pratap et al., 2020a). Conneau et al. mention language interference as the reason for this phenomenon (Conneau et al., 2021). However, no experimental evidence could be found in the literature about these reasons. So, the focus of this research is on investigating acoustic-phonetic similarities among the languages and exploiting them to improve speech recognition of low-resource languages. The relevant previous work has been discussed in Section 1.4 but they are briefly discussed here.

Before the time of ample language resources and computational resources, there were studies on efficient data sharing across languages. This includes heuristic, data-driven and hierarchical (heuristic and data-driven) phone clustering into classes. Data for these clustered classes' models is then shared among languages. However, these studies cover only Context-Independent (CI) acoustic models. As a data-driven approach, Kohler et al. (Kohler, 1996) have measured an entropy-based distance for a set of phonemes tokens. Context-Dependent (CD) acoustic-phonetic similarity has been studied by Imperl et al. (Imperl et al., 2000) and later extended by Le et al. (Le et al., 2006). Both of these studies measure the distances of two polyphones as a weighted sum of the monophonic distances of these polyphones. In (Le et al., 2006), a knowledge-based approach has been applied where a manually made hierarchical graph of phonemes has been used with hand-crafted questions. Each layer (step) in the tree has been assigned different scores (to be used as distances) and has been used to measure the distance between two phonemes. Polyphones are re-clustered to train a multilingual acoustic model. However, a knowledge-based approach needs sufficient manual effort as it requires a hierarchical graph to measure the distance between monophones. It makes enrolling a new language challenging since the same manual effort will be required for each new language. Moreover, phonemes are clustered based on IPA symbols which implies that the cross-lingual phonemes with the same IPA representations have been considered acoustically similar here as well.

Recently, some efforts have been made to interpret the learning of multilingual speech recognition systems (Feng et al., 2021, Żelasko et al., 2020) in the context of phonemes overlap. The Phoneme Error Rate (PER) of each phoneme in monolingual ASR has been compared with that of the multilingual system (Żelasko et al., 2020). However, no monotonic trend has been observed with the growing number of languages a phoneme shared. The authors described this as *"unexpected"* because the phonemes shared by more languages provide more training data and thus the expected error trend would be decreasing. The motivation for research into multilingual speech recognition is based on an assumption that the articulatory representations of phonemes are too close across the languages to be considered language-dependent units (Schultz and Waibel, 2001). Many resource-rich languages such as English, German, Dutch and many other European languages share more than 50% of phonemes with others but still depict degraded performance in multilingual setups (Conneau et al., 2021, Feng et al., 2021, Hou et al., 2020, Pratap et al., 2020a, Żelasko et al., 2020). It calls for investigating the reason for the trends in the performance of multilingual speech recognition systems.

3.4.2 Limited benchmark data sets

One significant challenge in multilingual speech recognition has been the availability of benchmark data sets. Different approaches towards English speech recognition have been using standard benchmark data sets, such as LibriSpeech (LS), Switchboard and WSJ etc to compare their performance with other works (Prabhavalkar et al., 2023). Until 2020, GlobalPhone and Babel data sets have been the 'standard' multilingual data sets (where data from all the languages have been recorded under similar conditions). However, neither of them has been freely available to the public. So, many previous works on multilingual speech recognition have been done on very diverse data sets. Usually, the target languages have been chosen based on mutual intelligibility and overlap in acoustic and lexical content (Datta et al., 2020, Gaur et al., 2021) to build a multilingual speech recognition system. Furthermore, the speech data is sometimes locally collected or an in-house data set (Kannan et al., 2019, Klejch et al., 2022, Pratap et al., 2020a). The studies with technical novelties to overcome multilingual challenges, being on a diverse set of languages makes it difficult to compare them fairly (Gaur et al., 2021). The performance of proposed techniques has been mostly compared with their monolingual baselines (Gaur et al., 2021, Kannan et al., 2019, Klejch et al., 2022). It limits the fair comparison among different approaches. In Table 3.1, a few recent works have been summarised in terms of corpora or data sets used in these works. It shows that very diverse data sets and different numbers of languages have been targeted for different techniques which makes a fair comparison very hard. Furthermore, no criteria are defined for the selection of languages being combined.

Recently, some multilingual speech data sets such as MLS (Pratap et al., 2020b) and CV (Ardila et al., 2020) etc. have been publicised and are freely available. It has encouraged and streamlined multilingual speech recognition significantly. In the following section, all major multilingual speech data sets are revisited.

3.5 Data sets

A few open-resource multilingual data sets have been available in the recent past. Lately, some more resources have been introduced with the paradigm shift towards multilingual speech recognition. In this section, common multilingual speech recognition data sets being used as benchmarks by researchers are described. Though the speech data sets of different languages have been available, each corpus has its recording constraints such as speaker variability, background environment and recording equipment etc. So, two different data sets of two different languages are not a sane choice for building a multilingual speech recognition system. GlobalPhone has been the first prominent step towards collecting a standard speech corpus for multiple languages.

3.5.1 GlobalPhone

GlobalPhone is the first multilingual data set which has been collected under uniform constraints across the languages. The project started in 1995 and an initial release was

TABLE 3.1: Summary of some works on multilingual speech recognition systems, the data sets used and the baselines the work has been compared with

Work	# of languages	Corpora	Duration (Hours)	Baseline
Unsupervised multilingual (wav2vec2.0) (Conneau et al., 2021)	53	Babel, Common Voice,MLS	∼56K	Available mono/multilingual systems
Large-scale e2e multilingual ASR and LID MTL (Hou et al., 2020)	42	Babel, Common Voice, Voxforge, Aurora4 and many others	∼5K	-
Massively multilingual (Pratap et al., 2020a)	51	Facebook in-house data set (publicly video shared by users)	∼16K	Authors' monolingual system
MOE for multilingual ASR (Gaur et al., 2021)	6	"Unknown" data set for English, French and four dialects of Arabic	∼100K	Authors' system without MOE
Streaming e2e model (RNN-T) (Kannan et al., 2019)	9	Google's in-house data set	∼37K	Authors' monolingual system
MTL language-specific phone recognition with multilingual ASR (Sailor and Hain, 2020)	3	Data set released for "low resource speech recognition challenge for Indian Languages" in IS 2018	∼120	Authors' monolingual system
Transliteration based data augmentation (Thomas et al., 2020)	4	Mongolian, Javanese, Dhoulo and Georgian from Babel	∼210	Authors' monolingual system

published in 2002 (Schultz, 2002) when the data of 15 languages was released initially and extended to 5 more later on (Schultz et al., 2013). The objective of GlobalPhone has been to collect uniform speech and text data from multiple languages for the development of Large Vocabulary Continuous Speech Recognition (LVCSR) systems. Until the last publication on this data (Schultz et al., 2013), about 400 hours of transcribed speech of nearly 20 languages recorded by more than 2000 native speakers has been released. For Chinese-Shahmghai, there are no transcriptions available. For Arabic, the data is transcribed in Roman but not in Arabic text.

GlobalPhone speech data sets are mostly clean read speech data where each speaker was asked to read a passage from a news article. About 100 native speakers of each language read a passage for about 20 minutes in a single recording. However, they

TABLE 3.2: Details of GlobalPhone data set. The statistics are collected from the relevant published papers (Schultz, 2002, Schultz et al., 2013) and corresponding ELRA catalogues. The amount of audio is in hours.

Lang.	**Lang. Family**	**Audio (hours)**		**# Spks.**
		Total	**Transcribed**	
Arabic	Afro-Asiatic (Semitic)	35	12	170
Ch-Mandarin	Sino-Tibetan (Sinitic)	31	7.03	132
Ch-Shanghai	Sino-Tibetan (Sinitic)	10	9.83	41
Croatian	Indo-Euro (Slavic)	16	15.58	94
Czech	Indo-Euro (Slavic)	29	7.87	102
French	Indo-Euro (Romance)	25	2.93	100
German	Indo-Euro (Germanic)	18	18.32	77
Japanese	Isolate	34	0.95	149
Korean	Isolate	21	20.78	100
Portuguese	Indo-Euro (Romance)	26	2.17	102
Russian	Indo-Euro (Slavic)	22	2.417	115
Spanish	Indo-Euro (Romance)	22	21.3	100
Swedish	Indo-Euro (Germanic)	22	21.67	100
Tamil	Dravidian		17.9	98
Turkish	Turkic	17	16.9	100
Bulgarian	Indo-Euro (Slavic)		20.98	0
Hausa	Afro-Asiatic (Chadic)		8.73	100
Polish	Indo-Euro (Slavic)		23.7	102
Thai	Kra-Dai (Tai)		23.1	98
Ukrainian	Indo-Euro (Slavic)		13.87	119
Vietnamese	Austroasiatic		1.42	160
Swahili	Niger-Congo			70

were instructed to pause after each sentence which helped in post-processing. During the post-processing, the sentences with significant errors were discarded and the minor mistakes were resolved in the text. An analysis of the top spoken languages around the world was carried out and nine out of the top 15 were selected to be part of the GlobalPhone. All the languages were selected considering the considering the following characteristics.

- Speakers' population size
- Economic and political relevance
- Geographic coverage: Asian, European, African, American and Indian etc.
- Phonetic coverage
- Script variety (orthography such as Latin, Cyrillic and Arabic)

In its initial release, GlobalPhone only released audio data along with its transcriptions but the latest release (Schultz et al., 2013) includes dictionaries, text corpus and n-gram language models as well. The languages of the GlobalPhone data set are given

in Table 3.2. The languages above the dashed lines were released in the initial release of GlobalPhone corpus (Schultz, 2002) while the others (below the dashed line) were released later. Except the Swahili, all the new languages (below the dashed line) have been discussed in the later release paper. However, Swahili is only found in the ELRA catalogue.

Most of the languages have been recorded in a clean office environment using the same *Sennheiser 440-6 close-speaking microphone* as the recording equipment. Only the subject and the recorder were in the room during the recording. Only the data of *Hausa* was recorded using the headsets. As the participants were asked to read the news article and were allowed to read the text beforehand, the data is read speech and very less prone to mistakes. Furthermore, the minor mistakes have been resolved during the post-processing.

The GlobalPhone corpus has been designed to be uniform across the languages in terms of data quantity and quality, environment and word and phonemes transcriptions. Audio transcriptions are provided on the word level, however, the dictionaries of all the languages consist of the same phonetic representations (IPA). Furthermore, the corpus has been designed to cover diverse phonetic, scripts and morphological variations. It includes tonal sounds (from Chinese, Vietnamese and Thai), consonantal clusters (from German), palatised phones (from Russian) and pharyngeal sounds (from Arabic) etc. Different languages cover phonographic segmental, consonantal, syllabic and featural scripts. Furthermore, GlobalPhone includes agglutinative languages (Korean and Turkish), compounding languages (e.g. German) and scripts which lack word segmentation at all (e.g. Chinese and Thai). Being a very rich and diverse data set, GlobalPhone is a very good basis for research in various domains such as multilingual and cross-lingual speech recognition, multilingual speech synthesis, speaker identification and language identification etc.

Though the GlobalPhone corpus has been a big step towards multilingual speech resources, it is not an open resource which hinders many researchers from using it. However, there was no other multilingual speech data set collected in such a careful manner and it remained a prominent benchmark for multilingual speech technologies for a while until the other resources have been made available to the scientists.

3.5.2 Babel

The Babel data set has been collected under The Babel project by the Intelligence Advanced Research Project Activity (IARPA). According to its documentation, the objective of the Babel project has been to develop robust and agile speech methods to;

- apply rapidly to any new language
- make keyword search effective over massive amounts of speech data from all the languages

Contrary to the GlobalPhone data, the Babel project intended to collect the data in real-time noisy environments rather than quiet and controlled environments. The goal of this data set has been to build speech recognition systems for a much larger set of languages using much smaller yet noisy data.

The Babel program was divided into four phases with the goal of data collection of a certain number of languages. Each phase included the data release of a *"surprise"* language as well and a limited time was allotted to build a keyword search system for the *surprise* language. For example, the very first phase (base phase) included four main languages which included Cantonese, Turkish, Pashto, Tagalog and one surprise language. The build time of a keyword search system was set to 4 weeks followed by 1 week for evaluation for the base phase period. The plan involved reducing the build time gradually for later phases.

The data set has been designed on the following basis

- **Diverse languages from the outset:** Data of very diverse languages was collected from the countries where those languages had been spoken.
- **Real recording conditions:** Recordings were done in a variety of environmental conditions such as in a moving vehicle or a cafe on the street etc. Though most of the data has been recorded over the mobile phone channel, several other devices have also been used to record some amount of the data such as hands-free and tabletop microphones etc.

During each phase, the data set was delivered in two packs i.e. Limited Language Pack (LLP) and Full Language Pack (FLP) which contain limited and full amounts of the recorded data respectively. Though most of the data was recorded using mobile phones in noisy environments, limited read speech clean data was also recorded. Each data set included audio data along with orthographic transcriptions. Furthermore, the hand-crafted pronunciation dictionaries have also been provided with mapping words to their Roman versions and their pronunciations in the Sampa format. Syllable breaks, word boundaries, and primary and secondary stresses have also been marked. The data has been released via the Linguistic Data Consortium (LDC)[2]. The data was annotated carefully maintaining the uniformity of annotation rules which have been well documented and available in a separate document. However, a lot of tags have been used to define the anomalies in the speech such as hesitation, idiosyncrasies and partial words etc. Babel data sets include 25 languages in total and the details are given in the Table 3.3.

Though the amount of average audio data per language is around 218 hours in the Babel data sets, each data set involves a lot of non-speech segments due to recording in real-time environments. For example, the analysis of the Turkish language shows that 97 hours out of 213 hours consist of non-speech (silence or noise) which is about 45% data

[2] https://catalog.ldc.upenn.edu/

TABLE 3.3: Details of Babel data set (languages, their families and the total amount of audio data). Languages marked with '*' are tonal languages.

Lang.	**Lang. Family (Region)**	Audio (Hrs)
Amharic	Ethiopian Semitic (Ethiopia)	204
Assamese	Indo-European (Assam)	205
Bengali	Indo-European (Bengal)	215
Cantonese*	Sino-Tibetan (Southeastern Chin)	215
Cebuano	Austronesian (Philippines)	200
Dholuo*	Nilotic (Kenya and Tanzania)	204
Georgian	Kartvelian (Georgia)	190
Guarani	Tupian (South America (Paraguay))	198
Haitian	Haiti	203
Igbo*	Niger-Congo (Nigeria)	207
Javanese	Austronesian (Java)	204
Kazakh	Turkic (Kazakhstan)	203
Kurmanji	Indo-European (Northern Kurdish)	203
Lao*	Kra-Dai (Lao, near Thailand)	207
Lithuanian*	Indo-European (Lithuania)	210
Mongolian	Mongolic (Mongolia)	204
Pashto	Indo-European (Afghanistan, Pakistan)	244
Swahili	Niger-Congo (East Africa (Kenya, Sudan,...))	350
Tagalog	Austronesian (Philippines)	213
Tamil	Dravidian (India/Sri Lanka)	350
Telugu	Dravidian (India)	201
Tok Pisin	English Creole (Papua New Guinea)	200
Turkish	Turkic (Turkey)	213
Vietnamese*	Austroasiatic (Vietnan)	201
Zulu*	Niger-Congo (South Africa, Zimbabwe etc.)	211

of the total amount. Similarly, 40% of data of Tagalog language is non-speech and this is the case for almost all the languages.

To annotate the various speech and non-speech events (e.g. cough, laugh and hesitations etc.) in the recordings, various tags have been used. As the Babel data set is used for several experiments in this work, further details on the Babel data set tags and their handling for this work are discussed in Appendix A.

Though the Babel data set was developed primarily for keyword search, it is an excellent resource to build or evaluate low-resource challenging speech technologies as it consists of real-time noisy low-resource speech data. It can also be used for multilingual speech recognition though the error rates on the Babel data set are quite high yet because of the very challenging nature of the data.

3.5.3 CommonVoice

CommonVoice multilingual speech data sets are part of a Mozilla project[3] which is intended to collect speech data of a huge number of languages to build multilingual

[3] https://commonvoice.mozilla.org/en

speech recognition systems. However, the data can also be used for other tasks such as language identification. The databases are being actively updated including more data and languages on the online portal. However, the discussion in this section is based upon the data set paper published in 2020 (Ardila et al., 2020).

In the CommonVoice project, the data is collected and validated through crowd-sourcing i.e. people from diverse languages and backgrounds volunteer to record and validate the data of their languages of interest. The data to be recorded is extracted from Wikipedia (Ardila et al., 2020) articles published in the respective language. For speakers, text sentences appear on the screen which they have to read. It means that the data is read speech but there is no guarantee that it is a clean speech as the choice of background environment is up to the speakers. Furthermore, the recording equipment is also not the same as speakers can record using whatever setup they have such as laptop mic, headsets, hands-free and table-top mic etc. Validation of recorded utterances is also done by crowd-sourcing based on the voting. A recording utterance along with its transcription appears before a listener who votes if the recorded speech is correct or not. Each utterance appears before three listeners and the utterances receiving two upvotes are considered correct. Whereas the utterances with two downvotes are deemed invalid. Only the validated utterances are included in the officially released training, development and testing sets while the unprocessed data or the utterances with an insufficient number of votes are released under the '*other*' category.

The validated data is bucketed into train, development and test sets using statistical power analysis. The number of utterances in the development and test sets is equal to the number needed to achieve a confidence level of 99% relative to the number of utterances in the training set. It is implemented in a way that there is no speaker overlap among these sets and the repetition of utterances with the same text is removed. The released audio data is mono-channel, 16-bit MPEG-3 format and 48kHz sampling rate. As this is the latest data set with an aim to build a modern speech recognition system, only audio files along with their transcriptions are included in the data set. No phonemic/phonetic dictionaries are included in the data set. As the data is moderately controlled (read speech and most likely to be indoors) and validated (through crowd-sourcing), CommonVoice data is a fairly plausible resource for building multilingual speech recognition systems.

Given its magnitude of nearly 25K hours from 50K speakers of 38 languages, it is the first open resource towards multilingual speech recognition. The aforementioned numbers are taken from the last published paper (Ardila et al., 2020), the portal (Online) mentions data recording of 28.7K hours with 19.2K hours of validated speech of 114 languages. Although GlobalPhone and Babel have been used by several previous researches, CommonVoice is well-received because of its free availability and is being used as a benchmark multilingual data set by several modern studies (Conneau et al., 2021).

Since the motivation of this work is cross-lingual acoustic-phonetic similarities whereas

CommonVoice is a better fit for end-to-end models and does not have words-to-phonemes mapping dictionaries, it is not used for experimentation in this work. As the corpus consists of nearly 114 languages to date, the list of the languages is avoided here. However, the latest details of the corpus can be found on the dedicated web page (Online).

3.5.4 VoxLingua107

VoxLingua107 (Valk and Alumäe, 2020) is a massively multilingual speech data of 107 languages which have been developed with an aim for language recognition tasks. So, it only contains speech data and corresponding language tags and does not include transcriptions of the audio. As it does not contain transcriptions, it cannot be used to build speech recognition systems. However, it has been used for audio-only training (such as pre-training multilingual self-supervised models). So, the data collection of VoxLingua107 is briefly discussed here.

Speech data of VoxLingua107 has been collected from YouTube videos in different languages. Random phrases are used for each language to search the videos on the platform. These random phrases are generated from Wikipedia articles. A list of the most important search phrases of a language is extracted using Term Frequency Inverse Document Frequency (TF-IDF) over a huge number of Wikipedia articles written in the corresponding language. Phrases with only 3 words have been finally selected as the optimal ones for video search. Polyglot Python package has been used for language identification on the search phrases to filter out any phrases containing word(s) from any other language. These phrases have been used for search on YouTube platforms.

Even the carefully generated search phrases can result in false positives i.e. the videos in some other language. Additionally, there might be videos with the title written in the target language but content in some other language. So, language identification has also been applied to the title, description and other meta-data (if available) of the videos to make sure they contain the audio in the target language. The audios have been segmented into shorter clips using LIUM SpkDiarization toolkit (Rouvier et al., 2013). It has been constrained to result in the segments of the duration between 2 to 20 seconds. Non-speech segments such as noise, music or silence have been removed before segmentation. A total of 14,044 hours of data from about 78 thousand videos have been extracted. After segmentation, the data contains 3.5 million utterances of 107 languages. The data has been validated by crowd-sourcing where people validated if the language spoken in audio is the same as the tagged language.

3.5.5 Multilingual LibriSpeech

Multilingual LibriSpeech is another large-scale multilingual speech data which is derived from read audiobooks from LibriVox (Pratap et al., 2020b). LibriVox is also a CommonVoice-style platform where people can contribute towards data collection by reading books. So, the MLS data set is also a read speech data but the recording equipment and environment depends on the speaker's choice. Though mostly the data

is expected to be clean, there can be segments with background noises or very low Signal-to-Noise Ratio (SNR). LibriVox contains massive amounts of audio recordings in multiple languages. Though currently LibriVox has audio recordings in about 47 languages, the duration of LibriVox audio recordings for the languages with more than 50 hours of recordings are given in the Table 3.4. The statistics are extracted using LibriVox APIs[4].

As it is evident from the Table 3.4 LibriVox contains impressively large amounts of recorded data in a lot of languages but it comes with some challenges. In LibriVox data, each recording session consists of one chapter of a book. Whereas smaller audio segments are needed to develop speech technologies due to computation limitations and training constraints. However, the challenge is to chunk the large audio clips into smaller segments and annotate the recordings. It would be a laborious task if done manually. MLS introduces a semi-supervised approach to segment and annotate the data and releases nearly 50K hours of speech data from 8 languages. These eight languages are selected from the top ten languages in LibriVox data (Table 3.4). About 44.5K hours of data belong to the English language while the rest of the seven languages make up 6K hours portion of the data. Data is not uniformly distributed even among the rest of the seven languages. The details of MLS languages and speech data are given in Table 3.5.

[4] https://librivox.org/api/info {Last accessed on: 15th November, 2023}

TABLE 3.4: Details of LibriVox total audio data set (includes unlabelled data). The languages, duration of available audio data and the number of recorded books are given here

Language	Duration (Hrs)	# of books
English	98334.82	16696
German	4124.48	721
Spanish	2319.26	432
Dutch	2303.13	210
French	2169.37	304
Multilingual	638.03	165
Portuguese	472.93	110
Italian	361.19	80
Russian	314.71	81
Polish	164	32
Latin	138.93	22
Church Slavonic	136.42	8
Hebrew	129.07	25
Japanese	97.67	38
Ancient Greek	92.62	46
Greek	69.72	26
Finnish	67.66	18
Chinese	65.24	29

TABLE 3.5: Languages and their durations in train, dev and test sets of MLS data set. All the durations are in hours.

Language	Train	Dev	Test
English (en)	44659.74	15.75	15.55
German (de)	1966.51	14.28	14.29
Dutch (nl)	1554.24	12.76	12.76
French (fr)	1076.58	10.07	10.07
Spanish (es)	917.68	9.99	10
Italian (it)	247.38	5.18	5.27
Portuguese (pt)	160.96	3.64	3.74
Polish (pl)	103.65	2.08	2.14

MLS authors have selected eight languages from LibriVox which includes English, German, Dutch, French, Spanish, Portuguese, Italian and Polish. The LibriVox recordings of 48KHz have been down-sampled to 16KHz. For audio segmentation, pre-trained acoustic models have been used to get the Viterbi token sequence along with their timestamps of the audio files. As the MLS data is released by Facebook (now Meta), they have used their in-house data for these languages to train language-dependent streaming ASR systems. Streaming models have been used to be more efficient for long segments (of LibriVox). Details about the speech recognition systems can be found in the original paper (Pratap et al., 2020b). The longest silence section is detected within 10 to 20 seconds from the start of the audio to split it. The audio is cut in the middle of the detected longest silence segment to chunk it. If there is no silence segment in 10-20 seconds, the audio is split at 20 seconds mark. This process implies that all the segments are between 10 to 20 seconds in duration which ensures sufficient spoken words in each segment and no extra-ordinary long segments. It helps with the semi-automated transcribing process described in the next paragraphs.

The MLS data is transcribed in a lightly supervised manner. First, the data is decoded through the aforementioned in-domain acoustic models along with language-dependent language models to generate pseudo-labels for the audio files. Mostly, 4-gram language models are used which have been trained on the text corpus of transcriptions of the in-house acoustic models training set. As the MLS corpus consists of read books, the text of the books would be needed for transcriptions which is itself challenging. Text for the English books has been extracted using HTML parser where the books are available online. For other languages, it has been challenging to find the text of recorded books and various approaches have been used to get the data such as PDF to text libraries and manually searching the books etc. Such challenges and the approaches to mitigate them have been discussed in detail in the paper (Pratap et al., 2020b). However, even if the text from all the books is available, another challenge is to normalise the text data to make it uniform across the books and languages and find the text read in each chunked audio segment from the whole book. For normalisation, all the unnecessary characters such as punctuation markings, superscripts and subscripts, emojis and escape symbols

etc have been removed. Furthermore, the end-of-line hyphen for partial words has also been removed and the partial parts of words have been joined together. Additionally, the characters outside the Unicode range of a language's characters have also been removed.

Once the text is normalised, each book has been chunked into 1250-word documents with a stride of 1000 words and 250 words overlap. For each chunk, the most relevant (chunked) text document is selected by measuring the TF-IDF similarity between chunked documents and the hypothesis from the speech recognition system. Having the closest text document, Smith-Waterman alignment (Smith and Waterman, 1981) is used to find the best match sub-sequence. In this way, a candidate label is generated for each audio segment. Candidate transcripts are filtered out if the word error rate between pseudo labels and the candidate labels is higher than 40%.

Data is split into training, development and test sets ensuring that

- there is no speaker overlap,
- speakers are balanced in terms of gender and duration in the test and development sets,
- there is sufficient audio in the test and development sets to validate the performance of ASR systems.

Several post-processing steps have been carried out including training of a model for speaker classification to ensure the aforementioned constraints given the data complexity. Please refer to the data set paper for further details on what kind of challenges were there and how they have been resolved (Pratap et al., 2020b). As the MLS data set has been released recently with the aim of a resource for the development of end-to-end speech technologies, it only consists of audio data along with the transcriptions and language models. The released language models have been trained using normalised text from all the books of *Project Gutenberg*[5]. The training text corpus also includes the books which are not in MLS audio data sets. Text is normalised before training a language model and 3-gram and 5-gram models are trained.

Since the transcriptions of the audio data are extracted in a semi-supervised way, there might be errors in the labels. So, MLS data sets include supervised data of 10 hours duration for each language. Furthermore, labels of development and test sets are also verified and corrected (where necessary) manually.

3.5.6 Few-shot Learning Evaluation of Universal Representations of Speech (FLEURS)

Fleurs (Conneau et al., 2022) is a recently released dataset with an aim to provide a few-shot data set for the evaluation of self-supervised speech recognition models. Since the

[5] https://www.gutenberg.org/

primary objective of Fleurs data set is about self-supervised models which are beyond the scope of this work, it is only described briefly here.

The Fleurs dataset is a multilingual speech dataset which serves as a crucial benchmark for evaluating few-shot learning in the context of universal representations of speech. It is an n-way parallel speech dataset comprising 102 languages. The dataset is built on top of the machine translation FLoRes-101 benchmark and provides approximately 12 hours of speech supervision for each language. Researchers can use FLEURS for various speech tasks, including Automatic Speech Recognition (ASR), Speech Language Identification (Speech LangID), translation, and retrieval.

The Fleurs dataset is built using the FLoRes-101 dataset (Goyal et al., 2021), which comprises 3,001 sentences originally extracted from English Wikipedia. These sentences have been translated into 101 languages by professional human translators. Since the FLoRes-101 test set is not publicly available, only the dev and devtest sets, totalling 2,009 sentences, are used in Fleurs. These sentences have been re-split into new train, dev, and test sets, containing 1,509, 150, and 350 sentences, respectively. For each sentence in the 102 languages (101 from FLoRes-101 plus English), three recordings by three different native speakers have been collected.

After recording the audio, each file is evaluated to determine if it accurately corresponds to the input sentence. Invalid recordings are discarded, resulting in 0 to 3 recordings per sentence in the final dataset. The recordings are made in uncontrolled environments, meaning they may have clean or noisy backgrounds. The speech sampling rate is 16 kHz. For transcriptions, translations from human translators of the FLoRes-101 data set are used.

3.6 Summary

Section Section 3.4.1 discussed previous work showing that a multilingual speech recognition system does not always guarantee improved performance compared to a monolingual system. Sometimes, the performance of a multilingual ASR system is degraded, even if the participating languages have significant phoneme overlap. This overlap is typically measured by the number of shared IPA-represented phonemes across the languages. However, some studies have shown that multilingual ASR systems can improve speech recognition performance compared to monolingual systems. It is argued that this improvement depends on the choice of languages combined to train the multilingual ASR system.

Section Section 3.4.2 highlighted that, due to the lack of freely available multilingual benchmark datasets, many previous works on multilingual ASR have used diverse non-standard or in-house datasets. Additionally, the background conditions of the different languages often vary. Because of this lack of standard datasets, many system improvements are compared with the work's own baseline, making many comparisons unfair or unclear.

Considering these observations from the literature, our objective is to investigate the reasons for performance degradation in multilingual ASR systems, devise a better measure of language closeness (compared to IPA-represented phoneme sharing), and improve low-resource speech recognition using standard benchmark datasets for fair comparisons.

Part II

Contributions

Chapter 4

Cross-lingual representations sharing

As described earlier in Section 1.4, the motivation of research into multilingual speech recognition is based on the assumption that the articulatory representations of identically IPA-represented phonemes across the languages are close enough to be considered language-independent units (Schultz and Waibel, 2001). It implies that for multilingual ASR training, a phoneme shared by more languages provides more training data and thus the error rate of that phoneme must be reduced in a multilingual setup (Żelasko et al., 2020). However, several languages with substantial cross-lingual phoneme overlap exhibit poorer performance in multilingual setups (Conneau et al., 2021, Hou et al., 2020, Pratap et al., 2020a) when compared with their monolingual counterparts. Two potential reasons for this trend have been suggested in the literature;

1. **Increased confusion:** Degradation in the performance of multilingual ASR systems for some languages is attributed to increased confusion for them compared to their monolingual models (Pratap et al., 2020a). Most multilingual models, including both high and low-resource languages, are usually trained on very skewed training data across the languages. For monolingual model training, a resource-rich language has a lot of in-domain training data. However, in the case of multilingual model training when the data from other languages is also included, the model is trained to optimise the performance for all the languages. So, the performance of a multilingual ASR system might decrease for resource-rich languages compared to their language-dependent ASR systems.

2. **Language-units interference:** *Conneau et al.* (Conneau et al., 2021) mention language interference as the reason for the increased error rate. During multilingual ASR training, cross-lingual data is also included which might have the same phonemes from other languages. Though the same phonemes from other languages have the same IPA representations, the underlying acoustic realisations might be

different. This cross-lingual interference increases confusion for the model to learn a phoneme properly given input signal which is regarded as language interference.

However, no experimental evidence could be found to dig out the reasons for the performance degradation from multilingual ASR systems for some languages. The argument of unbalanced sampling can be validated by simply sampling the data uniformly and very limited studies have been done to explore 'language interference'.

In this chapter, experimentation is done to train baseline ASR systems to observe the trends in the performance of multilingual ASR systems. A post-analysis is done to get insights into these trends and cross-lingual relationships. In Section 4.1, the relevant literature on cross-lingual similarities in the context of multilingual acoustic modelling is revisited. Section 4.2 discusses the grapheme and phoneme-based baseline speech recognition systems. Section 4.3 and Section 4.4 describes the experimental setup and results. A detailed discussion on the performances of ASR systems and to analyse cross-lingual relations is done in 4.5.

4.1 Background

Earlier studies, in the context of efficient data sharing among languages for multilingual setup, are on context-independent acoustic models (Kohler, 1996). Having 3-state HMM phoneme models for phonemes of English, German and Spanish languages, a relative-entropy-based distance measure has been used to estimate the similarity between two phonemes. These phonemes can be of the same language or from different languages. The distance measure has been calculated with the potential goal of clustering the closer phonemes for modelling purposes. Let λ_i and λj be phoneme models of two phonemes i and j, and their set of observations X_i and X_j. Distance between phoneme i and j has been calculated as;

$$\mathcal{D}(\lambda_i, \lambda_j, X_i, X_j) = \frac{1}{2}(d(\lambda_i, \lambda_j, X_i) + d(\lambda_j, \lambda_i, X_j))$$

where

$$d(\lambda_i, \lambda_j, X_i) = \log p(X_i|\lambda_i) - \log p(X_i|\lambda_j)$$
$$d(\lambda_j, \lambda_i, X_j) = \log p(X_j|\lambda_j) - \log p(X_j|\lambda_i)$$

However, models have been trained for the phonemes without considering their contextual phonemes in the study. Context-dependent acoustic-phonetic similarity has been studied by *Imperl et al.* (Imperl et al., 2000) and later extended by *Le et al.* (Le et al., 2006). With contextual modelling of phonemes, the number of possible polyphonemes rises exponentially. For example, considering only two contextual phonemes (triphones), the number of possible triphones raises to (n^3) for n phonemes. It implies that need to train n^3 triphone HMM models or a neural network with n^3 output classes. To reduce the number of models or output classes, (Imperl et al., 2000) have proposed a metric to

measure the distance between two triphones. Consider $\varphi_i^L - \varphi_i + \varphi_i^R$ and $\varphi_j^L - \varphi_i + \varphi_j^R$ two triphones, the proposed distance measure is the weighted sum of individual distances between both left, centre and right phonemes.

$$\begin{aligned} S_{TRI}(\varphi_i^L - \varphi_i + \varphi_i^R, \varphi_j^L - \varphi_i + \varphi_j^R) = & W_L \cdot S(\varphi_i^L, \varphi_j^L) + \\ & W_C \cdot S(\varphi_i^C, \varphi_j^C) + \\ & W_R \cdot S(\varphi_i^R, \varphi_j^R) \end{aligned}$$

where W_L, W_C and W_R are the weights assigned to the individual distances of left, centre and right phonemes respectively. Though the individual distance between any two phoneme $S(\varphi_i, \varphi_j)$ can be any distance measure, the one used by the authors is given as;

$$\begin{aligned} S(\varphi_i, \varphi_j) = \frac{1}{2} \sum_{k=1}^{N} [& |c(\varphi_i, \varphi_k) + c(\varphi_j, \varphi_k)| \\ & - |c(\varphi_i, \varphi_k) - c(\varphi_j, \varphi_k)|] \end{aligned}$$

where N is the number of phonemes and $c(\varphi_i, \varphi_j)$ is the number of confusions between phonemes φ_i and φ_j. Triphones with a distance below a certain threshold are then clustered for modelling. Both of the aforementioned studies (Imperl et al., 2000, Kohler, 1996) measure the distance between two polyphones as a weighted sum of monophonic distances of these polyphones. Le et al. (Le et al., 2006) have measured the distance between polyphones rather than triphones using a similar concept. However, a knowledge-based tree is constructed first for phonemes based on their linguistic properties. For instance, the root node decides if a phoneme is a vowel or consonant. The first child, in the case of consonants, makes decisions based on articulatory properties such as plosives and bilabials etc. In the case of vowels, these properties are vowel characteristics such as the position of the tongue (front or back etc). The distance between any two phonemes is then measured based on their distance in their levels in the tree. Since this approach is knowledge-based, it needs considerable manual effort and linguistic knowledge of the languages.

Recently, efforts have been made to interpret the learning of multilingual speech recognition systems (Feng et al., 2021, Żelasko et al., 2020). The Phonetic Token Error Rate (PTER)[1] of each phoneme in monolingual ASR has been compared with that of multilingual system (Żelasko et al., 2020) (shown in Figure 4.1). Each dot in the box plot of Figure 4.1 represents a specific phoneme and the horizontal axis shows the number of languages it is being shared with. The vertical axis shows the difference in error rates of a monolingual and multilingual ASR for a phoneme. Since the phonemes shared by more languages have more training data for multilingual ASR training, their error rate is expected to be lower (Żelasko et al., 2020). However, no monotonic trend has been

[1] PTER is similar to PER except for the tonal languages where PER considers a vowel and its tone are considered two different phonemes but PTER deems it as a single phoneme.

observed with the growing number of languages the phonemes shared. The authors have described this as an *"unexpected"* behaviour.

Furthermore, as discussed in Section 3.4.2, most works on multilingual ASR do not specify the criteria on which the languages are chosen for a multilingual setup. Sometimes the chosen languages are regarded as 'close languages' without an explicit definition of 'closeness'. Languages are usually regarded as closer based on the size of the overlapping set of phonemes across the languages (Datta et al., 2020). However, degradation in the performance of multilingual ASR systems for the languages with considerable phoneme overlap (Conneau et al., 2021, Hou et al., 2020, Pratap et al., 2020a) suggests that the size of overlapping phonemes set across two languages does not inform much about their acoustic similarities.

Motivated by the fact that many languages with significant phonemes overlap pose performance degradation in the multilingual setups (Conneau et al., 2021, Feng et al., 2021, Hou et al., 2020, Pratap et al., 2020a, Żelasko et al., 2020), the objective of this research is to study cross-lingual acoustic-phonetic similarities and their impact on multilingual speech recognition setups. The aim is to analyse whether the identically IPA-represented phonemes across different languages are acoustically similar too. It also explored how the cross-lingual acoustic-phonetic similarities impact the performance of multilingual ASR systems. Data-driven approaches are focused to investigate the phonetic unit representations and measure cross-lingual similarities.

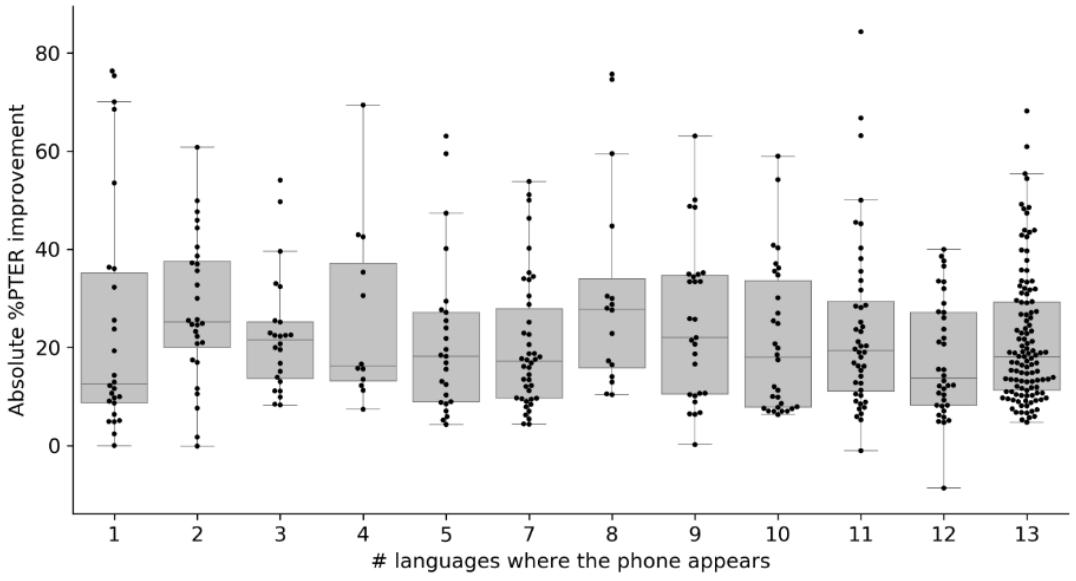


FIGURE 4.1: Absolute improvement in phonetic token error rate per token (multilingual-monolingual) (*figure credits*: (Żelasko et al., 2020)). Each dot in the box plot depicts a phonetic token. The horizontal axis represents the number of languages a token is being shared with and the vertical axis is the absolute difference in the error rates of the monolingual and multilingual ASR systems.

In Section 4.2, baseline monolingual and multilingual ASR systems are described which are trained to observe the shift in the performance of a few resource-rich languages.

4.2 Baseline speech recognition systems

A baseline multilingual ASR system is trained for foundational experiments. Training data is uniformly sampled from all the languages to validate the argument that the skewed training data causes degraded performance from multilingual ASR systems (Pratap et al., 2020a). Since the main objective of this study is to analyse acoustic-phonetic similarities, phoneme-based speech recognition systems are a better choice for analysis. However, such systems require lexicons (usually handcrafted) for each language which are not available for a lot of languages. So, initially, grapheme-based hybrid DNN-HMM speech recognition systems are trained with equal training data for all the languages. Hybrid DNN-HMM models are preferred over end-to-end models for analysis to avoid the effect of implicit language modelling in e2e models. Later, phoneme-based models are also trained for fewer languages and an analysis is done to understand cross-lingual acoustic phonetic similarities. Both, grapheme-based and phoneme-based ASR systems, are the same except that the pronunciation dictionary for a grapheme-based ASR is simply a mapping of words to their constituent characters whereas it is words to phonemic mapping in the latter case. Acoustic models in all these experiments are GMM-HMM and hybrid DNN-HMM models, and language models are 3-gram count-based models (unless explicitly mentioned).

4.3 Experimental setup

4.3.1 Data set

The experiments in the rest of the sections of this chapter are carried out using portions of the MLS data set. This is a big multilingual open resource data set which is quite rich in terms of duration, number of speakers, high quality of speech and easy availability. This data set covers eight European languages yielding more than 50 thousands hours of speech data by nearly 6000 speakers. MLS data set has been discussed in detail in Section 3.5.5. ISO 639-1 codes[2] of languages are used for representation of MLS data set languages throughout this thesis.

For the baseline experiments, a small portion of the MLS data set is sampled. However, both the train and test sets consist of the same duration for all the target languages. The motivation for uniform sampling from all the languages is to assess the 'increased confusion' argument (described at the start of this chapter) as the reason for the performance degradation of rich resource languages in a multilingual setup. For training and evaluation of grapheme-based ASR systems, approximately 15 hours and 2 hours are randomly sampled from train and eval sets of MLS corpus respectively. However, training data is doubled (30 hours) for the training of phoneme-based systems while the

[2] https://www.loc.gov/standards/iso639-2/php/English_list.php

test set remains the same. Speaker distribution is not monitored during data sampling assuming that random sampling would uniformly sample the speakers from data.

4.3.2 Acoustic modelling

Initially, a monophone GMM-HMM acoustic model is trained using uniform alignments. For all the GMM-HMM models in this work, three-state HMMs are used for monophones or polyphones (biphones and triphones etc) training. Remember that the term monophone or any polyphone is only used to be consistent with Kaldi documentation and previous work using Kaldi. The trained models are, however, phoneme-based (not the phone-based). The model is trained iteratively, force-aligning the transcription using the trained model and then retraining the acoustic model in each iteration. After various iterations (nearly 40), a triphone model is trained with the same iterative approach and using alignments from the final monophone model (for the first triphone iteration). Since triphone training after transformation proves to perform better, Maximum Likelihood Linear Transformation (MLLT) (Gales, 1999, Gopinath, 1998) is estimated using Linear Discriminant Analysis (LDA) on spliced MFCC features. Then a triphone model is trained again using the same iterative approach. MFCC features are transformed into speaker-adapted features using feature-space Maximum Likelihood Linear Regression (fMLLR) transformation (Gales, 1998). Using fMLLR transformed features has shown significant improvement in ASR performance (Gales, 1998, Parthasarathi et al., 2015). So, fMLLR transformed features are used to train a speaker-adapted model (referred to as Speaker Adapted Training (SAT) training). Alignments from the final model of the last triphone training are used as initial alignments for SAT training.

Using neural networks for the estimation of emission probabilities has been proved to outperform GMM-HMM models (Tachbelie et al., 2020b). So, two different hybrid DNN-HMM models are trained and their performances are compared. A DNN model is trained with greedy layer-wise supervision (Zhang et al., 2014) which outputs log-likelihood of HMM states (this model is termed as "*nnet2*" here). The output log-likelihoods are used as emission probabilities. In greedy layer-wise supervised training, the layers are added gradually. At the start of training, only one hidden layer is added and trained for a few steps. Then the second layer is added with randomly initialised weights. Previously learnt weights of the output layer are also randomly initialised again. The second layer is trained for a few steps before adding the third layer. This process continues until all the layers are added and the whole model is trained in the following epochs. The *nnet2* model, experimented with here, consists of six layers with an equal number of neurons.

Another hybrid DNN-HMM model is trained which is a TDNN network and trained with MMI (Povey et al., 2016) objective without using lattices (thus called Lattice-Free Maximum Mutual Information (LF-MMI)). 40 MFCCs are extracted for each frame of the speech signals using a window size of 25 ms and a shift of 10 ms. These features are then fed to DNN which consists of 12 factorised TDNN (TDNN-F) layers (Povey et al., 2018). Each TDNN-F hidden layer is of dimension 1024, factorised with a linear

TABLE 4.1: Nomenclature used in experiments sections to identify different training methods and models

Notation	Description
monophone	Monophone acoustic model
triphone	Triphone acoustic model
lda+mllt	Triphone acoustic model after LDA-MLLT features' transformation
sat	Triphone acoustic model with speaker adaptive training
nnet2	DNN-HMM acoustic model with greedy layer-wise training (Zhang et al., 2014)
lf-mmi	DNN-HMM acoustic model with DNNs trained using LF-MMI criteria (Povey et al., 2016)
mono	Monolingual acoustic and language model
multi	Multilingual acoustic and language model (unified data of all languages)
mono-lm	Multilingual acoustic model and monolingual language model

'bottleneck' dimension of 128. The model is trained with a cross-entropy objective function and the output of the model is the likelihood of the tied HMM states.

Table 4.1 summarises the nomenclature used to point to the aforementioned modelling techniques and models during further discussion in this chapter. All the ASR systems are built using the Kaldi toolkit (Povey et al., 2011).

4.3.3 Language modelling

In the following discussion, terms *monolingual language model* and *multilingual language model* are frequently used. A monolingual language model is trained using the text corpus of only the language of interest. On the other hand, a multilingual language model is a universal language model developed by mixing text corpora of all the languages. Both of them are 3-gram language models and only the transcriptions of the training set are used for the training of these models.

4.3.4 Evaluation metric

Word Error rate (WER) is the universally accepted metric to measure the performance of speech recognition systems. End-to-end systems, however, report performance in Character Error Rate (CER) sometimes. For the experiments, the performance of developed ASR systems is reported in terms of WER, CER and PER where applicable. Reference and hypothesis transcriptions are aligned using dynamic string alignment and the token error rate is calculated using the formula given in Equation 4.1 where the tokens could be words, phonemes or characters. The only difference is that for each kind of error, alignments are on respective token levels i.e. words, characters or phonemes.

$$\text{Token Error Rate} = \frac{S + I + D}{N} \tag{4.1}$$

where S,I and D are the number of substitutions, insertions and deletions in hypothesis text while N is the total number of tokens in the reference or ground-truth transcription.

MLS data set is transcribed on word level which can easily be decomposed into characters. So calculations of WER and CER are straightforward. However, for measuring PER, phoneme-level transcriptions are required. Phonemic alignments are automatically generated by force alignments using the trained monolingual models.

4.3.5 Tools

Speech recognition systems, reported in this chapter, are built using the widely used Kaldi toolkit (Povey et al., 2011). Language modelling is done using the SRILM (Stolcke, 2002) toolkit.

4.4 Experiments and results

4.4.1 Grapheme based ASR

As described in Section 3.1.1, the development of language resources (including collecting speech data and making pronunciation dictionaries) is a very demanding task. So, it is very hard to find pronunciation dictionaries for numerous languages. Though such resources are available for rich resource languages, hardly any open resource could be found for some of the languages of interest in these experiments. Moreover, the lexicon should cover at least all the words in train and test sets which is quite challenging while using off-the-shelf resources. The alternate approach, to overcome this issue, is using Grapheme-to-Phoneme (G2P) techniques (Novak et al., 2012) which is a learned statistical model to automatically transcribe words to their phonemes. Such models are trained using hand-crafted large enough lexicons, which can not be done for a lot of languages due to the lack of availability of rich pronunciation dictionaries. Furthermore, ASR performance becomes highly dependent on G2P performance. Due to these issues, initial experiments are carried out using grapheme-based lexicons. Such lexicons are automatically created by mapping words to their constituent characters.

All the languages in the MLS data set share Latin alphabets. So, all of them share a subset of characters which may help languages in multilingual scenarios, learning the shared representations. The sharing of symbols of all targeted languages is shown in Figure 4.2. The heatmap shows the proportion of characters in training data of languages on the horizontal axis that are shared with phonemes of languages on the vertical axis.

Acoustic modelling is done by training GMM-HMM and hybrid DNN-HMM models as described in Section 4.3.2. Results for all the models are reported here using the nomenclature tabulated in Table 4.1. Language model, as described in Section 4.3.3, is a 3-gram model which is trained using text corpora of the train set. For monolingual ASR of any language, text corpus for LM training is the transcriptions of the train set of that language. In the case of multilingual LM, a text corpus of unified transcriptions

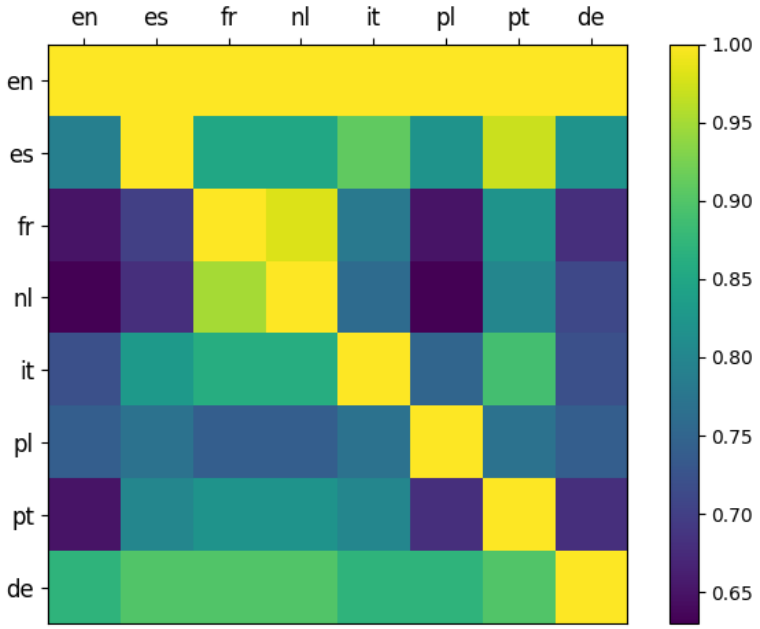


FIGURE 4.2: Graphemes sharing across the languages of MLS data sets. Each box of the heatmap represents the portion of graphemes of the languages on the vertical axis shared by the languages on the horizontal axis.

from train sets of all the languages is used for language modelling. Performance of ASR is primarily measured in terms of %WER, however, %CER is also calculated for reported results because the error analysis depicted some character level confusions due to language interference and lexicon overlap across the languages. For example, the German character 'ß' is frequently decoded as *'ss'* by the multilingual ASR system. Results of hybrid DNN-HMM techniques, *nnet2* and *lf-mmi*, are shown in Table 4.2 and Table 4.3 respectively.

Results for monolingual (*mono*), multilingual (*multi*) and multilingual AM with monolingual LM (*mono-lm*) are tabulated in the first three columns. The last column shows

TABLE 4.2: ASR systems performance on test set using grapheme-based *nnet2* trained DNN-HMM model. *mono:* Monolingual AM and LM models, *multi:* Multilingual AM and LM models, *mono-lm:* Multilingual AM and monolingual LM, *Rel. Imp.:* Relative improvement from the *mono-lm* comapred with *mono*

Language	*mono*		*multi*		*mono-lm*		Rel. Imp.	
	%WER	%CER	%WER	%CER	%WER	%CER	%WER	%CER
en	**53.45**	**31.52**	60.15	35.02	57.68	33.38	-7.91	-5.90
de	**47.42**	18.62	53.25	20.77	48.79	**18.29**	-2.89	1.77
it	43.05	13.12	44.57	12.75	**41.03**	**11.69**	4.69	10.90
nl	**50.64**	**21.55**	58.12	26.04	52.70	21.90	-4.07	-1.62
pl	53.80	16.24	57.36	19.61	**47.12**	**14.65**	12.42	9.79
pt	55.60	26.53	62.17	26.83	**54.21**	**22.33**	2.5	15.83
es	**33.00**	10.75	38.76	13.52	33.43	**10.74**	-1.3	0.09
fr	**46.29**	**23.28**	52.91	26.38	50.42	24.78	-8.92	-6.44
Overall	**47.97**	20.36	53.60	22.86	48.45	**19.93**	-1.00	2.11

TABLE 4.3: ASR systems performance on the test set using grapheme-based *lfmmi* trained DNN-HMM model. *mono:* Monolingual AM and LM models, *multi:* Multilingual AM and LM models, *mono-lm:* Multilingual AM and monolingual LM, *Rel. Imp.:* Relative improvement from the *mono-lm* comapred with *mono*

Language	*mono*		*multi*		*mono-lm*		Rel. Imp.	
	%WER	%CER	%WER	%CER	%WER	%CER	%WER	%CER
en	47.47	28.35	50.34	29.68	**47.14**	**28.15**	0.69	0.70
de	40.04	15.75	42.47	16.77	**38.76**	**14.52**	3.20	7.81
it	37.56	11.41	40.05	12.62	**36.38**	**11.15**	3.14	2.28
nl	**43.68**	**18.72**	54.40	25.39	45.89	20.18	-5.06	-7.80
pl	**47.08**	**15.82**	60.63	28.74	49.4	18.63	-4.93	-17.76
pt	49.43	22.29	58.96	27.49	**48.55**	**21.00**	1.78	5.79
es	30.28	10.57	35.89	13.37	**29.6**	**10.16**	2.24	3.88
fr	**39.69**	19.59	41.62	20.28	39.7	**19.46**	-0.02	0.66
Overall	**41.95**	**17.95**	48.07	22.05	42.14	18.1	-0.14	-0.83

the best performance improvement relative to the monolingual ASR systems. Negative relative improvement means a degradation in performance. Overall performance across the languages is also shown in the last row of the tables. It can be inferred from the results that *lf-mmi* performs better than *nnet2* DNNs when both the systems are trained using alignments from the same HMM-GMM model. Speech recognition performance improves steadily from *monophone* training towards hybrid DNN-HMM systems. This pattern is similar for *mono*, *multi* and *multi-am* models. The gap in performance of *mono* versus *multi* systems is significant in initial training setups but keeps on decreasing towards hybrid DNN-HMM systems. The performance lines, in Figure 4.3, for *mono* and *multi-am* setups of *lf-mmi* training, are very much similar. Though *multi-am* setup is comparable to *mono* but couldn't beat due to a marginal relative degradation of -0.14%. *nnet2* training showed more relative improvement in %CER than *lf-mmi*, but the overall %CER and %WER for *lf-mmi* are lower than that of *nnet2*.

4.4.2 Phonemes based ASR

As described in the Section 1.4, there is a significant research gap in studying the criteria of mixing languages for training multilingual ASR systems. In this context, very limited work has been done to study the cross-lingual acoustic-phonetic similarities. Identically represented phonemes across the languages may have very different phonetic realisations. Since the pronunciation dictionaries are not easily available for all languages or are not reliable without verification by language experts. Based on the availability of reliable pronunciation dictionaries, the scope of this ASR is limited to a few rich resource languages including English, German and Dutch. All these languages share some portion of their phonemes. The sharing of phonemes is shown in Figure 4.4 which is based on statistics extracted using alignments of train data. The map shows the percent phonemes in training data of languages on the horizontal axis that are shared with phonemes of languages on the vertical axis. Thus the map is asymmetric.

TABLE 4.4: Pronunciation dictionaries statistics for phoneme-based ASR systems training. Dictionaries are sourced from reliable open resources. Out-of-vocabulary words of train and test sets are transcribed by training a G2P model

Lang.	Source	No. of entries	No. of unique words			Avg. pronun./word
			Total	Manual	G2P	
en	CMU dict	28143	25745	19188	6557	1.09
de	bomp	30584	30569	12833	17737	1.0005
nl	fonilex	39407	24924	16948	7990	1.58

Though the test data is the same as described in Section 4.3.1 for the experiments, an additional 15 hours are randomly sampled and added to the train set of each language. Open resource lexicons for target languages (as given in Table 4.4) are used. But still, the languages do not cover all the words of train and test sets. To overcome this issue, *Phonetisarus* (Novak et al., 2012) is used to train a G2P model which is then used for annotating the words not covered by the original dictionary.

For training of acoustic and language models, the same techniques and pathways are chosen as for grapheme-based ASRs (described in Section 4.3.2). As the results of

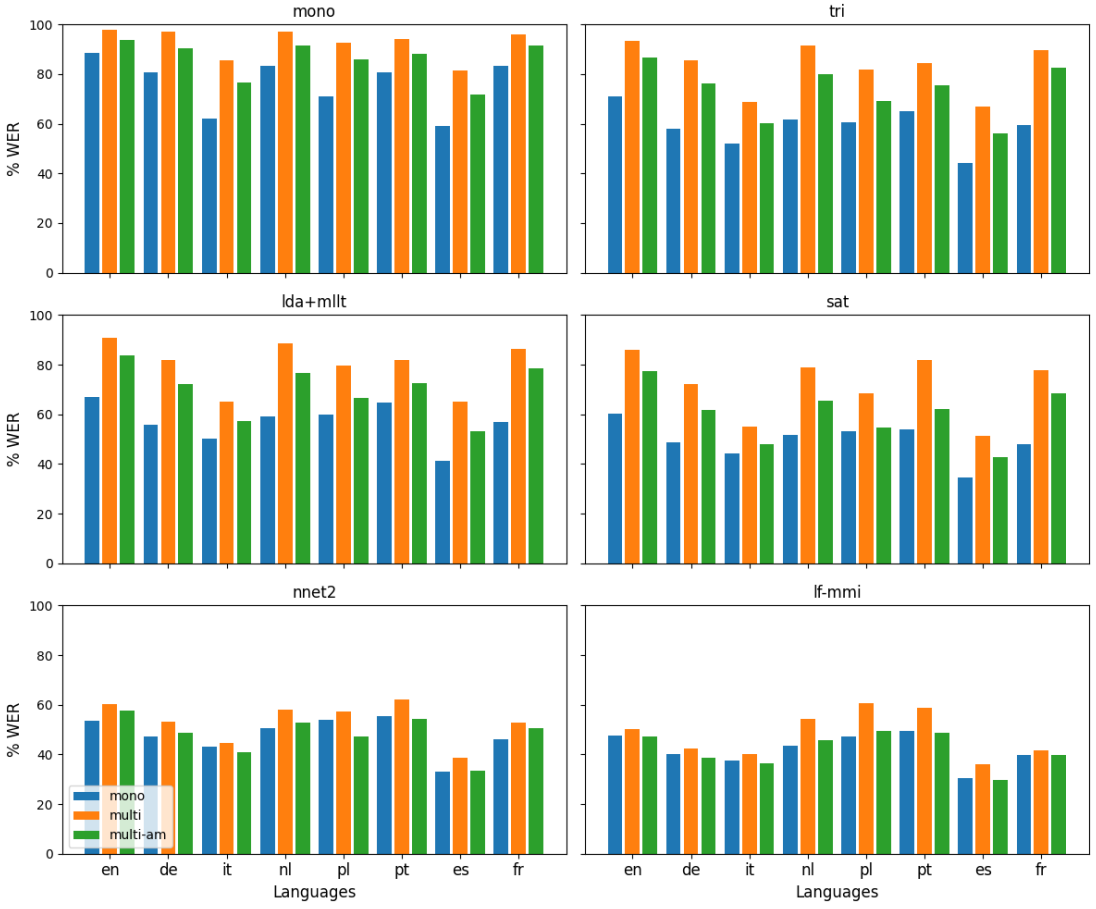


FIGURE 4.3: Performance of grapheme-based HMM models on various GMM-HMM and hybrid DNN-HMM training stages and techniques. Brief descriptions of these models are given in Table 4.1

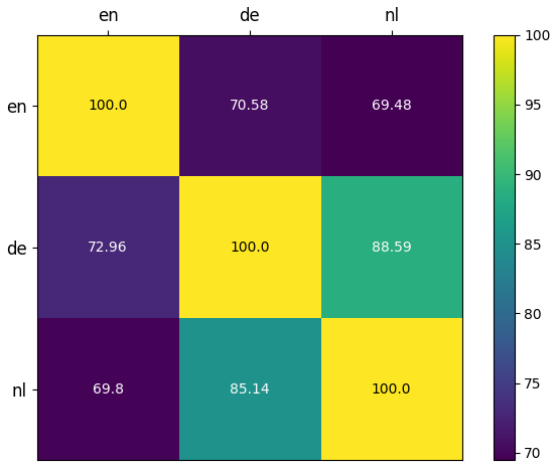


FIGURE 4.4: Percentage of IPA-representations based phonemes sharing across English (*en*), German (*de*) and Dutch (*nl*) languages. Each box of the heatmap represents the percentage of phonemes the languages on the vertical axis shared with those on the horizontal axis.

grapheme based ASR system show that *lf-mmi* outperformed *nnet2*, so only *lfmmi* hybrid DNN-HMM model is trained and the results (of *lf-mmi*) are shown in Table 4.5. The performance of speech recognition systems is shown in %WER and %PER. Similar to the grapheme-based ASR systems, the performance of *multi* goes worse than *mono* ASR systems. Using language-dependent LM with the multilingual acoustic model (*mono-lm*) helps reduce the word error rate. Since only the language model is replaced, %PER of *multi* and *mono-lm* remains the same. The phoneme-based ASR experiments are used for further analysis of cross-lingual similarities (in Chapter 6).

4.5 Discussion

4.5.1 Trends in the performance of multilingual ASR

It is evident from the results in the Table 4.5 that the performance of the multilingual ASR system degrades for all the participating languages when compared with their monolingual counterparts. The same amount of training data is used for all the languages to avoid any bias but still %PER of all the languages is increased in multilingual setup. This implies that the degradation in the performance of multilingual ASR systems for some languages can not be just attributed to the skewed data sampling.

TABLE 4.5: Phoneme-based *lfmmi* speech recognition system performance in terms of %WER/%PER. *mono:* Monolingual AM and LM models, *multi:* Multilingual AM and LM models, *mono-lm:* Multilingual AM and monolingual LM. Since only the LM is changed for *mono-lm*, the PER remains unchanged when compared with *multi*.

Language	*mono*	*multi*	*mono-lm*
en	43.84/28.10	47.48/31.06	46.40/31.06
de	37.77/26.86	40.81/28.35	38.11/28.35
nl	37.94/21.40	58.84/36.16	52.33/36.16

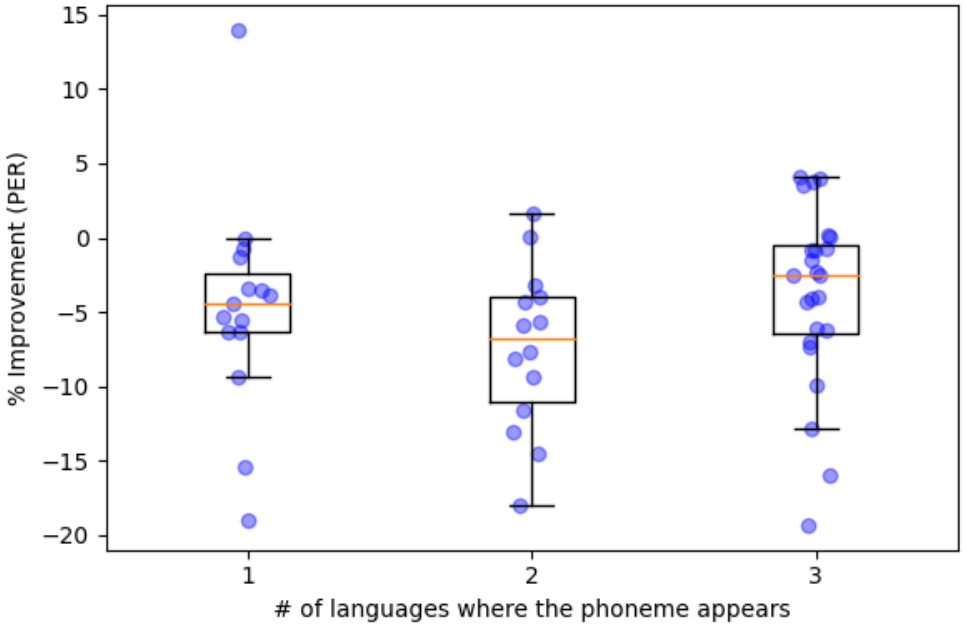


FIGURE 4.5: % Relative improvement in PER per shared phoneme of a multilingual ASR system (trained using *en*, *de* and *nl* speech data) compared with the monolingual ASR of *en* language. Each blue dot on the plot represents a phoneme and the horizontal axis shows the number of languages a phoneme is being shared with. The improvement in % is shown on the horizontal axis where the negative sign shows the degradation in the performance of the multilingual ASR system.

Even though there is a significant phonemes overlap among these languages, an increased amount of training data for those phonemes does not help to reduce their error. Similar to (Żelasko et al., 2020), the improvement in the phoneme error rate of each phoneme from multilingual ASR systems (compared with monolingual ASR systems) is calculated and plotted in Figure 4.5. Improvements with negative values show the degradation in the performance of the multilingual ASR system. As the analysis shows that the performance of shared phonemes is also degraded in multilingual setup, it intrigues us to investigate the reasons for that. The next section (Section 4.5.2) digs into the question of how a monolingual ASR perceives an input signal of some other language.

4.5.2 Cross-Lingual ASR performance analysis

The evidence from the aforementioned comparison and previous work that the multilingual systems suffer performance degradation due to language interference, intrigues us to go deeper into these confusions. To understand how a monolingual acoustic model perceives data from some other language, the speech data of a (target) language is decoded using acoustic models trained on all other (source) languages. For each source-target language pair, some of the target language phonemes might be shared by the source languages. An analysis is carried out on output from source languages' acoustic models given a target language utterance and its transcription. Given an utterance u_{L_A} of a target language L_A, the phonemic output generated from source languages' acoustic models

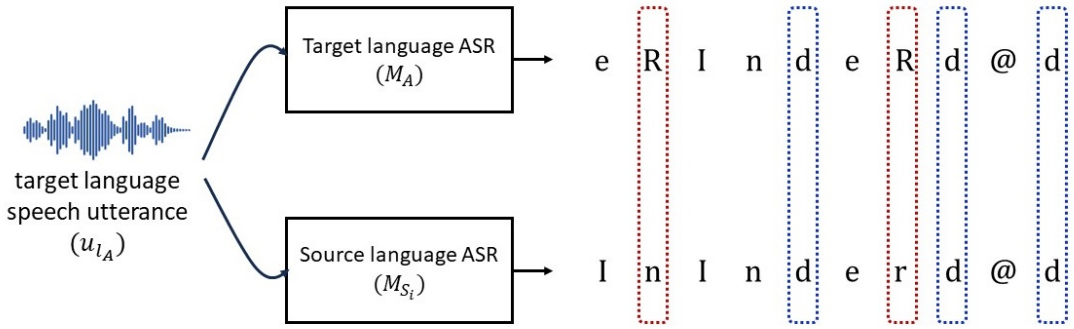


FIGURE 4.6: An illustrative example in recognition from source and target languages' acoustic models given a speech utterance u_{L_A} of the target language L_A.

are compared with ground-truth phonemic transcription. In Figure 4.6, an example of target language English decoded by its own monolingual ASR and German source language acoustic model is shown. Though it is only for illustrative purposes, some patterns can be seen in the figure. Ground-truth phonemic transcription is generated by forced alignments. The objective of this experiment is to analyse the disparity in the performance of the in-domain (target language) and out-of-domain (source language) acoustic models for a given speech signal. When a source language acoustic model produces errors given an input speech utterance of a target language, one possibility is that there might be a consistency in those errors. So, an entropy-based *Mapping Confidence (MC)* score is introduced as a metric to measure the uncertainty in the recognition results of the source language acoustic model. MC score is a measure of patterns in the output of a source language ASR for a certain phoneme. It measures how consistently a source language (L_S) acoustic model is decoding i^{th} phoneme (φ_i^A) of the target language (L_A) as the j^{th} phoneme of the source language (φ_j^S).

$$MC_{\varphi_i^A} = \frac{\sum_C p_j \log(p_j)}{\log(C)} \tag{4.2}$$

where

$$p_j = \frac{\#\text{ of } \varphi_i^A \text{ recognised as } \varphi_j^S}{\#\text{ of total occurrences of } \varphi_i^A}$$

and C is the total number of phonemes in the source language. Though entropy indicates the uncertainty in distribution, its value alone does not inform its statistical significance in this case. So, the denominator in Equation 4.2 normalises the entropy calculated in the numerator.

When the utterance u_{L_A} is decoded through a source language ASR, the output is the phonemes of the source language. Output from the source language acoustic model and the ground-truth phonemic transcriptions are then aligned together. For each source language phoneme, the MC score is calculated. An analysis is done for both shared and unshared phonemes.

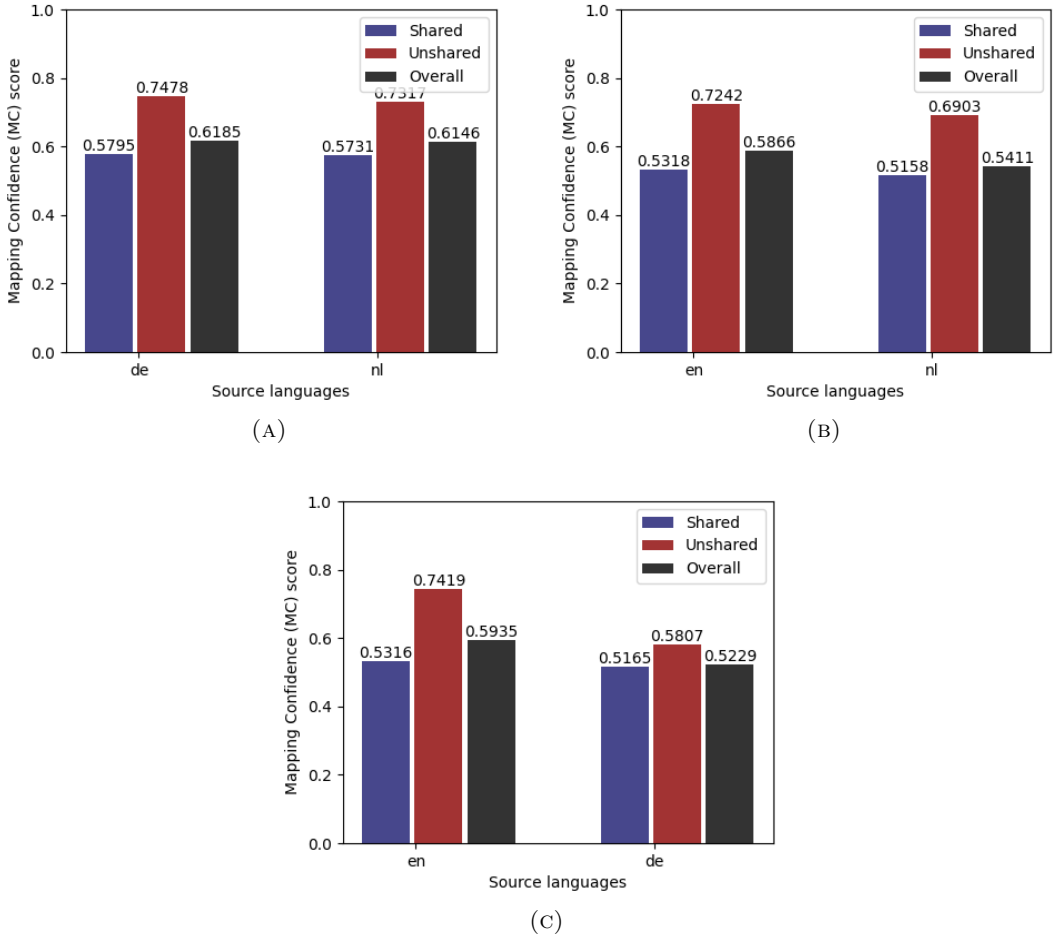


FIGURE 4.7: Mapping Confidence (MC) scores for shared and unshared phonemes of *en* (4.7a), *de* (4.7b) and *nl* (4.7c) target languages from all source languages. MC is an entropy-based score and thus lower values show higher consistency patterns in the output from the source language ASR system.

For the experiments done in Section 4.4.2, MC scores are calculated for each source-target language pair for both shared and unshared phonemes. The scores are averaged over all the phonemes in shared and unshared phoneme sets and plotted in Figure 4.7. Though the MC scores are higher for unshared phonemes than for shared phonemes, values of MC scores are around 0.54 for shared and 0.70 for unshared phonemes. Ideally, the MC score can approach 0 for i^{th} target language phoneme if a source language acoustic model always recognises it as a j^{th} source language phoneme. Since MC is an entropy-based score, even the 0.70 value represents that there are some patterns in recognition by source language acoustic models of unshared phonemes as well.

To understand what the values of the MC score mean, consider them as an entropy measure of a binary class posterior distribution (shown in Figure 4.8). The value of entropy is minimum when the probability is 0 or 1 and the value goes to highest when the probability for both classes is exactly equal. Defining this relationship in terms of

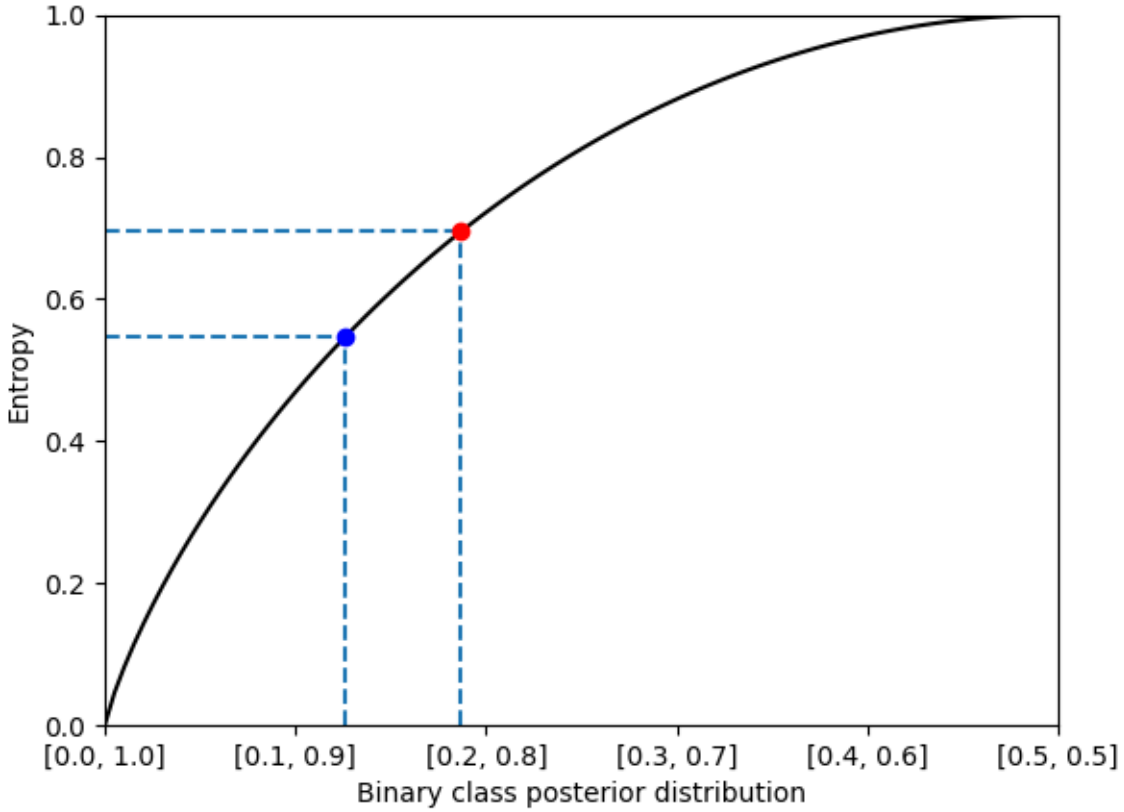
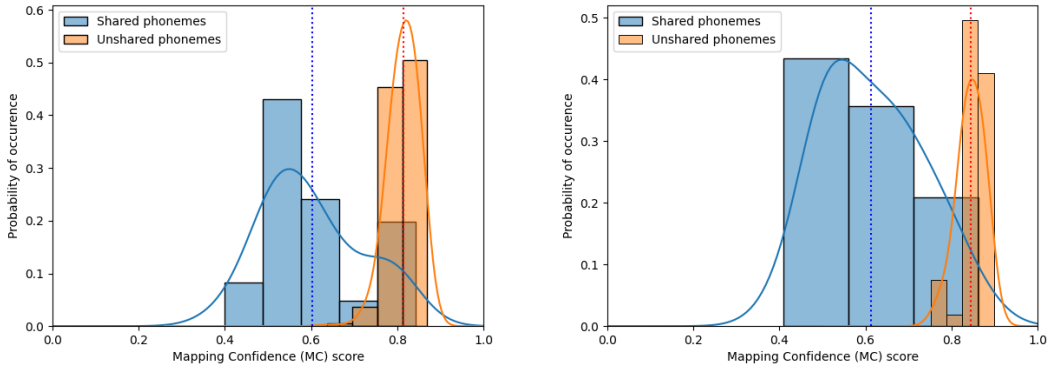


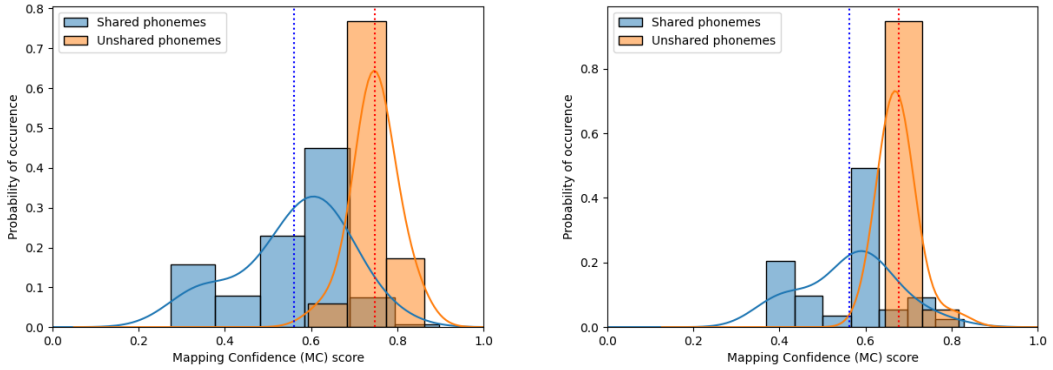
FIGURE 4.8: Probability versus entropy plot in the case of binary classification. Blue and red dots on the curve show the average MC score of shared and unshared phonemes respectively $(0.54, 0.70)$ to roughly illustrate the patterns in the outputs of different acoustic models given a speech signal

the problem here, the entropy (or MC score) is minimum when there is an exact one-to-one mapping in recognised phonemes from the target language and a source language acoustic models. MC score values go to highest when a source language acoustic model confuses a phoneme uniformly with all of its phonemes. In Figure 4.8, values of average MC scores for shared and unshared phonemes over all the languages are plotted which gives probability values of 0.13 and 0.19 for shared and unshared phonemes. It roughly means that a source language acoustic model recognises target language phonemes with at least a consistency of 80%. It implies that the outputs from the source language acoustic model have strong consistency patterns and do not produce entirely random outputs even for unshared and unseen phonemes.

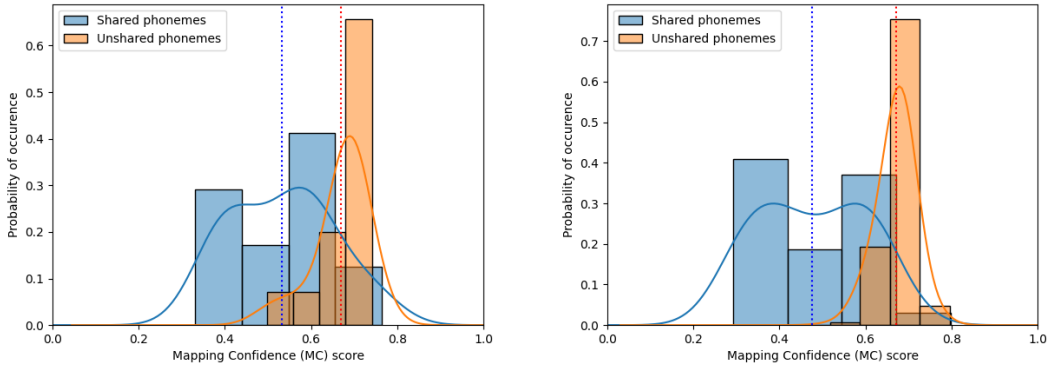
Though the average MC scores for shared and unshared phonemes are around 0.54 and 0.70, some phonemes have more consistency in mappings (lower MC score) than others (higher MC score). The overall distributions of MC scores for each source-target language pair are analysed by plotting a histogram of MC scores and their approximated distribution. Histograms for all the languages are shown in Figure 4.9. For each target



(A) *de_en* (B) *nl_en*



(C) *en_de* (D) *nl_de*



(E) *en_nl* (F) *de_nl*

FIGURE 4.9: Histogram and distribution plot of phonemes MC scores for all source-target pairs (captions in the format of *src_tgt*). The histogram and distribution in blue are for phonemes which are shared by source and target languages while orange represents unshared phonemes. Dotted lines are the mean MC values corresponding to each coloured phoneme set.

TABLE 4.6: Comparison of source-target language similarity in terms of shared phonemes and average MC scores

Target Language	Source Languages (phoneme shares / MC score)		
	en	de	nl
en		0.70 / 0.62	0.69 / **0.61**
de	0.73 / 0.59		0.89 / **0.54**
nl	0.70 / 0.59	0.85 / **0.52**	

language, MC scores histograms are plotted for all the source languages and the bin width is set for both, shared and unshared phonemes, according to the Freedman-Diaconis rule (Freedman and Diaconis, 1981). It has been designed to roughly minimise the L2 distance between the relative frequency density (histogram) of a distribution and the density of the theoretical probability distribution. Freedman-Diaconis rule sets bin width as;

$$BW = \frac{IQR(data)}{\sqrt[3]{n}}$$

where IQR is the interquartile range which is a difference between the 75th and 25th percentile of data.

For each source-target pair, plots for shared and unshared phonemes are independent of each other which means that the probability of occurrence on the vertical axis sums to 1 for each set of phonemes i.e. shared phonemes and unshared phonemes. It can be seen that for most of the plots (especially for the shared phonemes), the distribution peak occurs before the mean MC score (dotted line). It indicates that the majority of the phonemes of a target language are recognised as some consistent source language phonemes by the source language ASR system. It could be a one-to-one, one-to-many or many-to-one mapping. Rather than carrying out a human-based analysis of these mappings, it is proposed to train a *mapping model* to learn these mappings.

4.5.3 Comparison with phoneme sharing

In the Table 4.6, mean MC scores are compared with the phoneme sharing of languages shown in Figure 4.4. Rather than showing phoneme shares in percentage, they are scaled from 0 to 1 to compare with mean MC scores. The maximum value of phoneme shares can reach to one and the lower and the upper limits (or best and the worst values) for MC score are 0 and 1. It is evident from the Table 4.6 that the higher phoneme sharing does not guarantee a lower MC score for a source-target language pair. For example for *en* target language, *de* language shares more phonemes with it when compared with *nl* but *nl* has a lower MC score than *de*. For *de* and *nl* target languages, MC scores are lower for the languages that have higher phoneme sharing, however, the magnitude of change in MC score is not as high as in phoneme shares. It shows that the consistencies in decoded results from source languages ASR do not depend on the magnitude of the shared phonemes set.

Given the observations that there exists patterns or consistencies when the data of a target language is decoded through a source language ASR, it is proposed to train a neural network model to learn these mappings. In the following chapters, details of mapping models, the use of these models to analyse cross-lingual acoustic phonetic similarities and their application to improve low-resource speech recognition are discussed.

4.6 Summary

In this chapter, baseline speech recognition systems have been trained for three West Germanic languages i.e. English, German and Dutch from the MLS dataset. Phoneme-based hybrid DNN-HMM acoustic models have been trained on uniformly sampled 30 hours from the train set of the MLS data set. A multilingual model has also been trained by mixing the training data of all the languages. The performance of acoustic models has been estimated on 2-hour test sets uniformly sampled from the MLS test set. Results have shown that the performance of multilingual ASR systems degrades for all the languages compared with their monolingual counterparts. The comparison of the error rate of each phoneme from monolingual versus multilingual acoustic models has shown that a phoneme being shared among more languages does not guarantee an improvement in its error rate from the multilingual acoustic model.

This motivated us to analyse the behaviour of acoustic models of different (source) languages given a target language speech utterance. For this analysis, the mapping confidence (MC) score has been calculated for each target language phoneme. MC score is an entropy-based measure to estimate the patterns in the output from a source acoustic model and the ground-truth phonemic transcription. The results have shown that there are consistent patterns for all the languages when decoded from source languages' acoustic models with an average MC score of 0.54 and 0.70 for shared and unshared phonemes. A probability versus entropy plot has been presented as an aid in understanding the significance of the MC score which depicts that significant patterns exist in ground-truth and source language's acoustic model phonemic outputs. It is proposed that such patterns can be learnt by training a machine learning model.

Chapter 5

Mapping models

Mapping models lie at the heart of the rest of the work described in this thesis. In this chapter, training of baseline mapping models and gradual improvements in their performance are discussed in detail. As described in Section 4.5, mapping models are trained to learn the mappings between posterior distributions from a source language and a target language ASR given a target language utterance. The concept of the mapping model is somewhat analogous to KL-HMMs (Aradilla et al., 2007). In both models, the DNN output in a hybrid DNN-HMM framework produces posterior distributions of phonemes. However, while the mapping model is trained to map posterior distributions from acoustic models of different languages, KL-HMMs aim to map phoneme posteriors from the DNN to the HMM states' emission probabilities for graphemes. Although the motivation and parameter estimation processes differ between KL-HMMs and mapping models, KL-HMMs can be viewed as a type of mapping model. The mapping models discussed here specifically map posterior distributions of phonemes from the acoustic models of different languages, both source and target. The posterior distributions can be output distributions from a phoneme, grapheme or sub-word units based ASR systems. It can be from end-to-end or hybrid DNN-HMM models. However, training methods of mapping models are not affected by any of these constraints. Though a mapping model can be trained on top of any of the said systems, phoneme-based hybrid DNN-HMM speech recognition systems are preferred over e2e systems for the initial experiments here due to several following reasons.

- As a starting point, phoneme-based systems are used because the motivation of the initial work is to compare cross-lingual acoustic-phonetic similarities (Chapter 6). Additionally, the limited size of the universal phoneme set allows a significant cross-lingual phoneme overlap as compared to graphemes which makes it less challenging for models to learn cross-lingual mappings.
- Hybrid DNN-HMM models are preferred over end-to-end ASR systems to avoid the effect of implicitly entangled language models in E2E systems which might manipulate the analysis of cross-lingual acoustic-phonetic similarities.

- One of the motivations of this work is to leverage these models to improve speech recognition for low-resource languages. Though E2E modelling has dominated recently, hybrid DNN-HMM outperforms them in limited data scenarios (Kürzinger et al., 2020). The only downside of hybrid systems is using manually created lexicons. However, the advancement of G2P and text to IPA transliteration approaches such as Phonetisarus (Novak et al., 2012), Epitran (Mortensen et al., 2018), and open source LanguageNet G2P models (Hasegawa-Johnson et al., 2020) have alleviated this problem.

The structure of this chapter is as follows. Some necessary terms are defined in Section 5.1 for better readability. Then, baseline mapping models are explained in Section 5.2. Structural and optimisation improvements in baseline mapping models are described in Section 5.3 and Section 5.4. Section 5.5 discusses the training and evaluation setups for the experimentation. Results are presented and discussed in detail in Section 5.6.

5.1 Terminology

Before delving further into the details of mapping models, a few terms are defined here to avoid confusion. Mapping models are trained to map posteriors from space to the other. Some of the terms used for different posterior distributions are defined here and will be used in the rest of the thesis.

- **Target language:** The language of an input speech utterance.
- **Source language:** A language from the participating languages that is different from the target language.
- **Source posterior distribution:** Posterior distribution from a source language ASR given a target language speech utterance. Source posterior distributions are also shortened as **source posteriors**. The posteriors from i^{th} source language are denoted by P_{S_i}.
- **Target posterior distribution:** Posterior distribution from the target language ASR given a target language speech utterance. It can also be a one-hot vector from forced alignments. Target posterior distributions are also shortened as **target posteriors**. These are denoted as P_A.
- **Mapping model:** A machine learning model which transforms source posterior distributions to the target posterior distributions' space. A mapping model is specified as a source-target mapping model for specific source and target languages.
- **Mapped posterior distribution:** Output posterior distributions from a source-target language model given a target language speech utterance. Mapped posterior distributions are also shortened as **mapped posteriors**. For target language L_A, mapped posteriors from source language S_i are denoted as P_{S_iA}.

5.2 Baseline mapping models

Let M_A and M_{S_i} be the monolingual acoustic models of the target and i^{th} source language respectively, a mapping model $\mathcal{M}_{S_iA}$ is trained to transform posteriors $P_{S_i} \in \mathbb{R}^{d_{S_i}}$ from M_{S_i} to the posteriors $P_{S_iA} \in \mathbb{R}^{d_A}$ where d_{S_i} and d_A are the number of output classes in a posterior distribution from M_{S_i} and M_A respectively.

Mapping models are trained on frame level and each frame of a speech signal is assumed independent. KL divergence objective function is used for training. Let $X = \{x_1, x_2, \ldots, x_T\}$ be a set of observations in a batch of the target language, for which posterior distributions ($P^Z = \{p_1^Z, p_2^Z, \ldots, p_T^Z\}$ where $Z \in \{A, S_i\}$) are attained from all monolingual acoustic models. A mapping model is trained using KL divergence loss to map posteriors from source acoustic models (P^{S_i}) to the target language posteriors (P^{S_iA}). The loss function for a batch is given as;

$$\mathcal{L}_{S_iA}(\theta) = \sum_{t=1}^{T} \sum_{k=1}^{d_A} p_{t,k}^A \cdot (\log p_{t,k}^A - \log p_{t,k}^{S_iA}) \tag{5.1}$$

where $p_{t,k}^A$ and $p_{t,k}^{S_iA}$ are the target and hypothesis posteriors for k^{th} class of t^{th} input frame in a batch.

Target posteriors for training can be extracted in two ways either by decoding training data from the target language acoustic model or by force alignments of the training data. For forced alignment, an audio file is processed through a pre-trained acoustic model, typically a hybrid DNN-HMM model, using Viterbi decoding. During this process, the model evaluates the likelihood of each HMM state generating the observed acoustic features at every time frame in the audio. The outcome of forced alignment is a time-aligned transcription where each phoneme in the transcription accurately corresponds to its respective segment of the audio waveform.

The acoustic model gives a posterior distribution with probabilities for all the output classes for each frame in case of decoding. However in force alignments, usually the alignments are produced as one hot vector i.e. probability is 1 for the aligned class and 0 for all others. Though a mapping model can be trained with target posteriors extracted through any of these ways, KL loss reduces to Cross-entropy loss in forced alignment case. With target posterior distributions of one-hot vector the term $p_{t,k}^A \cdot \log p_{t,k}^A$ will always be zero and the Equation 5.1 is reduced to

$$\mathcal{L}_{S_iA}(\theta) = -\sum_{t=1}^{T} \sum_{k=1}^{d_A} p_{t,k}^A \cdot \log p_{t,k}^{S_iA} \tag{5.2}$$

which is the cross-entropy loss. Mapping models are trained using either of these losses in the following chapters.

A separate mapping model is trained for each source-target language pair which implies

that a total of $N(N-1)$ models are required to be trained for N participating languages. Baseline mapping models consist of three stacked Fully-Connected (FC) layers.

The proposed baseline mapping model learns mappings on the frame level without considering the contextual information but the connected speech is a continuous signal which poses co-articulation and temporal smearing. Thus assuming each frame independent of its context is a very weak modelling approach. Furthermore, pairwise models are trained for all languages which restricts applying baseline mapping models on a large scale as $N(N-1)$ models are needed for N languages. So, in the next few sections, various improvements are explored to address the aforementioned shortcomings of the baseline mapping models.

5.3 Structural improvements

5.3.1 Contextual information

Continuous speech signal has temporal smearing, co-articulation and several other characteristics which implies that the signal at a given time step is not independent of its context. This suggests that the output posterior distributions from an ASR also depend on their contextual frames. Since the baseline mapping models in Section 5.2 are trained on frame-level posterior distribution with an independence assumption, the impact of considering context is studied here.

To include context for mapping model training, various modifications are experimented with.

- **Contextual input:** Mapping model architecture is kept the same as for the baseline mapping models, but input to the model is the posterior distribution of the current frame concatenated with posterior distributions of a few left and right frames. Different context windows are explored to study the effect of contextual information on the performance of mapping models. Using the baseline architecture, posterior distributions of a few left and right frames are concatenated to the input along one dimension.

- **Auto-regressive mapping model:** Auto-regressive mapping model is also explored where output at previous time frames is also fed along with right and left input context. Input and previous outputs are processed through separate layers. The output of final layers from separate modules is then combined and fed to the shared output layers.

- **Convolution layers:** Convolution layers are added before the fully connected linear layers to capture the context. Contextual frames are concatenated along the second dimension before inputting to the mapping model where initial CNN layers are expected to capture the dependencies. The output from CNN layers is flattened and fed to a few fully-connected layers.

- **Sequence modelling:** Sequence model architecture is also used for mapping models rather than training non-sequential neural networks. A sequence RNN model is expected to better capture the sequential information.

The implementation details are discussed elaborately in Section 5.5.4.

5.3.2 Sequence modelling

In order to capture the sequence information better, an encoder-decoder model is experimented with rather than the non-sequential mapping models. Given a speech utterance u, the output posterior distribution from i^{th} source language acoustic model M_{S_i} is fed to an encoder module. For output at a given time frame t, the encoder state at that time step is fed to a decoder along with the decoder's output at the previous time step $(t-1)$. The impact of adding an attention module before the decoder is also explored. For the attention decoder, encoder states are first fed to an attention module to compute the context vector which is then passed to the decoder module along with the prediction at the previous time step.

5.3.3 Multi-encoder single-decoder model

To deal with the requirement of $N(N-1)$ number of models of baseline mapping models, Multi Encoder Single Decoder (MESD) is proposed to train only one model per target language. An encoder-decoder model with multiple source language-dependent encoders and a single target language decoder is presented here. The architecture of the proposed model is shown in Figure 5.1.

Given a speech utterance u, source posterior distributions (P_{S_i}) are fed to the corresponding encoder (E_{S_i}). Embeddings from the final encoder layer are then fed to a single decoder through a language switch. The switch connects the encoder of a specific language to the decoder depending on the input ASR. It implies that mappings can be obtained even having input from only one source-language ASR.

For a target language, all the baseline mapping models from different source languages map different inputs to the same output space. So, the underlying intuition for the MESD model is that each encoder can extract source-language independent features which are then fed to the decoder which could map its input to the target language posteriors' space.

Though a mapping model contains multiple encoders, any encoder can be used with a decoder during inference and MESD does not require a data stream from all the encoders for a given utterance. It implies that mappings can be obtained by having input even from only one source language at a time.

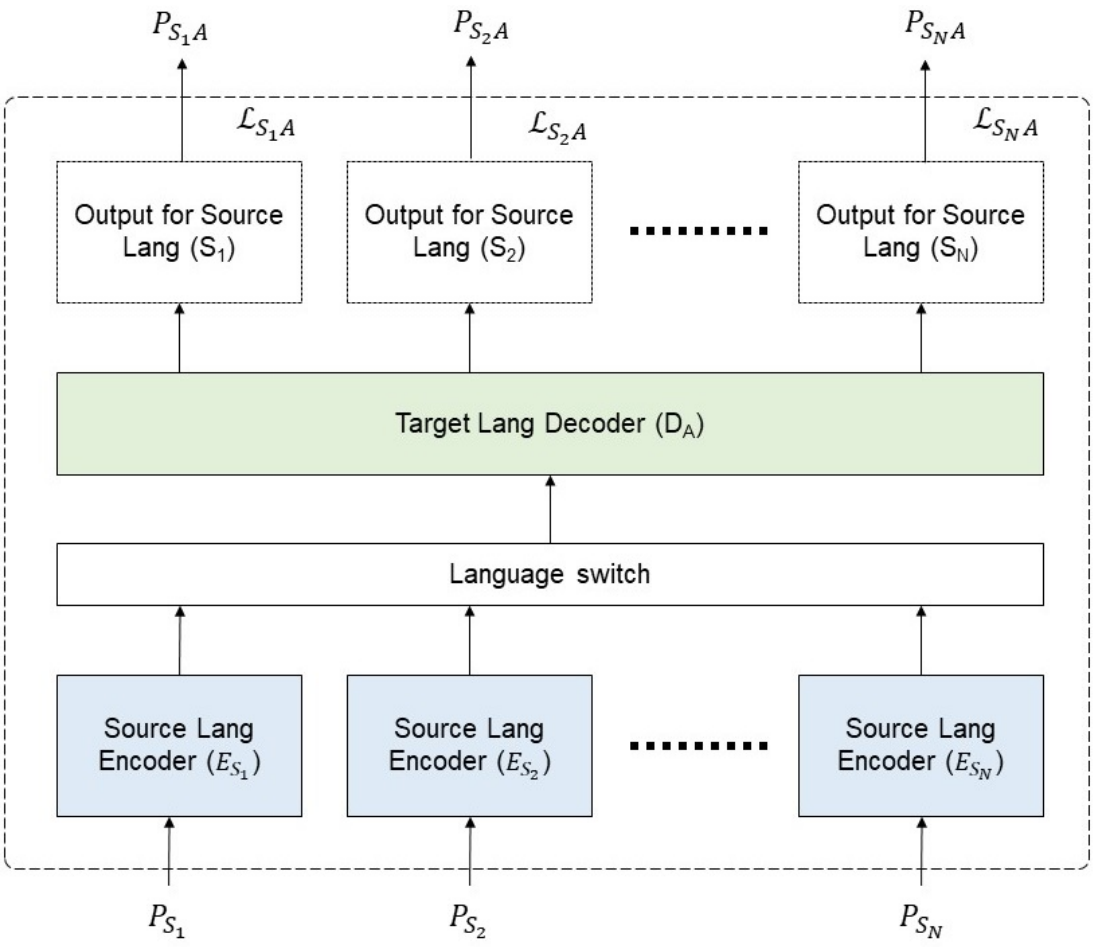


FIGURE 5.1: Architecture of the proposed multi-encoder single-decoder model. The model consists of $N-1$ source language-specific encoders and a single decoder. The input of each encoder E_{S_i} is the source posteriors from i^{th} source language. The single decoder outputs the mapped posteriors from any of the encoders.

5.4 Optimisation improvements

5.4.1 MESD training loss

For the training of the proposed MESD model, outputs from all the source AMs for a given utterance u are sequentially fed as input to respective encoders which are propagated to the single decoder in the forward pass. The loss of a batch is then calculated as the weighted sum of all the encoder-decoder pair losses.

$$\mathcal{L}_A(\theta) = \sum_K w_k \cdot \mathcal{L}_{S_kA} \tag{5.3}$$

where K is the number of total source languages ($K = N-1$), $w_k = \frac{1}{K}$ in the case of mean average and $\mathcal{L}_{S_kA}$ is given in Equation 5.1 and Equation 5.2 which is still a frame-based loss. It allows mapping model training to converge in low-resource settings as a small amount of data provides millions of examples. However, this might cause unbalanced training across languages as the mean value can be continuously decreasing

Algorithm 1 Multi-Encoder Single Decoder

Require: M_{S_i}: ASR systems of source languages
Require: α: step size hyper-parameters
randomly initialise θ_{e_i}, θ_d
while not done **do**
  Sample batch of target language utterances $\mathcal{T}$
  for source languages S_i **do**
    Obtain posterior distribution $P_{S_i}^{\mathcal{T}}$ using M_{S_i}
    Evaluate $\nabla_{\theta_{e_i},\theta_d}\mathcal{L}_{S_iA}^{\mathcal{T}}(f_{\theta_{e_i},\theta_d})$
  end for
  initialise $\mathcal{L}_A$ to zero
  for source languages S_i **do**
    Calculate weights for each source language loss w_i
    Update $\theta_{e_i} \longleftarrow \theta_{e_i} - \alpha\nabla_{\theta_{e_i}} w_i\mathcal{L}_{S_iA}^{\mathcal{T}}(f_{\theta_{e_i},\theta_d})$
    $\mathcal{L}_A \longleftarrow \mathcal{L}_A + w_i\mathcal{L}_{S_iA}^{\mathcal{T}}(f_{\theta_{e_i},\theta_d})$
  end for
  Update $\theta_d \longleftarrow \theta_d - \alpha\nabla_{\theta_d}\mathcal{L}_{\mathcal{A}}$
end while

when loss for one of the languages is decreasing monotonically but increasing in the same fashion for the other one. This can cause the model to learn mappings for one language much better than for another. To cope with this issue, a dynamic weighting scheme is applied to weight the losses for each encoder-decoder loss. For the experimentation here, rank sum weighting (Roszkowska, 2013) is used to assign the weights. In this scheme, weights are assigned based on their normalised ranks. So, w in Equation 5.3 now becomes

$$w_r = \frac{2(K+1-r)}{K(K+1)} \tag{5.4}$$

where r is the rank of the language when the languages are sorted on decreasing values of their losses. It helps to restrict the model from biasing towards a specific language or a group of languages.

The algorithm of MESD training is shown in 1.

5.4.2 LR Annealing

Training with a large initial learning rate followed by gradually decreasing learning rates proves to outperform training with either a large or small learning rate throughout the whole training process (Nakkiran, 2020). Various methods are widely used for learning rate annealing (George and Powell, 2006, Smith, 2015, Smith et al., 2017). For the experimentation here, an annealing method is used which depends on the improvement of the model's accuracy over time. At each epoch e, the learning rate η is multiplied by a factor f if $Accu_e - Accu_{e-1} < m$ where f and m are predefined factorisation and threshold values respectively. $Accu_x$ is the accuracy of the model on the dev set after epoch x.

TABLE 5.1: Details of the Babel data sets used for training speech recognition systems.

Lang	Train		Eval	
	# hours	# spks	# hours	# spks
Tamil *(tam)*	110.67	372	16.08	61
Telugu *(tel)*	67.27	243	13.92	60
Cebuano *(ceb)*	74.26	239	15.51	60
Javanese *(jav)*	76.39	242	16.25	60

$$\eta^{'} = \begin{cases} \eta f, & \text{if } Accu_e - Accu_{e-1} < m \\ \eta, & \text{otherwise} \end{cases}$$

5.5 Experimental Setup

Mapping models can be trained on top of different ASR systems and is done in later chapters, all the experiments reported in this chapter are done with the same data set and underlying ASR systems for the sake of consistency and fairness. The scope of this chapter is limited to the performance of baseline mapping models and experimentation with various structural and optimisation improvements to enhance the performance of mapping models.

5.5.1 Data sets

For the experimentation here, mapping models are trained on the top of acoustic models of four low-resource languages from IARPA Babel speech corpus (Gales et al., 2014). Full language packs of Tamil (*tam*), Telugu (*tel*), Cebuano (*ceb*) and Javanese (*jav*) are used for baseline ASR training and evaluation. Since the eval data of Babel is not publicly available, train and dev sets of Babel data sets are used as train and eval sets respectively for the experiments. The details of the data sets are tabulated in Table 5.1. These data sets consist of conversational telephone speech and are quite challenging because of limited bandwidth, conversational styles, channel and background environment conditions. A limited amount of scripted read speech is also included in each language pack.

Full amounts are used for the training of baseline monolingual speech recognition systems. However, for training the mapping models, a subset of 30 hours is chosen from

TABLE 5.2: Examples (in millions) for training of mapping models for each target language. Train set: 29 hours; Dev set: 1 hour; Eval set is same as for the ASR

Language	Train	Dev	Eval
Tamil *(tam)*	3.234	0.358	1.664
Telugu *(tel)*	3.232	0.356	1.915
Cebuano *(ceb)*	3.241	0.348	1.943
Javanese *(jav)*	3.225	0.365	1.854

each language pack. Utterances containing only non-speech or silence are discarded while randomly sampling the 30 hours. This data is further divided randomly into 29 and 1-hour portions as train and dev set to train the mapping models. Since the mapping models are trained on frame level, 30 hours provide millions of examples for the sufficient training of these models. The examples, used for building the mapping models, are given in Table 5.2.

5.5.2 Speech recognition systems

As mapping models are trained using posterior distributions from acoustic models, they are briefly described here. Monolingual acoustic models of all the languages are hybrid DNN-HMM models. 40 MFCCs are extracted for each frame of the speech signals using a window size of 25 ms and a shift of 10 ms. These features are then fed to DNN which consists of 12 factorised TDNN (TDNN-F) layers (Povey et al., 2018). Each TDNN-F hidden layer is of dimension 1024, factorised with a linear *'bottleneck'* dimension of 128. The acoustic model is trained using lattice-free MMI criterion (Povey et al., 2016). Neural network outputs posteriors of the clustered monophone classes. Clustering results in different clusters for each monolingual ASR training and thus the outputs from different acoustic models against an identical speech signal are not directly comparable. The ASR systems are built using the Kaldi toolkit (Povey et al., 2011).

5.5.3 Performance metric

The performance of mapping models is reported in terms of accuracy. The accuracy of a mapping model is measured as the ratio of correctly mapped frames (CMF) to the total number of frames (TF). Correctly mapped frames are defined as the frames where the most probable class from the mapping model is among the top k most probable classes of target posterior distribution. Accuracies of mapping models in this chapter are usually shown for various values of $k \in \{1, 5, 10\}$. The detailed method to calculate the accuracy of a mapping model $\mathcal{M}_{S_iA}$ is given in Algorithm 2.

5.5.4 Structural improvements

The implementation details of the structural improvements discussed in Section 5.3.1 are discussed here.

- **Contextual input:** To consider the contextual information while training the mapping model, the experimentation is started by appending contextual frames to the input in 1-D. Consider l left and r right context, source posterior distributions from $t-l$ frames are concatenated on the left of source posteriors at current time t and distributions of $t+r$ frames are appended on the right of the input.

- **Auto-regressive mapping model:** Using a baseline mapping model, an auto-regressive setup is also explored where the output of a few previous time steps is also fed to the model. Input (with concatenated context) is fed to a separate module (*contextual input* module) of a few linear layers and concatenated output

Algorithm 2 Accuracy measure of a mapping model

Require: $P^A = \{p_1^A, p_2^A, \cdots, p_T^A\}$: Target posterior distribution
Require: $P^{S_iA} = \{p_1^{S_iA}, p_2^{S_iA}, \cdots, p_T^{S_iA}\}$: mapped posterior distribution from mapping model N_{S_iA}
Require: k: Number of most probable classes to consider from target posteriors for accuracy calculation

```
for p_t^{S_iA} in P^{S_iA} do
    cands_{t,k} ⟵ Indices of top k most probable classes in p_t^A
    o_t ⟵ Index of most probable class in p_t^{S_iA}
    if o_t in cands_{t,k} then
        CMF ⟵ CMF + 1
    end if
    TF ⟵ TF + 1
end for
accuracy ⟵ CMF / TF
return accuracy
```

of a few previous time frames is fed to another module (*contextual output* module) consisting of some linear layers. The output of both modules is combined and then fed to the final linear layers. The architecture is shown in Figure 5.2.

- **Convolution layers:** To make use of convolution to capture the context, posteriors from contextual frames are concatenated in 2-D. Posteriors from $t-l$, t, and $t+r$ are stacked and convolution layers are added before linear layers of the mapping model to capture the context. Various configurations of convolution layers are experimented with.

- **Sequence modelling:** Finally, an encoder-decoder architecture is used with an attention module in between them for sequence modelling. Both encoder and decoder with a single GRU layer and dot attention are found to outperform other variants during the experimentation.

5.5.5 Optimisation improvements

For better convergence, learning rate annealing is used for the training. For an encoder-decoder mapping model, the learning rate is factored by 0.8 if the accuracy of two adjacent epochs is less than by an absolute value of 0.25.

Along with LR annealing, rank sum weighting is employed to calculate the total loss for the MESD model. A batch of the same target utterances is fed to each source-language encoder and decoder sequentially and the overall loss is calculated using Equation 5.3. All three losses are sorted on their decreasing value and the first ranked language gets the largest weight and so on. It helps to stop the model from biasing towards a specific language.

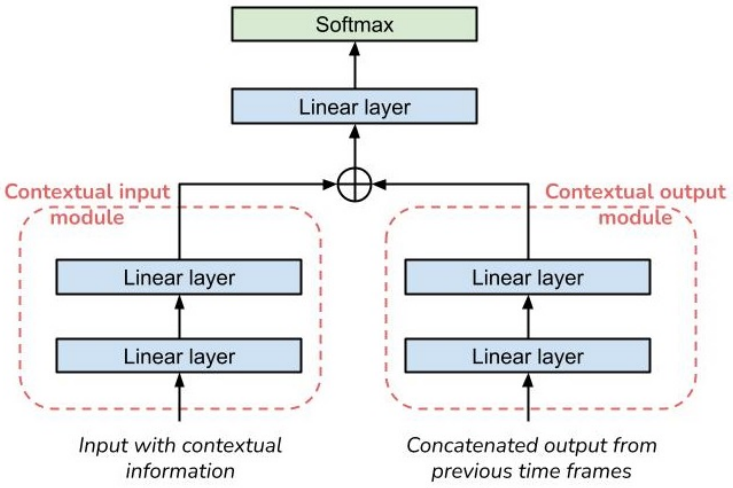


FIGURE 5.2: Architecture of baseline auto-regressive mapping model. Contextual input and output from previous time frames are given to separate fully-connected modules. The output of the final layers from both module is then combined before feeding to the shared layers.

TABLE 5.3: Hybrid DNN-HMM ASR system performance in terms of %PER. Mapping models are trained on top of *mono* ASR systems.

Language	%PER	
	mono	*multi*
Tamil (*tam*)	43.96	43.67
Telugu (*tel*)	43.66	46.36
Cebuano (*ceb*)	36.67	41.02
Javanese (*jav*)	41.60	45.54

5.6 Results and discussion

5.6.1 Speech recognition systems

The results of the baseline speech recognition systems for all the languages are shown in Table 5.3 in terms of %PER. Monolingual (*mono*) baseline systems are the language-dependent acoustic, pronunciation and language models which are trained on a language-specific data set. On the other hand, all the components of multilingual (*multi*) systems are language-independent (the train sets of all the languages are mixed before training of acoustic and language models). The mapping models are trained on top of monolingual (*mono*) acoustic models. The error rates here for the Babel data set are higher than the MLS data set (in Table 5.3). This is mainly due to the very challenging nature of the Babel data set. Since the mapping models are trained on top of these models, the performance of mapping models is also influenced by the performance of acoustic models.

5.6.2 Baseline mapping models

As described earlier a mapping model is trained for each source-language pair of Babel data set. So, 12 mapping models are trained on the top of baseline ASR systems using data sets outlined in Section 5.5.1. Since even the limited amounts of data provide millions of training examples, baseline mapping model training converges in early epochs for all the networks with an accuracy of nearly 50%. The training curves with accuracy measure for one mapping model (N_{ceb_tel}) are shown in Figure 5.3 and accuracies of mapping models are tabulated in Table 5.4.

Analysis reveals that most of the time when the most probable output class from a trained mapping model is not accurate i.e. it is not the same as the most probable target class, the correct target AM class is usually still among a few most probable classes of the mapped distribution. So, the network accuracy is recalculated considering top n classes of the *mapping model* output. The results in Table 5.4 show that the accuracy for most of the networks is dramatically increased to nearly 90% from less than 50% by considering the top two most probable classes of the mapped posteriors.

Although the baseline mapping models seem promising and details of their use for analysis and downstream ASR tasks are shown in the next chapters, there is still a lot of room to improve their accuracies. In the next sections, the results of several structural and optimisation improvements are presented. As described earlier $N(N-1)$ mapping models are needed to train for N languages in the baseline system which makes this technique difficult to use at scale. So, proposed structural and optimisation improvements are not done for all models to save time and explored for randomly selected one source-target mapping model ($\mathcal{M}_{ceb\_tel}$) and then the best architecture and configuration are used for the rest of the mapping models.

5.6.3 Structural improvements

Initial experiments are done with different input and output contexts and the results are shown in the top section of the Table 5.5. It is evident that using context wins gain

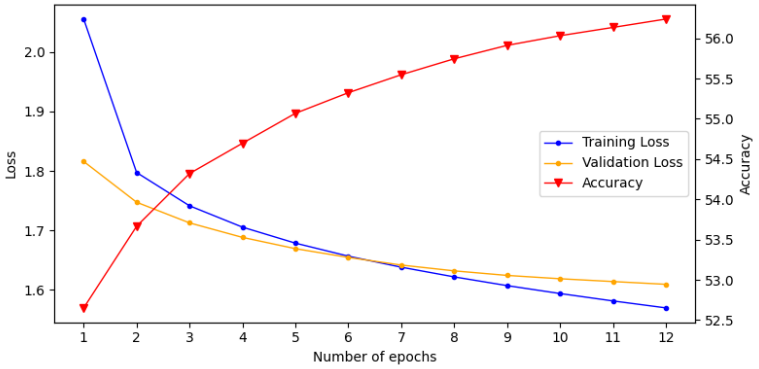


FIGURE 5.3: Training curve of mapping model for target language *tel* from source language *ceb* . The curve shows that the training converges in very early epochs.

TABLE 5.4: Accuracy of baseline mapping models considering top n mapped classes. The accuracy of a source-target mapping model is calculated using Algorithm 2.

Target Lang	Source Lang	*mapping model* accuracy			
		n=1	*n=2*	*n=5*	*n=10*
tam	*tel*	42.91	88.16	94.91	97.91
	ceb	44.43	84.63	91.82	96.13
	jav	41.89	85.82	92.82	96.69
tel	*tam*	54.44	92.08	96.87	98.58
	ceb	35.51	90.26	95.40	97.91
	jav	50.54	90.71	95.70	98.10
ceb	*tam*	45.73	85.50	93.71	97.23
	tel	46.17	87.87	93.98	97.51
	jav	47.04	88.50	94.67	98.03
jav	*tam*	47.81	85.63	93.36	96.58
	tel	48.29	86.31	93.74	97.03
	ceb	48.05	86.28	93.61	96.95

in the performance. Appending only the left 5 frames to the input gives a gain of 1.59% which increases to 2.64% when concatenating the same number of frames from the right context as well. A marginal gain of 0.5% is achieved using the output of the previous 3 frames in the auto-regressive architecture. For this experimentation, the architecture of Figure 5.2 is used. Both the contextual input and the contextual output modules consist of one fully connected layer. Two fully connected layers are used after combining the output of final layers from both contextual input and output modules. In addition, multiplication and concatenation methods are experimented with to combine the output of contextual input and output modules. Results of `E3` are reported with concatenation as it outperforms addition and multiplication. The gain achieved in this experiment (`E3`) in comparison with `E2` might be attributed to increased model capacity. So as an ablation study, the model capacity is increased by adding a few more layers for the same input of experiment `E3` (experiment `E3'`). The number of FC layers in contextual input and output modules is increased to 2 layers per module and three FC layers are stacked after concatenation of output from both modules. It drops the accuracy to 60.32% which is marginally worse than the performance of the model with smaller capacity. So, the architecture of initial experiments (`E1-E3`) is retained for the rest of the experiments. Increasing the context window from 5 to 10 frames slightly increases the gain for all aforementioned scenarios (`E4-E6`).

In further experiments, convolutional layers are applied to extract the features from contextual frames. Posterior distributions of the current time frame and several contextual frames are concatenated in 2-D. Convolution filters are applied on this 2D input to capture the context-aware features. For the experimentation here, 3 convolutional layers are applied followed by 4 FC layers. Various convolutional layer configurations experiment and results with the best configuration (kernel size = $(3, 5)$, stride = 1 and one input and output channel) are reported. The left and right contexts of 5 and 10 frames are

TABLE 5.5: Improvements in the accuracy of $\mathcal{M}_{ceb\_tel}$ mapping model for various structural and optimisation improvements

Exp ID	Model	% Accuracy	Model size
Baseline from previous work			
E0	Baseline	56.12	1.25M
Using context with baseline architecture			
E1	E0 + input_left=5	57.71	1.64M
E2	E1 + input_right=5	60.45	2.03M
E3	E2 + output_left=3	60.95	2.92M
E3′	E2 + output_left=3	60.32	5.02M
E4	E0 + input_left=10	57.75	2.03M
E5	E4 + input_right=10	61.05	2.81M
E6	E5 + output_left=3	61.56	3.70M
With convolutional layers			
E2′	E2 + conv_layers=3	60.52	2.38M
E5′	E5 + conv_layers=3	62.07	3.11M
Sequence modelling			
E7	Enc-Dec (*uni-direc*)	56.83	0.76M
E8	Enc-Dec (*bi-direc*)	60.35	1.07M
E9	E8+ Attention	**62.74**	1.21M
Multi-Encoder sequence modelling			
E10	Multi encoder-decoder	58.77	2.59M
E11	E10 + Rank loss	61.91	2.59M

experimented with. A marginal gain is achieved at the cost of increased model capacity (E2′ and E5′).

The same source-target pair model is trained using an encoder-decoder architecture. Various configurations with different RNN layer types including simple RNN, GRU and LSTM are experimented with. As an attentional decoder in also experimented for the encoder-decoder architecture, attention types from simpler dot attention to Luong (Luong et al., 2015) and Bahdanau (Bahdanau et al., 2016) attention are applied. Results with the best configuration (GRU layers and dot attention) are reported. A significant gain is attained by using a bidirectional GRU (E8) rather than a unidirectional one (E7). It achieves an accuracy of 60.35% with a model size of only 1.07 million parameters which is the smallest among baseline and all the improvements made on that. Adding an attention module to this, the model is still the smallest in size but outperforms all other configurations. E9 even outperforms a model (E6) which is three times bigger in terms of model capacity.

The best encoder-decoder model (bidirectional GRU and attention) is then leveraged for the proposed MESD model. In the MESD model for a target language, there are three encoders and only one decoder. The model size is 2.59 million parameters which is about 1.5 times smaller than the accumulative model sizes of baseline mapping models required for a target language. Initially, MESD is trained using the mean loss of all the

languages in each batch, but the performance gets worse significantly (E10). Though the criteria for LR annealing in this model is the average accuracy of all three languages, accuracy for only *ceb* is shown to be consistent with the rest of the entries. The accuracy is significantly dropped using the MESD model. On analysis, it is found that the performance for one of the languages is monotonically increasing but decreasing for *ceb* encoder-decoder part. It leads us to optimisation improvements discussed in the next section.

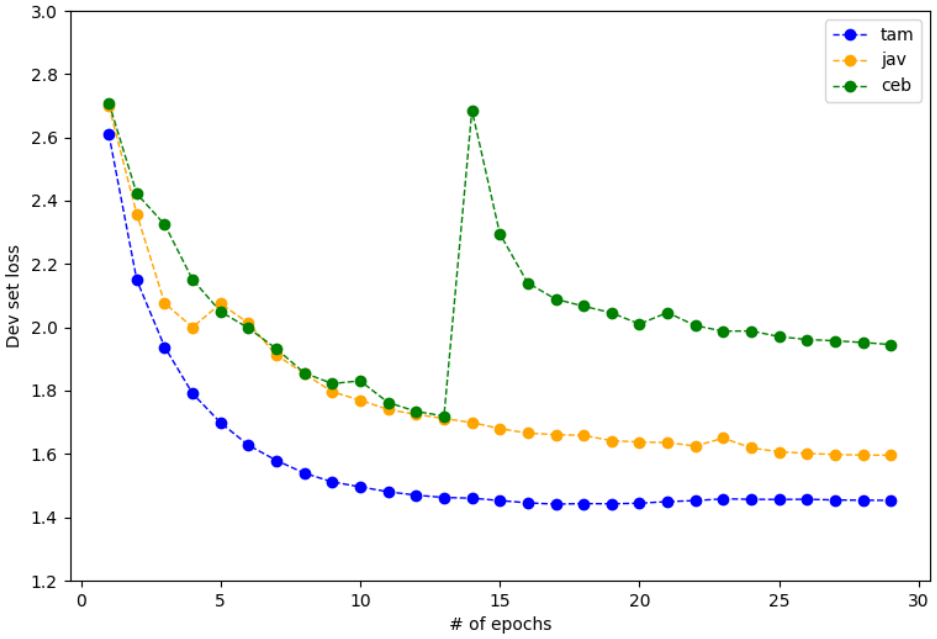
5.6.4 Optimisation improvements

As described earlier in Section 5.4.1, using the mean weighted loss of all source languages for the training of a target language MESD mapping model might cause biased training for some source languages. To cope with this issue, the use of rank sum weighting is introduced for source languages' losses (as described in Section 5.4.1). For *tel* target language, losses of all source languages are compared for mean weighted and rank sum weighted losses approaches. Losses and %PER of *tel* target language dev set for all the source languages over the epochs are shown in Figure 5.4 and 5.5 respectively.

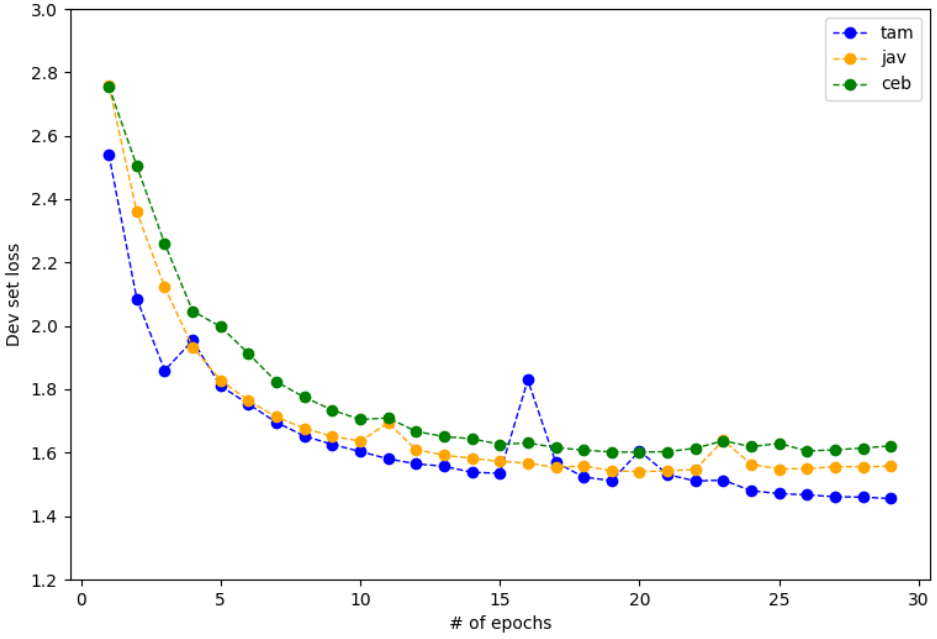
It is evident from the Figure 5.4 that at the point of convergence, deviation in losses for all the encoder-decoder pairs is very high when a mapping model is trained with mean weighted losses. However, the losses for source languages are comparatively very close to each other in the case of training using rank sum weighted losses. Similarly, the same trend can be observed for %PER of *tel* dev set for all the source languages in Figure 5.5. In Figure 5.5, %PER for *tam -tel* pair is lower in case of mean weighted losses (Figure 5.5a) if compared with rank sum weighted losses (Figure 5.5b). However, highly divergent %PER with mean losses make the rest of the source languages have high %PER which is significantly reduced when rank sum weighted losses are used.

TABLE 5.6: Accuracy of MESD mapping models in comparison with the baseline mapping models considering top-1 ($n = 1$) mapped class. The accuracy of a source-target mapping model is calculated using Algorithm 2

Target Lang	Source Lang	*mapping model* accuracy	
		Baseline	*MESD*
tam	tel	42.91	47.53
	ceb	44.43	56.85
	jav	41.89	58.22
tel	tam	54.44	76.87
	ceb	35.51	72.30
	jav	50.54	75.44
ceb	tam	45.73	46.40
	tel	46.17	46.80
	jav	47.04	47.49
jav	tam	47.81	47.83
	tel	48.29	48.33
	ceb	48.05	48.54

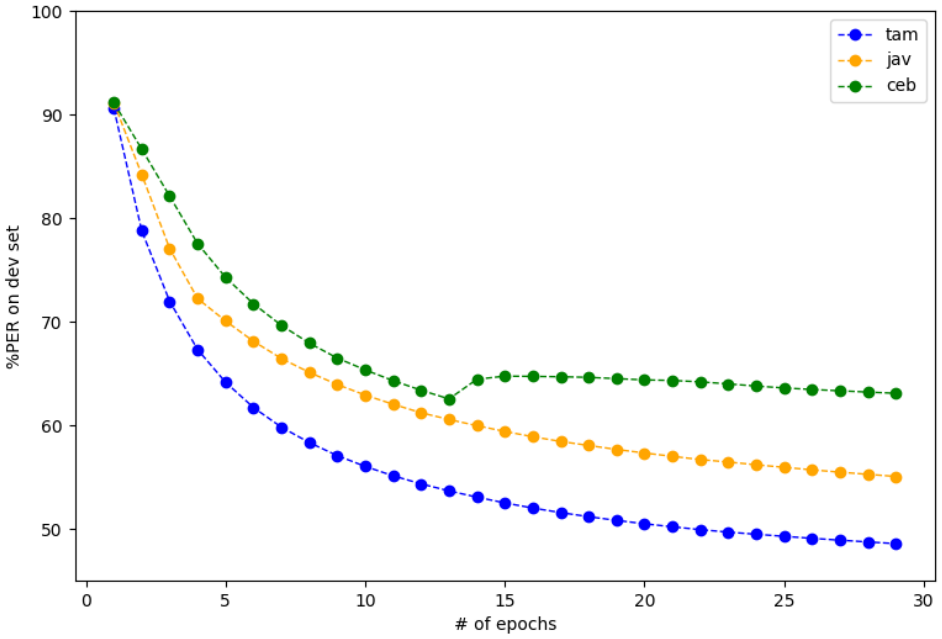


(A) Mean weight loss

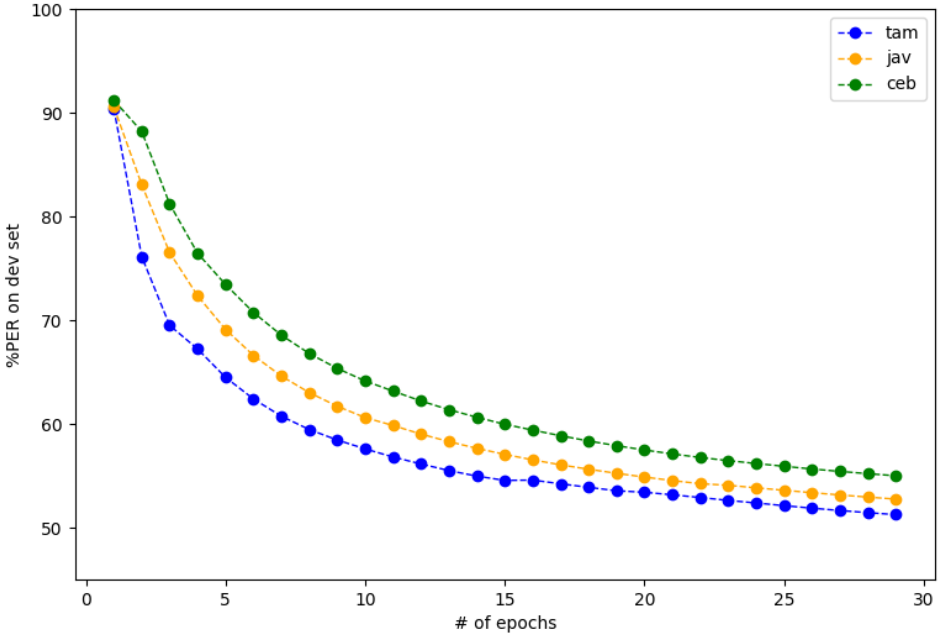


(B) Rank sum weight loss

FIGURE 5.4: Losses of the target language (*tel*) dev set for all the source languages (*tam* , *jav* and *ceb*) with mean weights (5.4a) and rank sum weights (5.4b) approaches. It is evident that over the epochs, the losses of all the source languages are less diverged with rank sum weights when compared with mean weights.



(A) Mean weight loss



(B) Rank sum weight loss

FIGURE 5.5: %PER of the target language (*tel*) dev set for all the source languages (*tam* , *jav* and *ceb*) with mean weights (5.4a) and rank sum weights (5.4b) approaches. It is evident that over the epochs, %PER for all the source languages are less diverged with rank sum weights when compared with mean weights.

TABLE 5.7: Statistics of speech and non-speech examples (in millions) for training and evaluation of mapping models. *total:* Number of total examples, *NS:* Number of non-speech frames (examples), *%:* Percentage of speech examples in the data set

Lang	Train			Dev			Eval		
	total	NS	%	total	NS	%	total	NS	%
Tamil *(tam)*	3.23	0.15	95.47	0.36	0.02	95.46	1.66	1.35	29.50
Telugu *(tel)*	3.23	0.11	96.64	0.36	0.01	96.71	1.91	1.27	23.50
Cebuano *(ceb)*	3.24	1.42	55.89	0.35	0.16	55.68	1.94	1.43	26.64
Javanese *(jav)*	3.22	1.35	58.42	0.36	0.14	58.89	1.85	1.28	30.95

To cope with the drop in accuracy of the MESD model (`E10` when compared with `E9`), it is trained using rank sum weighted losses. It significantly increases the accuracy from 58.77% to 61.91%. This improvement in accuracy is about 3.2% if compared with mean loss training. Though this is not as good as the single encoder-decoder model (`E9`), MESD is used for the rest of the experiments due to two-fold advantages;

- It reduces the pain of training $N(N-1)$ models by reducing number of required models to only N
- The model size of an MSED model is smaller than the total size of three single encoder-decoder mapping models making processing faster

So, the MESD model with rank sum weighted loss is then used for training the mapping models for all the participating languages and the results are shown in Table 5.6. Though the MESD architecture yields an absolute gain of up to 36%, *ceb* and *jav* languages show a very marginal gain when compared with the baseline. However, still, an overall gain of 10% is achieved over the baseline model.

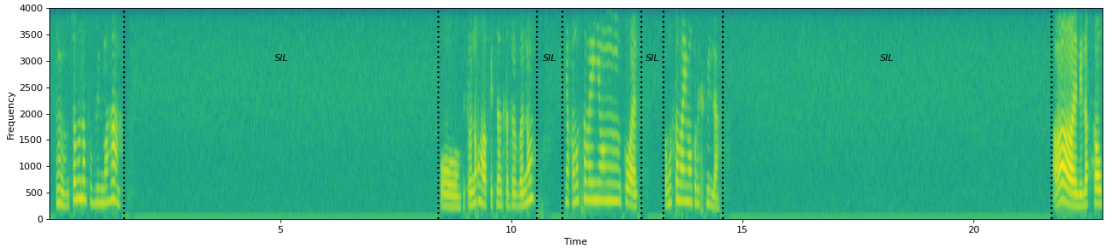
5.6.5 Analysis and Discussion

Since the MESD model improves mapping massively for some language pairs and very marginal for others, the trends in the accuracies of the mapping models are investigated. The analysis reveals that the Babel data set consists of a lot of non-speech segments. As the mapping models' accuracies are reported on the Babel dev set which contains significant non-speech utterances (which are labelled as silence phoneme), the performance of the mapping model is very much dependent on the mapping accuracy of the silence phoneme. To get statistics of speech and non-speech frames in the data sets, a frame-wise analysis is done on data alignments.

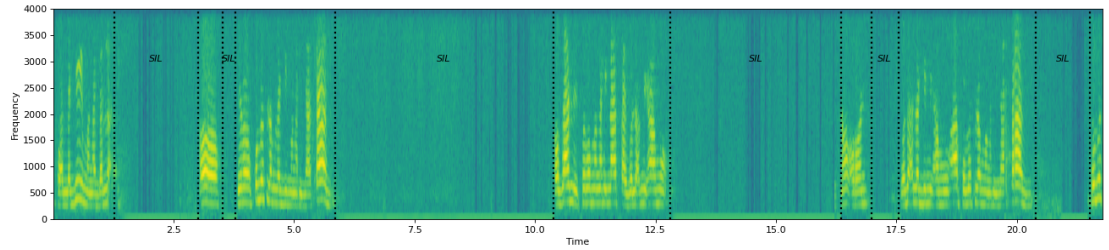
Train, dev and eval data sets are force-aligned using the in-domain monolingual pre-trained hybrid DNN-HMM ASR system (described in Section 5.6.1). During analysis of these alignments, the frames with silence phonemes as the most probable class are considered non-speech frames while the rest are considered speech frames. Earlier in Table 5.2 the number of train, dev and eval sets' examples have been tabulated, here Table 5.7 summarises the alignment analysis and shows the statistics of speech and

non-speech examples for each data set. It is evident that only around 30% data of the evaluation set consists of speech frames which implies that the accuracy calculation of a mapping model is highly dependent on the accuracy of silence phoneme mapping.

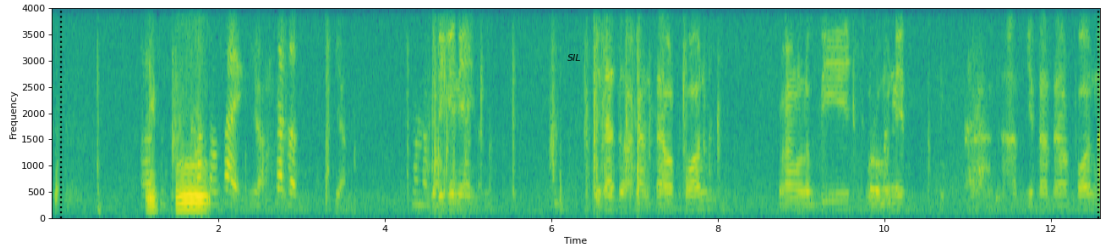
Though the non-speech utterances are dropped during the sampling for the train set, 45% of the training data of *ceb* and *jav* still consists of non-speech frames. Since a significant portion of the Babel data set is conversational speech recorded in noisy environments, a lot of *ceb* and *jav* utterances have a significant duration of non-speech in a segment with very little speech. A few different cases of a lot of non-speech are shown in Figure 5.6. A spectrogram of a simple case is shown in Figure 5.6a where a very long utterance contains a few speech segments. Non-speech in this case is just silence. Figure 5.6b shows a spectrogram where an utterance has a lot of non-speech but the non-speech is silence but a background noise. Since Babel data contains telephonic conversation in an open environment, the spectrogram of a segment is shown in Figure 5.6c where only a short single word is spoken by the speaker of interest at the start of a more than 12 seconds long audio. The rest of the audio contains considerably loud background talking (as evident from the spectrogram). Since anything other than speech is not annotated and considered silence by the ASR systems, such examples are very noisy for mapping



(A) A *ceb* training example with a significant amount of silence



(B) A *ceb* training example with a significant background noise



(C) A *jav* training example with a lot of background talking

FIGURE 5.6: Spectrograms of a few segments of Babel data sets are shown as examples of speech versus non-speech analysis

TABLE 5.8: Accuracy of mapping models with and w/o considering non-speech (NS) frames for top-1 ($n = 1$) mapped class. The accuracy of a source-target mapping model is calculated using Algorithm 2

Target Lang	Source Lang	w/ NS		w/o NS	
		Baseline	*MESD*	*Baseline*	*MESD*
tam	*tel*	42.91	47.53	49.31	51.22
	ceb	44.43	56.85	38.51	43.50
	jav	41.89	58.22	42.30	46.07
tel	*tam*	54.44	76.87	54.12	55.72
	ceb	35.51	72.30	47.70	50.50
	jav	50.54	75.44	49.41	52.63
ceb	*tam*	45.73	46.40	45.45	48.15
	tel	46.17	46.80	46.97	49.86
	jav	47.04	47.49	50.07	52.02
jav	*tam*	47.81	47.83	44.23	47.90
	tel	48.29	48.33	45.03	48.77
	ceb	48.05	48.54	44.16	48.73

models to train.

The noisy training examples are unavoidable unless the sampling is done in a very controlled training which might cause a selection only from very clean speech and thus a mismatch between the training and evaluation set. However, for more insightful results from mapping models, evaluation is carried out on the evaluation set ignoring the mapping of silence phonemes. Table 5.8 shows the accuracies of mapping models for $n = 1$ when the silence phonemes are ignored. The accuracies of mapping models in Table 5.8 are consistent across the languages and more comprehensible. Though the performance gap is not very huge, MESD still performs around 3% on average better than the baseline mapping models. Mapping from a source language ASR which is from the same language family as the target language yields the highest accuracy. The trend remains the same for both baseline and MESD models.

5.7 Summary

In the last chapter (Chapter 4), it has been proposed to learn the patterns in the output from a source language acoustic model and the ground-truth phonemic transcriptions of a target language utterance. The objective of this chapter has been to explain the proposed mapping models to learn the patterns. A feed-forward neural network with four layers has been trained as a baseline mapping model. Given a posterior distribution from a source language acoustic model for a target language utterance, the objective of a mapping model is to map it to the posterior distribution from the target language acoustic model. The accuracy of a mapping model is calculated as the ratio of the correctly mapped frames to the total number of frames in the test set. A frame is considered a correctly mapped frame if the most probable mapped class is the same as the most probable class in the target posteriors. Mapping models have been trained

for four low-resource Babel languages i.e. Tamil, Telugu, Cebuano and Javanese. A separate mapping model has been trained for each source-target language pair and an accuracy of 46.44% has been achieved.

However, baseline mapping models are based on various assumptions such as the mutual independence of frames. So, many structural and optimisation modifications have been explored for mapping models. A multi-encoder single decoder model (MESD) has been proposed to reduce the number of mapping models from $N \cdot (N-1)$ to N for N number of participating languages. An average accuracy of 49.59% has been achieved by MESD models.

Both baseline and MESD mapping models are employed to bridge some gaps in multilingual speech recognition research to improve low-resource speech recognition which are discussed in the next chapters.

Chapter 6

Cross-lingual acoustic-phonetic similarities

In this chapter, proposed mapping models are employed to investigate the cross-lingual acoustic-phonetic similarities. As discussed in Section 3.4.1, multilingual speech recognition systems mostly benefit low-resource languages but suffer degradation in the performance of several languages relative to their monolingual counterparts (Hou et al., 2020, Pratap et al., 2020a). Limited studies have focused on understanding the languages' behaviour in multilingual speech recognition setups. Mapping models are trained on top of monolingual ASR systems of different languages. An analysis is done to measure the similarities between mapped and the target posterior distributions against a target language speech signal.

Section 6.1 revisits the literature to underscore the motivation of this work to measure cross-lingual similarities. Section 6.2 explains the way mapping models are being exploited for cross-lingual acoustic-phonetic similarities. Experimental setup details are discussed in Section 6.3. Results are presented and discussed in Section 6.4 while the work is briefly summarised in Section 6.5.

6.1 Motivation and background

This work is motivated by the fact that the previous works show that multilingual speech recognition systems do not yield a significant reduction in word error rate for various languages including English, German, French and several others (Conneau et al., 2021, Hou et al., 2020, Pratap et al., 2020a). Many open-source data sets are available for these languages and are generally regarded as resource-rich languages. Most of the multilingual models, which include resource-rich languages, are trained on unbalanced

This chapter is based on our research paper in Interspeech 2022;
M. U. Farooq and T. Hain, *"Investigating the Impact of Cross-lingual Acoustic-Phonetic Similarities on Multilingual Speech Recognition"*, Interspeech, 2022.

data. So, degradation in the performance of these languages is attributed to increased confusion for them in the multilingual setup (Pratap et al., 2020a). Pratap et al. (Pratap et al., 2020a) have trained several massively multilingual speech recognition systems with different configurations. The word error rate gets worse for many languages even for a fairly large model of 1B parameters when no language information is provided. Though the trend of %WER getting worse is very obvious in the resource-high languages zone, there are several spikes even in the low-resource languages region as well. It implies that attributing this trend to the unbalanced sampling is not guaranteed to be true. A similar trend has been shown for many languages by (Hou et al., 2020). Even the state-of-the-art unsupervised wav2vec2.0 model does not improve the error rate for the languages including English, German and French etc. while using their full amounts in the MLS dataset for fine-tuning (Conneau et al., 2021). Conneau et al. (Conneau et al., 2021) mention language interference as the reason for the increased error rate. However, no experimental evidence of any of these justifications could be found in the literature.

The argument about unbalanced sampling can simply be verified by training a multilingual speech recognition system using a balanced data set. The experiments in Section 4.4.2 shows that some languages' ASR error rate increases even with the balanced sampling. It implies that the long-tail problem does not have much effect on the degradation of WER of these languages. The other argument about increased confusion might have an effect potentially because a phoneme-based multilingual ASR, for instance, is trained on the basis of multilingual assumption (Schultz and Waibel, 2001). Phonemes of several languages are represented by the same IPA representations but might have different underlying acoustic realisations. In Section 3.4.1 and Section 4.1, earlier studies in the context of efficient data sharing among languages have been discussed. Recently, some efforts have been made to interpret the learning of multilingual speech recognition systems (Feng et al., 2021, Żelasko et al., 2020). The phoneme error rate of each phoneme in monolingual ASR has been compared with that of multilingual system (Żelasko et al., 2020) (shown in Figure Figure 4.1). However, no monotonic trend has been observed with the growing number of languages the phonemes shared. The authors described this as *"unexpected"* because the phonemes shared by more languages provide more training data and thus the expected error trend would be decreasing. Motivated by the fact that many languages with significant phonemes overlap pose performance degradation in the multilingual setups (Conneau et al., 2021, Feng et al., 2021, Hou et al., 2020, Pratap et al., 2020a, Żelasko et al., 2020), the objective and contribution of this work is to study acoustic-phonetic similarities to understand if the cross-lingual phoneme sharing is truly a sharing and how does it impact on multilingual speech recognition setups?

In this chapter, a novel technique is proposed to estimate the cross-lingual acoustic-phonetic similarities for CD hybrid DNN-HMM acoustic models to study the cross-lingual similarities. A hybrid DNN-HMM system is preferred over end-to-end modelling to avoid the influence of the entangled language model in e2e speech recognition systems. The same data set of three West Germanic languages (English, German and Dutch),

as used for phoneme-based ASR experiments in Section 4.4.2, is used to study the impact on multilingual performance. The behaviour of monolingual acoustic models of different source languages against a speech signal of the target language is studied by differentiating the posterior distributions. To compare distributions of source and target AMs, a separate baseline mapping model is trained for each <*source, target*> pair to map source posterior distributions to the target posterior distributions.

6.2 Cross-lingual acoustic-phonetic similarities

The motivation for research into multilingual speech recognition is based on the assumption that the articulatory representations of phonemes are very close across the languages and can be considered language-independent units (Schultz and Waibel, 2001). However, several languages with substantial cross-lingual phoneme sharing exhibit poorer performance in multilingual setups. This calls for a study to understand the reason for degradation or improvement in multilingual setups when compared with corresponding monolingual systems.

Though e2e speech recognition systems have dominated the conventional HMM-based ASR system, hybrid DNN-HMM systems still outperform e2e ASRs with the limited amounts of training data (Povey et al., 2016). Furthermore, the output from e2e speech recognition systems is stronger because they are able to learn some language characteristics. Thus their output is influenced by the implicit language model which adversely affects the acoustic analysis.

In hybrid speech recognition systems, a deep neural network is trained to produce a posterior distribution of tied states of HMM models. For a language with Q number of phonemes, a maximum number of Q^n polyphonemes with n context width can exist i.e. English language with 44 phonemes can have $44^2 = 1936$ biphones. Each of these biphones can be modelled by a separate HMM model of S number of states. So theoretically, the total number of tied states for a language with Q number of phonemes and S number of states per HMM model is given by $Q^n \times S$. Usually, the conventional HMM models have been trained using triphones considering one left and one right phoneme along with the centre one. However, any number of context phonemes can be considered while training the model. So, the term polyphonemes is conventionally used here for phonemes with any context width greater than one. As the theoretically possible states are $Q^n \times S$, the number of states increases exponentially when either the context width or the number of phonemes is increased. For instance, training a triphone 3-state HMM model of a language with 20 phonemes would have $20^3 \times 3 = 24000$ states. It implies that there would be an output layer of size 24000 of the neural network of a DNN-HMM-based ASR system. It is very hard to train a neural network with such a large output layer and the performance of such a system is not expected to be very good. In reality, many polyphonemes out of these possible combinations might never occur in language or many are quite similar to each other. One possible way of reducing the number of possible states is to drop the ones which never occur in the training set.

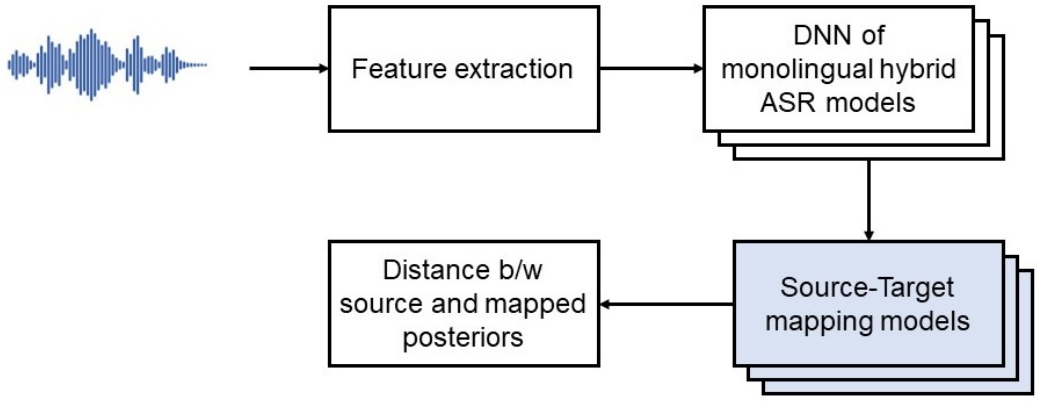


FIGURE 6.1: Proposed system architecture for analysis of cross-lingual acoustic phonetic similarities. Mapped posteriors from diverse monolingual source acoustic models are compared with the target posteriors.

However, that poses a potential risk of losing the polyphonemes which never occur in the training set but might present in the language or evaluation set. So, the total number of states (and thus the output layer dimension of DNN) is reduced by clustering many polyphonemes together. For instance, the Kaldi toolkit builds a decision tree based on statistics from GMM-HMM model alignments. The bottom layers of these trees are the number of clustered states. Since these trees are dependent on the statistics from the alignments from a language-dependent ASR, each language yields a different phonetic decision tree in its monolingual ASR. So, the number of tied states differs for each language and thus the size of the output layer of DNN of each DNN-HMM model is different. Not only the size is different, but each output class might represent a different set of polyphonemes. Furthermore, each class might represent a single polyphoneme or multiple polyphonemes based on the clustering. It means that the posterior distributions from the two models are not directly comparable even for an identical speech signal and through the monolingual models of quite closer languages.

In this work, a data-driven approach, to transform posteriors from diverse models to a directly comparable form, is proposed. Having the posterior distributions from various models given the same input speech signal, it can be analysed how different monolingual models perceive an identical speech signal. As the underlying assumption of mapping models is that some patterns might exist between posterior distributions from the target model and a source language ASR given a target language speech signal, the mapping model is not able to learn well when a source model significantly confuses a phoneme with some other phoneme or multiple phonemes. So, a similarity measure is calculated between the target and the mapped posterior distributions to approximate the closeness of the two languages. Furthermore, these measures are compared for different source languages to estimate the cross-lingual acoustic-phonetic similarities.

6.2.1 Similarity measure

Let M_A and M_{S_i} be the monolingual acoustic models of target and source languages respectively. The target language is the language for which the similarity is being measured against the source languages. A baseline mapping model $\mathcal{M}_{S_iA}$ is trained to translate posteriors $P_{S_i} \in \mathbb{R}^{d_{S_i}}$ from M_{S_i} to the posteriors $P_{S_iA} \in \mathbb{R}^{d_A}$ where d_A is the dimension of posteriors from M_A. As presented in the previous chapter, the mapping models are able to learn some language-related relationships between the posterior distributions of a source and the target acoustic models. They can learn the phonemes of the target language which are more amenable to cross-lingual transfer than the others. A few hours of speech data can give millions of examples that provide sufficient training data for the mapping model. KL divergence, the most widely used measure to differentiate two posterior distributions (Kullback and Leibler, 1951), is calculated as a similarity measure between the posterior distributions from the target language AM and the mapped posterior distributions from the source language AM. The proposed system architecture is shown in Figure 6.1.

Let $X = \{x_1, x_2, \ldots, x_T\}$ be a set of observations of the target language, for which posterior distributions ($P^Z = \{p_1^Z, p_2^Z, \ldots, p_T^Z\}$ where $Z \in \{A, S_i\}$) are attained from all monolingual acoustic models. Posteriors from source acoustic models (P^{S_i}) are mapped to target posteriors (P^{S_iA}) using *mapping model* $\mathcal{M}_{S_iA}$. The similarity between a source and the target language for a given set of observations is calculated as

$$D_X(M_T, M_{S_i}) = \frac{\sum_{t=1}^{T} p_t^A \cdot (\log p_t^A - \log p_t^{S_iA})}{T} \tag{6.1}$$

where D_X, the similarity measure, is the average KL divergence over all the examples. The value can range from the best 0 to the worst infinity. If two posterior distributions are perfectly matched, the value of the similarity measure would be 0 and can range from 0 to infinity otherwise.

6.3 Experimental setup

6.3.1 Data set

Experiment results are reported using three languages (English *en*, German *de* and Dutch *nl*) of the West Germanic family. From previous works and the results reported in Table 4.5, the performance of these languages is either degraded or shows a very minor improvement in multilingual setups despite a sufficient number of the shared phonemes (Figure 4.4).

The same data sets utilised for experimentation in Section 4.4.2 for phoneme-based ASR systems, are used for the study of cross-lingual acoustic phonetic similarities here. While the speech recognition systems discussed in Section 4.4.2 are used to train the mapping models on top of them, 30 hours of training set of each language is further divided into

29 hours for train and 1 hour for the validation set. A limited amount of data for each language is used in this study due to several reasons.

- On the phonemic level, 30 hours provide millions of examples for *mapping model* training (tabulated in Table 6.1). The training of mapping models has shown that it converges in early epochs (as shown in Figure 5.3) with the training data of this size, implying that the amount of data is sufficient for acoustic-phonetic similarity analysis.
- Experiments with the limited data for model fusion provide a realistic scenario for low-resource languages and the technique can be extended for resource-deficient languages.
- In continuation of the experiments from Section 4.4.2, the objective of the limited-resource setup has been to investigate whether the performance of the selected languages declines in multilingual scenarios. This decline is attributed to the dominance of their robust monolingual counterparts, which are trained on significantly larger amounts of data due to their resource-rich nature (Pratap et al., 2020a).

6.3.2 Speech recognition systems

Monolingual hybrid DNN-HMM models are trained for all the participating languages. 40 MFCCs are extracted for each frame of the speech signals using a window size of 25 ms and a shift of 10 ms. These features are then fed to DNN which consists of 12 factorised TDNN (TDNN-F) layers (Povey et al., 2018). Each TDNN-F hidden layer is of dimension 1024, factorised with a linear *'bottleneck'* dimension of 128. The acoustic model is trained using lattice-free MMI criterion (Povey et al., 2016). The output of the neural network is the posteriors of the clustered left biphone states. The clustering in each monolingual ASR training is different and thus the outputs from different acoustic models against an identical speech signal are not directly comparable. ASR systems are built using the Kaldi toolkit (Povey et al., 2011).

The Kaldi toolkit converts each phoneme into four further phonemes based on the position of a phoneme in a word. For example, a phoneme \ε\is converted into \ε_**B**\, \ε_**I**\, \ε_**E**\and \ε_**S**\where **B**, **I**, **E** and **S** represents beginning, intermediate, end and singleton positions. To avoid confusion, the terminologies '***phonemes***' (p_l) and '***positional phonemes***' are used for actual phonemes in a language and the Kaldi phonemes

TABLE 6.1: Examples (in millions) for training of mapping models for each target language. Train set: 29 hours; Dev set: 1 hour; Eval set is same as for the ASR

Language	Train	Valid	Test
en	3.48	0.12	0.24
de	3.35	0.12	0.24
nl	3.34	0.12	0.24

respectively. During alignments for the training of the models, if the phoneme \ε\occurs at the start of a word it is replaced by \ε_**B**\and it is changed with \ε_**E**\if it occurs at the end of a word. If the phoneme takes place somewhere between the other phonemes of a word, it is replaced by \ε_**I**\. However, there might be some very short words consisting of only a single phoneme, which is changed with \ε_**S**\. Furthermore, the Kaldi introduces two more phonemes i.e. *SIL* and *SPN* for silence and speech noise in the utterance. *SIL* phoneme is for silence on sentence, word or phoneme boundaries whereas *SPN* phoneme is to label the speech segments with spoken noises such as laughs, coughs or any other vocal noises made by speakers which do not have any linguistic information. Both of these phonemes are converted into five further phonemes on the same lines. Besides converting into four positional phonemes, the original *SIL* and *SPN* phonemes (without positional indication) are also retained. So, if a language has p_l phonemes, a Kaldi model is trained for more than $(4 \times p_l + 10)$ phonemes. Note that the word 'more than' is used since there are some disambiguation symbols as well. As the experimentation here is primarily on top of the LF-MMI models, it is worth mentioning here that LF-MMI models are usually trained for biphones using two-state HMM models (Povey et al., 2016). So, the total number of possible HMM-tied states for a language becomes more than $(2 \times (4 \times p_l + 10)^2)$. Though these states are clustered before training to reduce the number of classes, biphones with the same phoneme but different positional indications or HMM classes are not guaranteed to be in the same class which causes further challenges for learning of the mapping models.

Furthermore, Kaldi uses the logits (referred to as pseudo-likelihood in the Kaldi) of the neural network as emission probabilities of the HMM states. As described for the training of mapping models to map the posteriors, log-softmax is applied on these logits to convert them to posterior distributions for further processing i.e. ASR inference and training of the mapping models.

6.3.3 Mapping model

Baseline mapping models (described in Section 5.2) are trained for each source-target language pair. It is important to note that the mapping models from language A to B and from B to A are not trained on the same data. For the mapping model from A to B, the input consists of the source posteriors from the acoustic model of language A, and the target is the posteriors from the acoustic model of language B, using the audio data of language B. Conversely, for the mapping model from B to A, the input comprises the source posteriors from the acoustic model of language B, and the target is the posteriors from the acoustic model of language A, using the audio data of language A. Please be aware of the different audio input data in each case.

Each mapping model consists of three fully connected linear layers with a model size of about 1.25 million parameters. The target posteriors, used for the training of the mapping models, are extracted by decoding training data through a language-dependent ASR. KL-divergence loss is used for the training of the baseline mapping models. As

there are three participating languages, $n \cdot (n-1) = 3 \times (3-1) = 6$ mapping models are trained in total. The amount of training data for each target language is given in Table 6.1.

6.3.4 Similarity measure

All acoustic models used for experiments in this work are trained using lattice-free MMI criterion (Povey et al., 2016). Though the study can easily be extended for polyphonemes of any context, left biphones are modelled for the experimentation here and the term *biphones* will be used during the remainder of the discussion.

Given the phoneme sets of two languages (the target and a source language), biphones can be categorised into the following subsets of biphones:

- *Shared biphones:* If there is some overlap between the phoneme sets of the target and the source languages, there would be some biphones which occur in both (source and the target) languages. They can be further divided into two subgroups.
  - *Shared seen biphones (SS):* The set of shared biphones which are seen by both languages during training of their acoustic models.
  - *Shared unseen biphones (SU):* The set of shared biphones which are never seen by source language in its train set. One possibility is that the biphone might exist in the target language but the source language does not have such a combination. Another likely reason could be that the biphone does not exist in both of the languages. In the latter case, probably it will not affect the similarity measure a lot if no such biphone appears in the test set, but the mapping model is not expected to learn well in the earlier case. That might affect the performance of the mapping model.
- *Unshared biphones (U):* The biphones of the target language that are never seen by the source language due to non-overlapping phonemes.

As stated earlier, the similarity measure proposed in this work measures the similarity between the posterior distributions from the DNN of the hybrid ASR system. Each output class of these DNN models represent some clustered polyphonemes (or a single polyphoneme very rarely). Since hybrid DNN-HMM models cluster various polyphonemes into one output class, it does not guarantee that the biphones of the aforementioned categories would go to separate classes during the clustering. An output class might be a cluster of some shared, shared seen, shared unseen and unseen biphones. Or it could be any combination of these categories. However, for such clusters, the analysis of similarity distance is not expected to be very clear. It is discussed further in the results section, but the analysis is restricted to the classes which contain only clusters consisting of only one of the above categories. Furthermore, due to Kaldi's positional phonemes, the problem becomes more challenging since some positional phonemes of a

phoneme could be clustered with the same category but some of them might go with different category's positional phonemes. Additionally, positional phonemes increase the output dimensionality of the DNN and make it challenging for mapping models to learn the mappings.

A similarity measure is calculated for each of the aforementioned cases and an analysis of trends in distance measure is carried out to comprehend the language similarities and their behaviour in the multilingual setups.

6.4 Results and discussion

6.4.1 Baseline speech recognition systems

Monolingual (*mono*) baseline systems are the language-dependent acoustic, pronunciation and language models which are trained on a language-specific data set. The train sets of all the languages are then mixed to train multilingual (*multi*) acoustic and language models. The multilingual acoustic model is then used with the monolingual language model of the target language which is termed as *mono-lm* in the reported results. As described earlier usually the logits are used as emission probabilities of HMM states in Kaldi, log-softmax is applied before any processing in the experimentation here. An analysis shows that it is hard for mapping models to learn with a broader range of values of output distributions in the case of pseudo-likelihood (or logits). So, log-softmax is applied to compress the values of output distributions between 0 to 1. The results of the baseline systems for all the languages are given in Table 6.2 in terms of %WER and %PER. Since only the language model is changed for *mono-lm*, thus the PER remains unchanged when compared with that of a multilingual system. The results show that the error for all the languages increases in multilingual setup despite the balanced data duration for each language. This indicates that the reason for performance degradation in multilingual setups can not only be attributed to rich resources of these languages or unbalanced data sampling.

According to the assumption of multilingual systems (Schultz and Waibel, 2001) discussed earlier in Section 6.2 if the articulatory representations of phonemes are considered language-independent units then the performance of shared phonemes should improve with more training data in the multilingual systems. The languages, being

TABLE 6.2: Baseline ASR performance in terms of %WER and %PER. *mono*: language-dependent ASR system, *multi*: language-independent ASR system, *mono-lm*: language-independent acoustic model with language-dependent language model

Language	%WER/%PER		
	mono	*multi*	*mono-lm*
English *(en)*	43.84/28.10	47.48/31.06	46.40/31.06
German *(de)*	37.77/26.86	40.81/28.35	38.11/28.35
Dutch *(nl)*	37.94/21.40	58.84/36.16	52.33/36.16

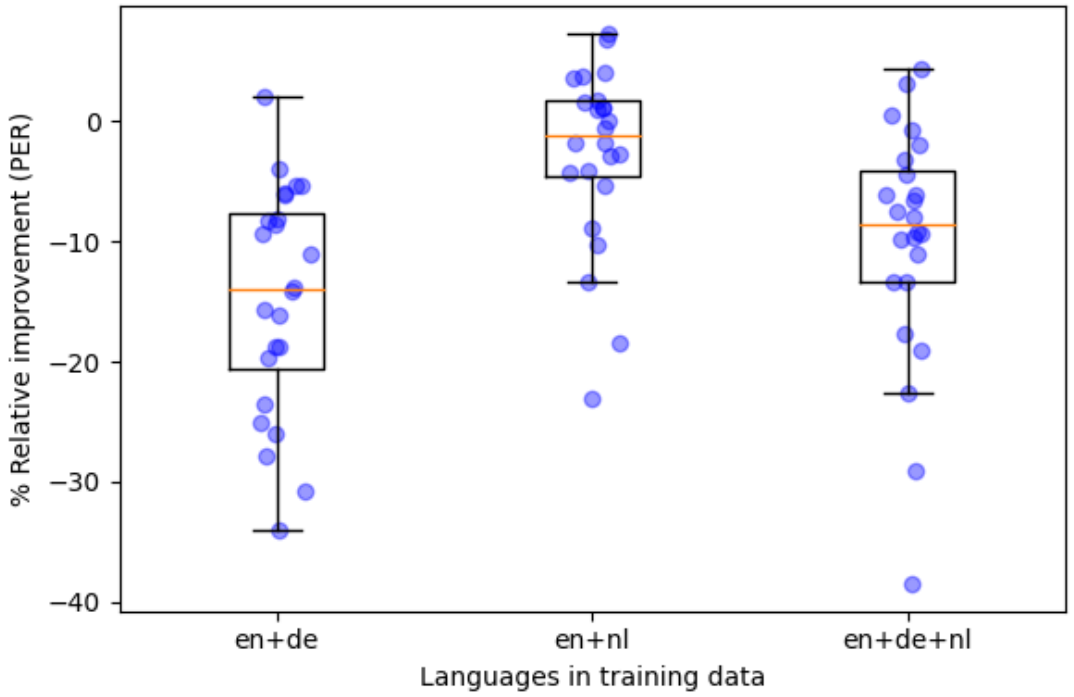
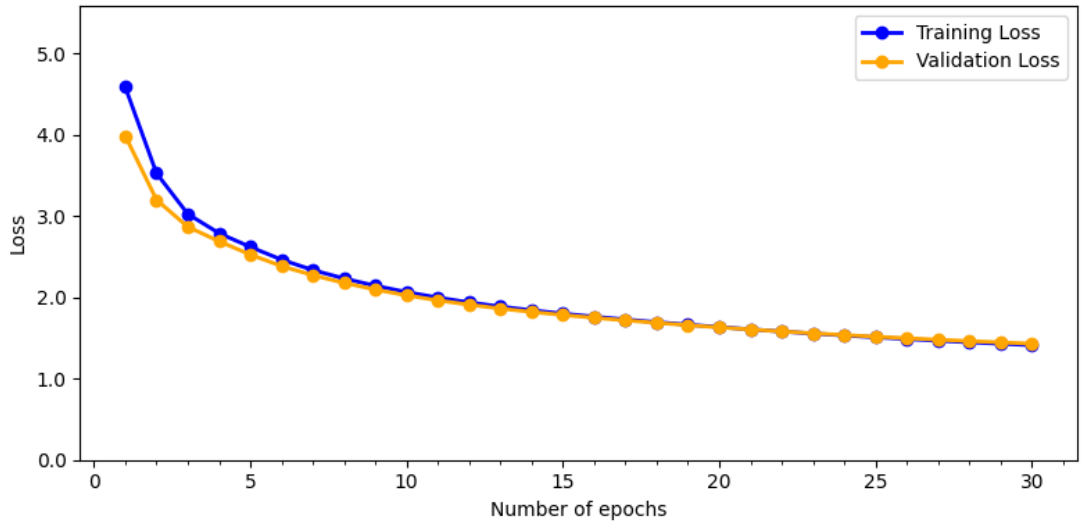


FIGURE 6.2: % Relative improvement in PER per shared phoneme compared with monolingual ASR for *en* target language

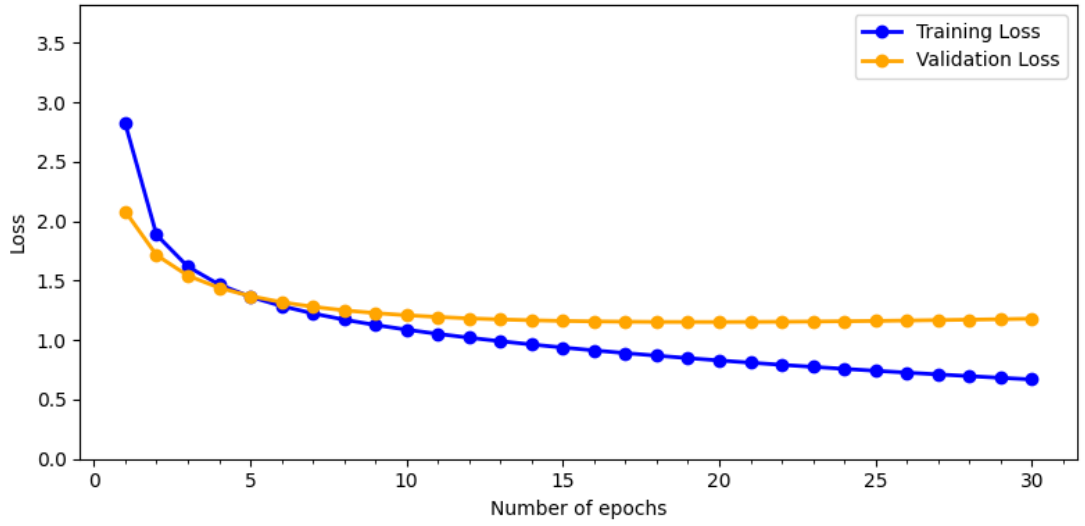
studied here, have an overlapping set of 24 phonemes. As a case study of *en* as the target language, the relative improvement in PER of shared phonemes is analysed with gradually increasing the languages in the training data. In Figure 6.2, the relative improvement in English phoneme error rate is shown with two bilingual models ($en + nl$ and $en + de$) and one multilingual model ($en + de + nl$). Bilingual models are trained by mixing the training data of English with either Dutch or German. In the case of the multilingual model, data from all the languages is mixed. It is evident from the results that even the performance of shared phonemes is degraded in bilingual and multilingual setups, though the bilingual model with *nl* causes less detrimental effects than *de* for the *en* language.

TABLE 6.3: Accuracy of baseline mapping models for MLS data set considering top n mapped classes. The accuracy of a source-target mapping model is calculated using Algorithm 2

Target Lang	Source Lang	*mapping model* accuracy			
		n=1	*n=2*	*n=5*	*n=10*
en	de	42.89	54.18	68.08	77.22
	nl	43.89	55.96	70.89	80.36
de	en	53.77	66.96	81.68	89.49
	nl	51.42	64.58	79.38	87.54
nl	en	46.17	58.11	72.21	81.03
	de	48.54	60.76	74.69	82.96



(A) Training curve for *de* _*en* mapping model



(B) Training curve for *nl* _*en* mapping model

FIGURE 6.3: Training curves for the mapping models from *de* and *nl* source languages to *en* target language

6.4.2 Mapping models

As described earlier a separate baseline mapping model is trained for each source-language pair of MLS data set. So, $N(N-1) = 3 \cdot 2 = 6$ mapping models are trained on the top of acoustic models of the baseline ASR systems (described in Section 4.4.2) using data sets outlined in Table 6.1. Since even the limited amounts of data provide millions of training examples, baseline mapping model training converges in early epochs for all the networks with an accuracy of nearly 50%. The training curves for the mapping models from source languages to *en* target language are shown in Figure 6.3.

Table 6.3 shows the accuracies of mapping models for all the target languages. As described earlier in Section 5.5.3, accuracies are shown considering top n classes with a

TABLE 6.4: Speech and non-speech examples in train, dev and test sets for mapping models. '*NS*' shows the non-speech frames in the data while '%' is the percentage of speech data among the total data. All the numbers (except % percentage) are in millions.

Lang	Train			Dev			Eval		
	total	NS	%	total	NS	%	total	NS	%
en	3.48	0.67	80.63	0.12	0.02	80.57	0.24	0.05	79.23
de	3.35	0.65	80.59	0.12	0.02	80.34	0.24	0.06	75.82
nl	3.26	0.79	75.73	0.12	0.03	76.49	0.24	0.06	73.26

range of different values of n. Similar to the trends in the mapping model on the Babel data set, the accuracy of mapping models is increased with the increasing number of n. However, the increase is not as much as in the mapping models for Babel data sets.

As it has been observed the Babel data sets contain a lot of non-speech segments and the overall performance of mapping models has been seriously influenced by the accuracy of those frames. So, the ratio of speech and non-speech frames is calculated for the MLS data set here as well. Since the model is trained for biphones, all the classes where non-speech phonemes are the central phonemes are considered non-speech examples. The statistics of speech and non-speech examples in the mapping model training data sets are summarised in Table 6.4. It can be seen that the ratio of speech to non-speech examples in the MLS data set is much higher than that for the Babel data set which means that there are a relatively very small number of non-speech segments in the data set here. The reason is that the MLS data set consists of read speech recorded in a very clean environment and segmented in a semi-automated way which minimises the non-speech recordings in the data.

6.4.3 Similarity analysis

Table 6.5 shows the similarity measure for English (*en*) test set when decoded through German (*de*) and Dutch (*nl*) acoustic models. English, German and Dutch vocabularies consist of 39, 45 and 39 phonemes respectively. However, as described earlier Kaldi divides each phoneme into four further phonemes. Furthermore, it includes two more phonemes for silence and spoken noises and divides each of them into five further phonemes. Additionally, some disambiguation symbols are also added to distinguish the word end. Further details on disambiguation symbols can be found in Kaldi documentation[1]. Consequently, Kaldi models are trained based on biphones using 176, 199 and 183 phones in English, German and Dutch. Theoretically, the possible number of HMM states, for example for English, are $2 \times 176^2 = 61952$. Similarly, German and Dutch can have up to 79202 and 66978 HMM states. However, after clustering the classes are reduced to 3344, 3232 and 3288 respectively. In clustering, shared seen biphones *(SS)* may share the same cluster with unseen *(SU)* or unshared biphones *(U)* and vice versa. So

[1] https://kaldi-asr.org/doc/graph.html#graph_disambig

for insightful observations, analysis is restricted to the clusters which have only one biphone (shown with *"R"* prefix). It is clear from the results that the KL divergence-based similarity measure increases from shared seen biphones towards unshared biphones.

Shared seen biphones (or restricted shared biphones) are supposed to be in source clusters as well which implies that the source acoustic model should comprehend these biphones during decoding. It implies that a mapping model should learn one-to-one mapping to the same biphone class if the multilingual assumption holds. It means that at a given time t, the most probable class of source and target posterior distributions ($max(p_t^A)$ and $max(p_t^{S_i})$) should belong to same biphone *(it is referred as correct Source Acoustic Model Class (SAMC))*. The results (*"Correct SAMC"* in tables) show that the source acoustic models are not very good at recognising these biphones but the lower values of KL divergence (*KL-Div* in parenthesis) indicate that it is easier for *mapping models* to learn a one-to-one mapping in these cases. It implies that the source ASR has a pattern in errors of these biphones sets which the mapping model could learn easily. On analysis, it appears that the source acoustic models confuse these biphones with several close biphones. For example, a biphone \i,z\from English test set is frequently confused with \i,s\and \ɛ,s\by German acoustic model.

Since the similarity measure is calculated between the two posterior distributions i.e. mapped and the target posteriors, trends in the entropy of the output of the mapping models ("Entropy" column in the Table 6.5, Table 6.6 and Table 6.7) are analysed for the given input speech utterances. It turns out that the trends in the entropy measure are the same as those of similarity measure (KL-Div). It suggests that the similarities between two languages can be estimated by just calculating the entropy of the mapping model which is less computationally expensive. The entropy of a mapping model is an indicative measure to see how much the model is confident in its output given the input

TABLE 6.5: Posterior distributions similarity analysis for the *en* test set. KL divergence (*KL-Div*) and entropy (*Entropy*) are computed between *en* target posteriors and the mapped posteriors from *de* and *nl* models. *% Correct SAMC* is the measure of source AM performance to recognise shared phonemes correctly. *KL-Div* and *Entropy* in parenthesis show the KL divergence of mapped posteriors for SAMC phomenes.
Phoneme sets; *SS*: Shared Seen, *RSS*: Restricted Shared Seen, *RSU*: Restricted Shared Unseen, *RU*: Restricted Unshared

AM	Biphone subsets	% Correct SAMC	KL-Div (SAMC)	Entropy (SAMC)
de	*SS*	37.08	1.32 (0.32)	2.43 (1.4)
	RSS	13.30	1.75 (0.84)	3.22 (2.64)
	RSU	-	1.88	3.40
	RU	-	2.27	3.59
nl	*SS*	38.04	1.23 (0.32)	2.21 (1.29)
	RSS	12.90	1.64 (0.82)	2.95 (2.44)
	RSU	-	1.71	2.86
	RU	-	1.81	3.1

TABLE 6.6: Posterior distributions similarity analysis for the *de* test set. KL divergence (*KL-Div*) and entropy (*Entropy*) are computed between *de* target posteriors and the mapped posteriors from *en* and *nl* models. *% Correct SAMC* is the measure of source AM performance to recognise shared phonemes correctly. *KL-Div* and *Entropy* in parenthesis show the KL divergence of mapped posteriors for SAMC phomenes.
Phoneme sets; *SS*: Shared Seen, *RSS*: Restricted Shared Seen, *RSU*: Restricted Shared Unseen, *RU*: Restricted Unshared

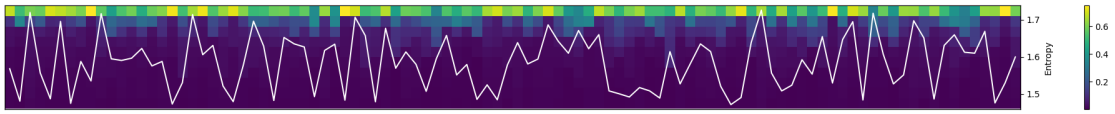
AM	Biphones subset	% Correct SAMC	KL-Div (SAMC)	Entropy (SAMC)
en	*SS*	43.56	0.83 (0.22)	1.76 (1.07)
	RSS	14.90	1.21 (0.74)	2.52 (2.12)
	RSU	-	1.15	2.21
	RU	-	1.27	2.54
nl	*SS*	36.37	1.05 (0.31)	1.95 (1.24)
	RSS	13.68	1.38 (0.82)	2.5 (2.09)
	RSU	-	1.54	2.67
	RU	-	1.41	2.56

speech signal. The value of entropy can range from 0 to $\log(C)$ where C is the number of output classes of the DNN. If the entropy is equal to $\log(C)$, it means that all the classes are equally likely. In terms of mapping models, it implies that a mapping model has not learnt a clear pattern for the given example.

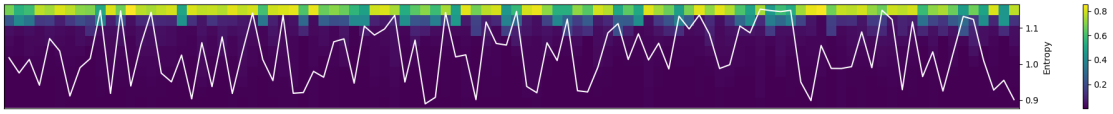
In the case of unseen *(RSU)* and unshared *(RU)* biphones, the entropy values for mapped posteriors increase compared to all the language pairs. Since unseen and unshared biphones are not seen by source language speech recognition systems, a one-to-one mapping is not expected to be learnt at least for unshared phonemes as those biphones do not exist in the source language. Thus '*Correct SAMC*' measure cannot be used for

TABLE 6.7: Posterior distributions similarity analysis for the *nl* test set. KL divergence (*KL-Div*) and entropy (*Entropy*) are computed between *nl* target posteriors and the mapped posteriors from *en* and *de* models. *% Correct SAMC* is the measure of source AM performance to recognise shared phonemes correctly. *KL-Div* and *Entropy* in parenthesis show the KL divergence of mapped posteriors for SAMC phomenes.
Phoneme sets; *SS*: Shared Seen, *RSS*: Restricted Shared Seen, *RSU*: Restricted Shared Unseen, *RU*: Restricted Unshared

AM	Biphones subset	% Correct SAMC	KL-Div (SAMC)	Entropy (SAMC)
en	*SS*	46.51	1.33 (0.37)	1.75 (0.95)
	RSS	15.43	1.87 (1.26)	2.36 (1.77)
	RSU	-	2.113	2.66
	RU	-	2.173	2.69
de	*SS*	40.90	1.44 (0.46)	1.89 (1.19)
	RSS	17.30	1.88 (1.15)	2.33 (1.81)
	RSU	-	2.04	2.43
	RU	-	2.13	2.58



(A) de to en mapping model

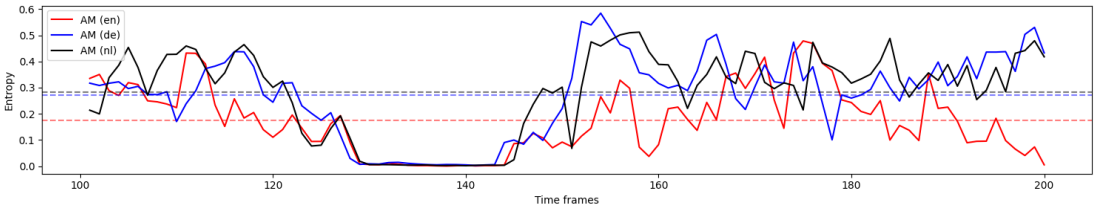


(B) nl to en mapping model

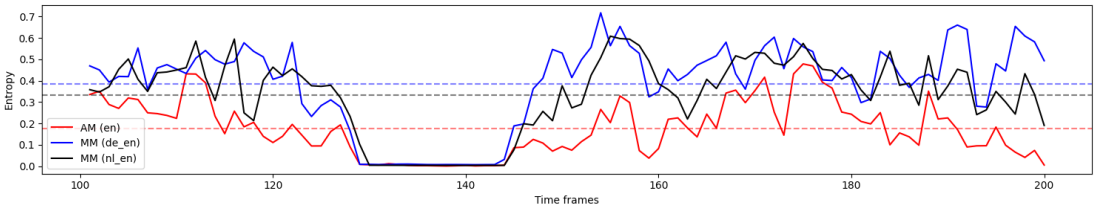
FIGURE 6.4: Posteriorogram and entropy plot of N_{de-en} and N_{nl-en} mapping models with one-hot vectors as input. Behaviour for only a hundred source biphone classes is shown. The posteriorogram shows sorted probabilities of the top ten mapped classes. Each box on the horizontal axis is a one-hot vector of the source language and on the vertical axis is the probability of a mapped output class

unshared and unseen biphones (in the above Tables). It is relatively more challenging for mapping models to learn a mapping for unseen and unshared phonemes compared to the seen biphones. These classes show the lowest performance in mapping model accuracies. Analysing mapping model accuracies (of the Table 6.3) reveals that the mapping models accuracy for unseen and unshared classes is only around 25% and 35% respectively whereas this number is around 65% in case of seen biphones. However, this analysis is restricted to again restricted (*'R'*) classes where only one biphone is among the clustered classes. Analysis shows that mapping models try to map to the closest class in case of unseen and unshared classes which remains challenging, however. To keep a brief discussion, most of the analysis is shown around *en* target language. The same patterns are observed for the remaining two languages (Table 6.6 and Table 6.7).

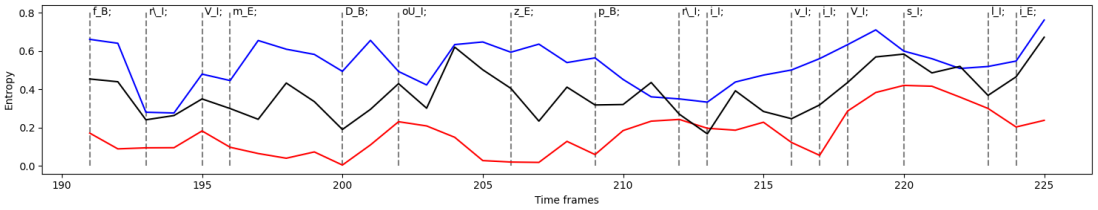
To study the learning of a mapping model further, it is sequentially fed the one-hot vectors as input to get insights on which classes learnt the mapping better. As a case study, the posteriorogram and entropy of mapping models from source languages to *en* is analysed. As the number of input and output classes are of the magnitude of thousands, behaviour for the top n source classes, which are mapped to the target classes with minimum entropy (more confidently), is plotted for better visualisations. Entropy and posteriorgram from both networks, N_{de-en} and N_{nl-en}, for same n ($n = 100$) is visualised in Figure 6.4. Only the top ten most probable mapped output classes are shown in the sorted posteriorogram. Each box on the vertical axis shows an output class with the most probable class on the top. It is worth mentioning that for each input class (one-hot vector) on the horizontal axis, the mapped output classes are different from the output classes of its neighbouring input frames. As only the top 10 most probable mapped classes are visualised, entropy at each input frame is also plotted (white solid line) to see the overall confidence of the mapping model for the given input class. It can be seen that for the same input one-hot vectors, the entropy range is lower for *nl-en mapping model* than the range for *de-en* model. It evidences that *nl-en mapping model* could learn better mappings and *nl* phonemes are more amenable to transfer to *en* phonemes



(A) Normalised entropies of *en* , *de* and *nl* acoustic models given an identical input speech utterance of *en*



(B) Normalised entropies of *en* acoustic model and mapped posteriors from $\mathcal{M}_{de\_en}$ and $\mathcal{M}_{nl\_en}$ mapping models given the same input speech utterance of *en*



(C) Selected frames from above data with phoneme annotation to visualise trends of different models for different phonemes (annotated in X-SAMPA)

FIGURE 6.5: For an input speech utterance of *en* language, normalised entropies of posterior distributions from all acoustic models (6.5a), *en* acoustic model and N_{de_en} and N_{nl_en} mapping models (6.5b), and selected frames with phonemes annotation (6.5c).

when compared with the same ability of *de* phonemes.

Though the entropy of the mapping model is shown in Figure 6.4 to see the confidence of the mapping models in the output posterior distributions, in-domain acoustic models do not show a zero entropy during the interference. Since the ASR systems are also data-driven and trained on training data, they are often not very confident (or at least not uniformly confident for all the input frames). So although the figure tells us about the entropy (or thus confidence) of mapping models given one-hot vector inputs, it does not inform on the quality of the training of the mapping models. Since the outputs of an in-domain acoustic model are used as targets to train mapping models, the performance of the mapping models depends on the performance of the in-domain acoustic model. So, to compare the confidence of the mapping models with that of an ASR, the entropy measures from different acoustic and mapping models are analysed given an identical speech utterance.

As an example to illustrate, an English(*en*) speech utterance is decoded through an in-domain (trained on *en* data) and two source language acoustic models (trained on *de* and *nl* data). Entropy is measured for all the frames and plotted in Figure 6.5a.

TABLE 6.8: Mean normalised entropies for test sets of all the target languages from all the acoustic and mapping models

Target Lang.	Source Languages (Normalised entropy)					
	en		*de*		*nl*	
	ASR	MM	ASR	MM	ASR	MM
en	0.192	-	0.268	0.360	0.275	0.320
de	0.237	0.257	0.177	-	0.251	0.259
nl	0.228	0.264	0.233	0.255	0.173	-

Source posteriors from *de* and *nl* acoustic models are mapped to *en* target posterior distributions using the corresponding mapping models ($\mathcal{M}_{de\_en}$ and $\mathcal{M}_{nl\_en}$). Normalised entropies of these mapped posteriors are plotted along with in-domain ASR normalised entropy (Figure 6.5b). Entropies for Only 100 time frames (from 100 to 200) are plotted for better visualisation. The dotted lines show the mean normalised entropy over all the frames. In a meticulous analysis of Figure 6.5a and Figure 6.5b, it can be seen that the entropy of *de* acoustic model is lesser than that of *nl* acoustic model for the given input speech signal. However, after mapping posteriors from these models to *en* posteriors, the mean entropy value is lower for the posteriors mapped from the *nl* model compared to the posteriors mapped from *de* ASR. It is in line with the previous results (Table 6.5) where *nl* mapping models have consistently shown lower KL-divergence and entropy values. The analysis here confirms that the mapping model from *nl* to *en* ($\mathcal{M}_{nl\_en}$) appears to learn better than ($\mathcal{M}_{de\_en}$) for the English target language. Though mapping models occasionally appear to be more confident than the in-domain ASR, the overall entropy of mapping models is a bit higher than that of the language-dependent ASR. The statistical significance of these entropy values could be understood in light of the analysis in Section 4.5.2.

In Figure 6.5c, normalised entropies for the selected frames of the same file are shown. The plot elements are the same as in Figure 6.5b except that the phonemes are also annotated along with their boundaries. The annotations are extracted from forced alignments to visualise the trends of entropies for different phonemes. As described earlier the Kaldi toolkit divides each phoneme into four phonemes, each phoneme annotation has two parts i.e. actual phoneme representation and their position in the word, both separated by an underscore symbol ('_').

Although the aforementioned particular example confirms the earlier entropy results of mapping models (Table 6.5), claims cannot be made based on a single example. So, the entropies are calculated for test sets of all the participating languages and tabulated in Table 6.8. It can be seen that the overall mean entropies of in-domain acoustic models are lower than those of out-of-domain acoustic models. Though all the acoustic models have higher confidence than the mapping models (MM), the entropy of mapping models is also dependent on those of acoustic models. For example, source acoustic models have higher entropy values for *en* data set and then those values for mapping models

TABLE 6.9: Bilingual ASR performance and mean KL-divergence versus % phonemes sharing of the target language with a source language (as the cross-lingual similarity measure)

Target	Source Languages (KL-Div/% phoneme sharing/% WER)		
Language	en	de	nl
en	0 / - / 43.84	1.56/ 70.58 / 46.35	**1.44** / 69.48 / 44.46
de	**1.00** / 72.96 /38.94	0 / - / 37.77	1.18 / 88.59 / 39.46
nl	**1.60** / 69.80 / 42.32	1.61/ 85.14 / 44.03	0 / - / 37.94

are further higher. Whereas the entropies of source acoustic models are lower for *nl* data set, hence those of mapping models are lower too. However, this trend also depends on the cross-lingual similarities and how much the two languages are amenable to learning mappings. The ratio of entropies of mapping models to acoustic models ranges from 1.032 to 1.342.

Trends in the performance of mapping models can also be dependent on training data sharing among the languages. Even if the two languages share a large number of phonemes, it does not guarantee that those phonemes or polyphonemes are seen by the training data of both languages. As an instance, *en* shares 130 and 110 positional phonemes with *de* and *nl* . As described earlier, only a subset of all possible polyphonemes is utilised in the spoken language, rather than the entirety. So, the *en* train set sees only 2368 and 1632 biphones from shared phonemes of *de* and *nl* respectively. A training set of *en* sees 1367 and 1303 biphones jointly with *de* and *nl* . So far, the numbers are higher for *de* source language which should mean that *de* is closer than *nl* for *en* target language which is opposite to the previous analysis. However, if the number of unseen biphones of *en* for *de* and *nl* are compared, *de* has not seen 1001 biphones of *en* during training of its ASR while this number is only 320 for *nl* . It implies that the training of *nl* ASR has seen a lot of *'en like'* data which might make it easy for *nl* to *en* mapping model (N_{nl_en}) to learn the mappings.

As a thorough analysis of the performance of mapping models and potential reasons for those trends has been shown, it is worth mentioning that there are some other factors which affect and make the objective performance of mapping models a bit worse. It has been described earlier that the output classes of the neural network of DNN-HMM are the clustered HMM states and each phoneme is further divided into four phonemes by Kaldi. In the implemented model, Kaldi uses a two-state model for each class and both states are not guaranteed to be clustered in a single class. The analysis reveals that there are various cases where the mapping model maps the input posteriors to the correct biphone technically but with the wrong positional biphone. For example, an *en* biphone class \ɪ_**B**,ŋ_**E**\is confused significantly with the class \ɪ_**I**,ŋ_**E**\and \ɪ_**B**,ŋ_**I**\. Although if the positioning of the class is ignored, \ɪ,ŋ\is still mapped to \ɪ,ŋ\but is considered an error because of the wrong positioning. Similar errors are encountered when even the positions are fine as well but HMM class is wrong.

The average cross-lingual KL divergence along with the WER of bilingual ASR systems is tabulated in Table 6.9. Diagonal entries are the monolingual ASR system. These results can be seen in comparison with cross-lingual phoneme sharing of Figure 4.4 (% phoneme sharing in Table 6.9). For example, Figure 4.4 infers that *de* shares 72.96% phonemes of *en* while *nl* shares only 69.80%. From phoneme sharing statistics, *de* is closer to *en* among *de* and *nl* , but the mean KL divergence is smaller for *nl* (Source lang.: *nl* and Target lang.: *en*). These numbers are more aligned with the %WER of bilingual ASR systems in Table 6.9. The same is true for *de* target language as well that the phoneme sharing shows a different picture than the entropy analysis. The overall analysis indicates that using the size of overlapping phonemes set across the languages to measure cross-lingual similarities is not a very accurate approach. Though it can inform to some degree, it does not inform much about how close the two languages are. That's why the multilingual setups of several languages result in the degradation of the performance of participating languages despite a significant phoneme overlap.

6.4.4 Basis for further experimentation

Though a thorough analysis has been done and described in the previous sections, it is quite hard to carry on an analysis on the phoneme level using a phone-trained acoustic model. The clustering of biphone classes also makes some analyses ambiguous and requires to analyse the restricted classes. Additionally, biphones make a huge number of possible output classes which makes it harder for a simple mapping model to learn the mappings among the different acoustic models. It is expected that the job of the mapping model would be easier if trained on the top of the mono-phone speech recognition systems. However, previous work on conventional HMM-based ASR systems has shown that polyphoneme (usually triphone) models perform better than monophones. In the next chapters, mapping models will be used for different data sets and tasks.

Before moving on to further experimentation, an analysis is carried out on the performance of Kaldi's LF-MMI-based speech recognition systems with monophones. All the monolingual ASR systems are trained with monophones keeping all the other constraints the same and the %WER is shown in the Table 6.10. It can be seen that the performance of the ASR system is marginally degraded overall in the case of monophones, the number of output classes is reduced significantly. For example, the number of classes falls to 232 in the case of monophones compared to 3344 classes of the biphone-trained model. Remember that these output classes are the clustered HMM states of positional

TABLE 6.10: Comparison of monolingual ASR performance trained with monophones and biphones

Tgt. Lang	biphone	monophone	% relative
en	36.73	36.74	0
de	27.26	28.05	-2.93
nl	32.34	32.71	-1.14

phonemes. Given a marginal degradation in the performance of the ASR system with a reduction of classes on the scale of 10 to 15 times, monophone-based models are trained for further experiments.

6.5 Summary

In line with our previous experiments, the work in this chapter has also shown that the languages which share more phonemes do not guarantee performance gain in multilingual setups. In this chapter, the analysis has observed that the phonemes with identical representations across languages are not guaranteed to be acoustically close. This study reveals that the behaviour of different languages for multilingual ASR is more complex than predicting from a cross-lingual phoneme-sharing perspective. In this chapter, cross-lingual acoustic-phonetic similarities are estimated by comparing the posterior distributions from the source and the target languages' acoustic models.

The source posteriors have been mapped to target language posteriors using a pre-trained mapping model for a given target language utterance. The mapped posteriors have been compared with the target posteriors (from the target language ASR) to estimate the cross-lingual similarities. The intuition is that the mapping model would be able to learn if there are any patterns in the source and the target posteriors. So, KL-divergence between mapped source and target posteriors has been employed as an estimate of the language similarities between a source and the target language. As the underlying assumption of mapping models is that some patterns might exist between posterior distributions from the target model and a source language ASR given a target language speech signal, the mapping model would not be able to learn well when a source model significantly confuses a phoneme with some other phoneme or multiple phonemes. The experiments have shown that only the entropy of the mapped posteriors can inform on source-target language similarities.

Mapping models have been trained on top of hybrid DNN-HMM acoustic models of three West Germanic languages i.e. English, German and Dutch. Bilingual acoustic models for all pairs have also been trained to observe the relationship of performance of the acoustic models with mutual phoneme sharing and the proposed mapping model's KL-divergence measure. The results have shown that the trends in the phoneme error rate from bilingual acoustic models are more aligned with the mapping models' entropy and KL-divergence of the mapped posteriors compared with the conventional measure of the number of overlapping phonemes.

Chapter 7

Acoustic model fusion for low-resource speech recognition

Multilingual speech recognition has gained attention as an effective way to address data scarcity in building ASR systems for low-resource languages (Abate et al., 2020, Karafiát et al., 2016, Tachbelie et al., 2020a). Efforts have been made to leverage cross-lingual and multilingual resources to improve performance (Grézl et al., 2014, Tong et al., 2018). End-to-end modelling is now preferred over conventional hybrid systems due to its lack of lexicon requirements. However, hybrid DNN-HMMs still perform better in limited data scenarios (Povey et al., 2016). In this chapter, the proposed mapping model approach of Chapter 5 is leveraged for hybrid DNN-HMM acoustic model fusion in a multilingual setup to improve the ASR performance for low-resource languages. Mapped posterior distributions from different source monolingual acoustic models for a target language input speech signal, are fused. Baseline and MESD mapping models are trained for each target language to compare their performance. Since training of the mapping models requires very limited data as compared to the ASR training for the target language, the objective is to improve the performance of target language ASR by exploiting the proposed mapping modelling approach.

In the following sections, the previous work done to make use of cross-lingual and multilingual resources to improve low-resource speech recognition is revisited first. In Section 7.2, the proposed approach discusses how mapping models can be used to improve speech recognition in limited data scenarios. Experimental details are described in Section 7.3 and the results are presented and discussed in Section 7.4.

This chapter is based on my publication in Interspeech 2022;
M. U. Farooq, D. A. H. Narayana, T. Hain, *"Non-Linear Pairwise Language Mappings for Low-Resource Multilingual Acoustic Model Fusion"*, Interspeech 2022.

7.1 Background

Over the past decade, various approaches have been used to exploit multilingual resources to compensate for the data scarcity problem in training automatic speech recognition systems for low-resource languages (Abate et al., 2020, Besacier et al., 2014, Imseng et al., 2014, Karafiát et al., 2016, Tachbelie et al., 2020a, Vu and Schultz, 2013). As discussed earlier in Chapter 1, one way is to train a unified acoustic model by mixing the training data of all the languages (Hou et al., 2020, Huang et al., 2013, Pratap et al., 2020a, Tong et al., 2018). A multilingual model then can be used directly for speech recognition of a low-resource language (Hou et al., 2020, Pratap et al., 2020a), or transferred to a specific language through a further language-specific training stage (Huang et al., 2013, Tong et al., 2018). Another approach is to use these DNN-based multilingual acoustic models to extract features to train a monolingual model (Ghoshal et al., 2013, Grézl et al., 2014, Veselý et al., 2012). For feature extraction, a multilingual DNN acoustic model is trained first on the mixed training data of multiple languages. Usually, a bottleneck layer is used before the final or a few final layers. Once the model is trained, the training data of a language is decoded through this model, and the embeddings (regarded as features) from the bottleneck layer are used as features to train a language-specific model.

Though e2e multilingual speech recognition systems are preferred over conventional ASR systems to avoid demanding manual lexicon creations (Graves et al., 2013b), DNN-HMMs still outperform e2e models in limited data scenarios such as low-resource languages (Kürzinger et al., 2020). The problem of creating lexicons for HMM-based ASR systems has been alleviated with the advancement of G2P and text-to-IPA transliteration approaches such as Phonetisaurus (Novak et al., 2012), Epitran (Mortensen et al., 2018) and open source LanguageNet G2P models (Hasegawa-Johnson et al., 2020). Previous work on e2e multilingual speech recognition systems shows that multilingual speech recognition systems do not guarantee the reduction in word error rate for target languages (Conneau et al., 2021, Hou et al., 2020, Pratap et al., 2020a) when compared with their monolingual counterparts. Especially in the case of multilingual systems consisting of high-resource languages, WER is not reduced for all languages (Conneau et al., 2021, Hou et al., 2020, Pratap et al., 2020a).

As discussed in Chapter 4, recent efforts to interpret the learning of multilingual speech recognition systems (Feng et al., 2021, Żelasko et al., 2020) observe that the error rate of an overlapped phoneme is not reduced in multilingual ASR with the growing number of its sharing languages. It implies that the number of shared phonemes is not a reliable metric to measure language similarities. Even balanced language data sampling can cause degradation or improvement due to underlying phonetic unbalancing (as shown in Section 4.4.2). It demands a very controlled language mixing for a target language ASR.

In this chapter, a proposed novel technique is presented to fuse outputs from different

monolingual models given a target language input speech signal. Various previous studies on monolingual speech recognition have fused outputs from different models for acoustic (Abdelaziz, 2018, Mallidi and Hermansky, 2016, Rebai et al., 2017, Wong et al., 2020) and language models (McDermott et al., 2019, Shan et al., 2019). In (Abdelaziz, 2018), several model fusion approaches have been discussed. Wong et al. (Wong et al., 2020) have combined the hypothesis from hybrid DNN-HMM and e2e ASR systems which performed better than all of the individual models. Similarly, hypotheses from acoustic and vision-based ASR systems have been fused by Abdelaziz et al. (Abdelaziz, 2018) for multimodal speech recognition.

However, monolingual models have never been fused in a multilingual setup so far. An obvious limitation is that different languages have different phoneme or character sets and the output classes of the monolingual model of each language are different from the others. In the case of e2e systems, output classes are characters or sub-word tokens which can be very diverse across the languages. In the case of phoneme-based hybrid DNN-HMM ASR systems, a different phonetic decision tree of each monolingual model causes different clusters and thus diverse output classes of DNNs. It makes it harder to use previous model fusion techniques to fuse monolingual outputs in a multilingual setup.

In this work, the proposed mapping models (discussed in Chapter 5) are leveraged to fuse the output of monolingual models of different languages for a given target language input speech utterance. A mapping model is trained for each <*source, target*> language pair to map posteriors from a source language acoustic model to the posteriors of the target language acoustic model. A given input speech signal is decoded through all monolingual acoustic models (of source and target language ASR systems). Then posteriors from all the source languages are mapped to the target language posteriors space using the specific source-target mapping model. These mapped posteriors are then fused in multilingual and cross-lingual fashion for phoneme recognition of the target language. In multilingual model fusion, the target posteriors are also included in the model fusion. However, in cross-lingual model fusion, only the mapped posteriors from source acoustic models are fused for decoding the target language data. The proposed approach is helpful especially for low-resource languages because;

- the *mapping models* can be trained with very limited amounts of data since a few hours can provide sufficient examples for phonetic-level training.
- controlled fusion of posteriors based on language similarity will allow controlling contribution of different source languages.

The mapped posteriors from the monolingual acoustic models are fused in a multilingual setup which not only outperforms the baseline multilingual ASR but also the monolingual ASR systems.

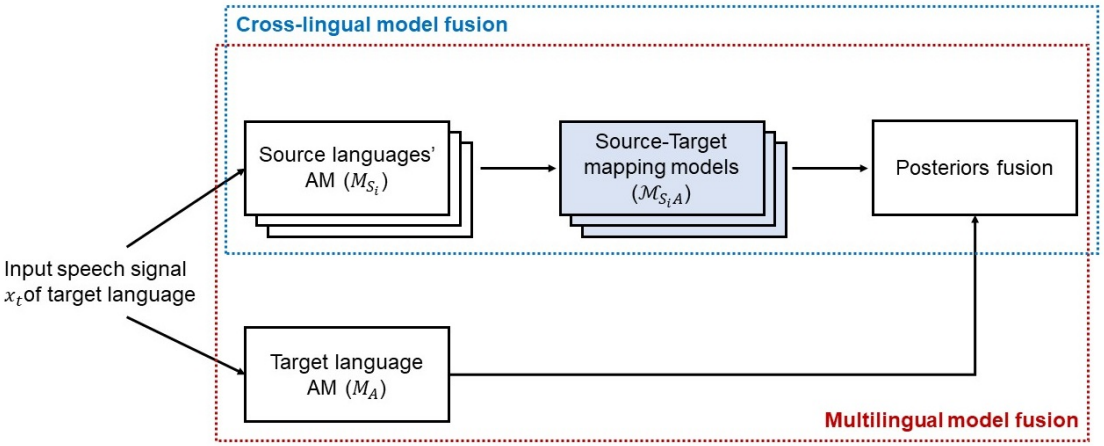


FIGURE 7.1: Proposed system architecture for model fusion. In multilingual model fusion, mapped posteriors from various source monolingual models are fused along with target posteriors. However, target posteriors are not used in cross-lingual model fusion.

7.2 Acoustic model fusion

In this section, the proposed approach to fuse outputs from different monolingual acoustic models for target language speech recognition is discussed in detail. The approach can be extended for any type of speech recognition system and output tokens such as phonemes, graphemes, or BPE tokens-based end-to-end or hybrid DNN-HMM models. In this chapter, the concept is proved for phoneme-based hybrid DNN-HMM models since hybrid models still outperform e2e ASR systems in limited training data scenarios (Povey et al., 2016). The approach is extended for end-to-end ASR systems later (in Chapter 8) but the scope of this chapter is limited to phoneme-based hybrid DNN-HMM-based speech recognition systems.

As described in previous chapters, a deep neural network is trained to produce the emission probabilities of tied states of HMM models in hybrid speech recognition systems. Theoretically, the total number of tied states for a language with Q number of phonemes and S number of states per HMM model is given by $Q^n \times S$, where n is the context width. However, many polyphonemes never occur in a language speech data and many are quite similar to the others. The total number of states is reduced by clustering many polyphonemes together. Each language yields a different phonetic decision tree in its monolingual ASR. Thus the number of tied states differs for each language and the output classes of DNN models of each monolingual ASR are different. So, the posterior distributions from DNNs of two different monolingual acoustic models are not directly comparable and thus not fusible as well.

Let M_A and M_{S_i} be the monolingual acoustic models of target and source languages respectively. A baseline mapping model N_{S_iA} is trained to translate posteriors $P_{S_i} \in \mathbb{R}^{d_{S_i}}$ from M_{S_i} to the posteriors $P_{S_iA} \in \mathbb{R}^{d_A}$. An underlying assumption is that a mapping model can learn some language-related relationships between posterior distributions of

source and target acoustic models. For example, a mapping model could learn the phonemes of the target language which are more amenable to cross-lingual transfer than the others. Furthermore, a few hours of speech data can give millions of examples that provide sufficient training data for the mapping model.

Let $X = \{x_1, x_2, \ldots, x_T\}$ be a set of observations of the target language, for which posterior distributions ($P^Z = \{p_1^Z, p_2^Z, \ldots, p_T^Z\}$ where $Z \in \{A, S_i\}$) are attained from all monolingual acoustic models. A mapping model is trained using cross-entropy loss to map posteriors from source acoustic models (P^{S_i}) to the target language posteriors (P^{S_iA}). The loss function for a batch is given as;

$$\mathcal{L}_{S_iA}(\theta) = -\sum_{t=1}^{T}\sum_{k=1}^{d_A} p_{t,k}^{A} \cdot \log p_{t,k}^{S_iA} \tag{7.1}$$

where T is the total number of training examples in a batch to train a mapping model N_{S_iA} which maps posteriors from i^{th} source language to the target language.

Posterior distributions from target AM and mapped distributions from source acoustic models are fused for phoneme recognition of a target language. For a given observation at time t, the final posterior vector is given as;

$$p_t^F = w_A \cdot p_t^A + \sum_{i=1}^{K} w_{S_iA} \cdot p_t^{S_iA} \tag{7.2}$$

where w are the scalar weights assigned to each posterior vector such that $\sum w = 1$ and K is the number of source languages. The proposed system architecture is shown in Figure 7.1.

For the experimentation in this chapter, model fusion is done in multilingual and cross-lingual settings. In ***multilingual model fusion***, target posteriors are also used in fusion along with mapped posteriors from various source languages acoustic models. As the posteriors from the target language acoustic model are also fused with mapped posteriors, multilingual model fusion is still dependent on the language-specific ASR system. So in the ***cross-lingual model fusion***, only the mapped posteriors from source language acoustic models are fused (the term $w_A \cdot p_t^A$ is omitted from Equation 7.2). The cross-lingual setting avoids using the target-language acoustic model which is helpful for low-resource languages. During model fusion, different methods are experimented with to assign weights to the posterior distributions. As shown and discussed in Section 6.4.3, the entropy of a <*source, target*> mapping model indicates similarities between a source and a target language. The same similarity measure is used along with mapping model accuracy to assign the weights. Later, rather than assigning manually, weights are also learnt in mapping model training.

7.3 Experimental setup

7.3.1 Data set

The experimentation is done on various languages from two different multilingual data sets i.e. MLS and Babel. The idea of experimenting with two different data sets is to analyse the performance of the proposed fusion approach for data sets of diverse domains. As the MLS data set has been recorded in a very clean environment and read speech, the recording quality is far better than the conversational Babel speech which has been recorded in an open environment. A significant proportion of Babel data sets has been recorded with noisy backgrounds and a lot of non-speech segments which pose a lot of challenges to build a considerably good speech recognition system. Hence the performance of the mapping models is also expected to be affected by that.

The proposed model fusion technique is applied for the three languages of MLS i.e. English(*en*), German(*de*) and Dutch(*nl*). The same amounts are used as described in Section 6.3.1. Monolingual speech recognition systems are trained on 30-hour train sets of each language and evaluated on 2 hours of test set. The train set is further divided into 29 hours of train and 1 hour of dev set for mapping model training. The evaluation set for the mapping model remains the same as for the speech recognition system.

Data sets from the Babel data, used for the experimentation here, have been explained in detail in Section 5.5.1. Full language packs of Tamil (*tam*), Telugu (*tel*), Cebuano (*ceb*) and Javanese (*jav*) are used for baseline ASR training and evaluation. Since the eval data of Babel is not publicly available, train and dev sets of Babel data sets are used as train and eval sets respectively for the experiments. The details of the data sets are tabulated in Table 5.1. Since most of the Babel data consists of Conversational Telephone Speech (CTS), it is quite challenging to build a speech recognition system being conversational and telephonic speech. Recording conditions are also very challenging such as noisy background environments, different phone sets (recording equipment) and cellular operators. A limited amount of scripted read speech is also included in each language pack.

Full amounts are used for the training of baseline monolingual and multilingual speech recognition systems. Multilingual ASR is trained by mixing data from all the languages. However, for training the mapping models, a subset of 30 hours is chosen from each language pack. This data is further divided randomly into 29 and 1-hour portions as train and dev set to train the mapping models. Since the mapping models are trained on a phonetic level, 30 hours provide millions of examples for the sufficient training of these models. The examples, used for building the mapping models, are given in Table 5.2.

7.3.2 Baseline ASRs

Baseline monolingual and multilingual acoustic models are hybrid DNN-HMM models. 40 MFCCs are extracted for each frame of the speech signals using a window size of

25 ms and a shift of 10 ms. These features are then fed to DNN which consists of 12 factorised TDNN (TDNN-F) layers (Povey et al., 2018). Each TDNN-F hidden layer is of dimension 1024, factorised with a linear *'bottleneck'* dimension of 128. Model architecture and configuration are the same for MLS and Babel experiments in principle, however, there might be slight differences. For instance, the handcrafted pronunciation dictionaries of Babel data sets come with the corpus while open-source lexicon is used for MLS data sets along with G2P models for transcription of OOV words. Further details of MLS and Babel models can be found in Section 4.4.2 and Section 5.6.1 respectively. The acoustic models are trained using MMI criterion with lattice-free approach (Povey et al., 2016). As discussed in the last chapter the output of the neural network is the posteriors of the clustered biphone classes in the case of the MLS data set, HMM states of monophones are clustered and are the output classes of the DNN for the Babel data set. The clustering in each monolingual ASR training is different and thus the outputs from different acoustic models against an identical speech signal are not directly comparable. The ASR systems are built using the Kaldi toolkit (Povey et al., 2011).

7.3.3 Mapping models

Experiments are done with both, baseline and MESD mapping models. A baseline mapping model (discussed in detail in Section 5.2) is a feed-forward regression network with four fully connected layers. Baseline mapping models are trained for each source-target language model and each model is of about 1.25 million parameters. So, for each language, $n-1$ mapping models are trained where n is the number of total languages. On the other hand, a MESD mapping model is a multi-encoder and single-decoder mapping model and one model is trained for each participating language. Technical details of MESD mapping models have been discussed in Section 5.3.3 and Section 5.4.1.

7.4 Results and discussion

7.4.1 Baseline ASR systems

Monolingual baseline systems (*mono*) are the language-dependent acoustic and pronunciation models which are trained on a language-specific data set. The train sets of all the languages are then mixed to train the multilingual models (*multi*). The performance of the baseline systems (in terms of PER) is given in Table 7.1 (above the dashed line). The results show that the error for all the languages is increased in the baseline multilingual setup. The mean phoneme error rate rises to 44.1% from 41.44% in the case of Babel data whereas it is increased from 25.38% to 31.28% for the MLS data set. Though PER is increased for most of the Babel languages, a mean relative reduction of 0.8% is from *tam* language. A massive relative increase of 25.19% in PER of the MLS data set is seen, but a significant contribution is from *nl* language. The investigation reveals that the speakers' distribution in the training set of *nl* language in the MLS data set is highly skewed. About 79% of data have been recorded by one speaker and the other 39 speakers contributed for the rest of the 21% of data. So, the ability of the multilingual model

TABLE 7.1: Speech recognition performance in terms of % phoneme error rate for Babel and MLS data sets. *mono*: Language-dependent ASR system, *multi*: Language-independent ASR system, *cross-mf*: proposed cross-lingual model fusion, *multi-mf*: proposed multilingual model fusion

	Babel					MLS			
	tam	*tel*	*ceb*	*jav*	*avg*	*en*	*de*	*nl*	*avg*
mono	43.96	43.66	36.67	41.6	41.44	28.10	26.86	21.40	25.38
multi	43.67	46.36	41.02	45.54	44.10	31.06	28.35	36.16	31.28
cross-mf	55.47	52.76	43.04	47.79	49.72	37.90	32.27	27.15	32.44
multi-mf	**41.96**	**42.05**	**35.54**	**38.87**	**39.55**	**23.57**	**25.35**	**20.23**	**23.05**

to learn speaker variability for *nl* weakens when speakers from *en* and *de* are mixed in the training data. Even if the performance for *nl* language is ignored, a degradation of 6.68% is still there in the case of MLS multilingual ASR performance.

Error rates for the Babel languages are quite higher than the MLS data set even though ASR systems of Babel languages are trained with more training data. This is due to the reason that the Babel data set is quite challenging and the error rates in literature are also very high (up to 80% word error rate) (Hou et al., 2020).

7.4.2 Acoustic model fusion

As described in Section 7.2, a multilingual acoustic model is imitated by fusing the target and the mapped posteriors. The performance of model fusion is measured for both data sets i.e. MLS and Babel sets. For each target language, the input speech utterance is decoded through all the monolingual ASR systems (of source languages as well as of the target language). Source posterior distributions are mapped to the target posteriors' space using respective mapping models. For multilingual model fusion, mapped posteriors and target posteriors are fused in a weighted sum before decoding. In cross-lingual model fusion, target posteriors are dropped and only the mapped posteriors are fused.

7.4.3 Multilingual model fusion

Multilingual model fusion is the linear weighted sum of all of the posterior distributions of a target language i.e. target posteriors and mapped posteriors from all the source languages. The results of multilingual model fusion *multi-mf* are shown in the last row of the Table 7.1. The setup of *multi-mf* is similar to multilingual ASR system *multi* (since it is imitating a multilingual setup) and directly comparable with that but it outperforms monolingual ASR (*mono*) systems as well for all the languages. For Babel data sets, multilingual model fusion gives a mean absolute improvement of 1.9% and 1.55% compared with monolingual and multilingual models respectively. In the case of MLS data sets, the mean absolute improvement of 2.4% is achieved when compared with monolingual models which rise to 8.81% if compared with multilingual ASR system. However, it is due to a very high multilingual MLS baseline as discussed in the last

TABLE 7.2: Performance of model fusion in cross-lingual setting. 'Y' represents the source languages being fused

Target Language	Fused languages *tam*	*tel*	*ceb*	*jav*	% PER
tam	N	Y	Y	Y	**55.47**
	N	Y	N	N	55.65
	N	N	Y	N	57.69
	N	N	N	Y	57.33
tel	Y	N	Y	Y	52.76
	Y	N	N	N	**52.37**
	N	N	Y	N	55.68
	N	N	N	Y	53.58
ceb	Y	Y	N	Y	**43.04**
	Y	N	N	N	45.94
	N	Y	N	N	45.28
	N	N	N	Y	43.91
jav	Y	Y	Y	N	**47.79**
	Y	N	N	N	48.40
	N	Y	N	N	48.90
	N	N	Y	N	48.25

section.

7.4.4 Cross-lingual model fusion

The results of cross-lingual model fusion are given in the first row of Table 7.1 after the dashed line. Results of cross-lingual model fusion show that without using the language-dependent ASR, a comparable phoneme error rate for a target language can be achieved. For cross-lingual fusion, mapped posteriors from only the source languages AMs are fused. On average, the performance of cross-lingual model fusion is degraded by 8.62% and 0.58% from baseline multilingual models for Babel and MLS data sets respectively.

The computation cost for fusing a large number of languages intrigues us to minimise the number of fusing languages. For a given target language, further experiments are carried out using the mapped posteriors from only one source language at a time. For the Babel data set, Table 7.2 shows that nearly similar results as *cross-mf* can be achieved using mapped posteriors from the closest source language AM only. In the case of the *tel* language, mapped posteriors from a single AM model of *tam* perform even better than the cross-lingual model fusion. The first row for each language is the same as *cross-mf* of Table 7.1 and the following rows are the cross-lingual mapped posteriors from only one of the source language AM.

The results shown in this and the last section are produced by using baseline mapping models and manually fine-tuned weights for model fusion. As it has been shown earlier that the MESD mapping models perform better than baseline mapping models, both

TABLE 7.3: Model fusion speech recognition error (in %PER) after improvements in acoustic model fusion. *Baseline MM*: results from the Table 7.1, *MESD MM*: using MESD mapping models for mapping posteriors before fusion, *+ learnt weights*: Using learnt weights for fusion

	multi-mf				*cross-mf*			
	tam	*tel*	*ceb*	*jav*	*tam*	*tel*	*ceb*	*jav*
Baseline MM	41.96	42.05	35.54	38.87	55.47	52.76	43.04	47.79
MESD MM	36.47	35.39	34.79	38.54	46.89	43.16	42.14	47.51
+ learnt weights	36.27	34.63	34.72	38.16	46.64	42.23	42.05	47.04

multi-mf and *cross-mf* experiments are done using MESD mapping models on top of speech recognition systems. Furthermore, manually tuned weights for model fusion might lead to a sub-optimal solution. So, rather than manually assigning, weights are learnt as a single layer on top of the mapping models. The input for this layer is the mapped posteriors from all source languages and the target is the ground-truth posteriors. Both of these modifications help reduce error rates further. Phoneme error rates of multilingual and cross-lingual model fusions for the Babel data set are tabulated in Table 7.3.

For all the languages, the improved mappings yield a reduction in error rate over the baseline multilingual and cross-lingual model fusion (*cross-mf*) but the improvement margin is dependent on mapping model accuracy. Languages with better mapping models show a relative reduction of up to 20% but it is marginal for others. As evident from Table 7.3, fusion using the learnt weights slightly reduces the error rate further.

7.5 Summary

A novel model fusion technique has been explained in this chapter to improve low-resource speech recognition. Model fusion approaches have shown promising results for several different downstream tasks including multimodal speech recognition. However, acoustic models of diverse languages cannot be fused because of being trained on different data and thus different output tokens. The work described in this chapter proposed an approach to fuse the knowledge from diverse monolingual acoustic models to improve the speech recognition of low-resource languages. A target language speech signal is decoded through multiple monolingual acoustic models (of diverse languages). The output posteriors from source acoustic models are mapped to the target posteriors using pre-trained mapping models (described in Chapter 5). The mapped posteriors have been fused in two ways i.e. multilingual and cross-lingual model fusion. In the multilingual model fusion approach, mapped posteriors are fused along with the target posteriors (from the target acoustic model) in a weighted sum. However, in cross-lingual model fusion, only the mapped posteriors have been fused without using target posteriors to avoid the usage of the target acoustic model during fusion. Since mapping models need limited amounts for training, the approach is helpful for low-resource languages.

The baseline experiments in this chapter have been done for four low-resource languages from the IARPA Babel data set i.e. Tamil, Telugu, Cebuano and Javanese and the three West Germanic languages from the MLS corpus i.e. English, German and Dutch. Using the baseline mapping models, the results have shown that multilingual model fusion yields an absolute average gain of 4.55% and 8.23% over the baseline multilingual acoustic model for Babel and MLS data sets respectively. Though cross-lingual model fusion uses posteriors from only source acoustic models, the error rate is still comparable and degrades by 5.62% and 1.14% for Babel and MLS data sets respectively. The experiments have shown that nearly similar results to the cross-lingual model fusion can be achieved by using posteriors from only one (closer) language. It would decrease the computation cost at a marginal expense of performance degradation.

Further experiments have been done to employ MESD mapping models and learning fusion weights in training. It has reduced the error by an absolute of 3.66% for multilingual and 5.27% for cross-lingual model fusion of Babel languages which drops the overall error rate even below those from their monolingual acoustic models. However, the improvements are fairly dependent on mapping models' accuracies. Languages with better mapping models show a relative reduction of up to 20% but it is marginal for others.

Chapter 8

Mapping models for e2e ASR systems

In previous chapters, the concept of mapping models and their use for phoneme-based hybrid DNN-HMM models have been demonstrated. However, the phoneme set of every language is a subset of a finite universal phonemes set of around 3800 phonemes (Moran and McCloy, 2019) which is a very small set compared to the set of characters of all the languages. For example, the size of the union set of phonemes of the four Babel languages (Cebuano, Javanese, Tamil and Telugu) in previous experiments is 56 while the number of total characters rises to 168. The small set of phonemes increases the chances of various diverse languages to share the same phonemes. Limited tokens set and overlap among the languages are expected to make the cross-lingual mapping learning task less challenging. State-of-the-art speech recognition systems are trained in end-to-end fashion nowadays (Graves et al., 2013b, Hou et al., 2020, Pratap et al., 2020a). For end-to-end ASR models, output tokens are usually characters or sub-word units which are very diverse across the languages (Hou et al., 2020, Pratap et al., 2020a). Even several languages which are acoustically similar or belong to the same language families are written in different scripts such as Turkish and Kazakh (Turkic), Urdu and Hindi (Indo-Aryan), and Greek and Armenian (Indo-European). In this chapter, the mapping model approach is extended for the e2e speech recognition models to analyse whether the assumption of learning cross-lingual similarities holds for end-to-end ASR models as well.

In this chapter, a novel data augmentation technique is also proposed to improve low-resource speech recognition using the mapping model approach for e2e ASR models. A lot of work has been done on audio data augmentation by altering original audio signals

This chapter is based on my publication in Interspeech 2023;
M. U. Farooq, T. Hain, *"Learning Cross-lingual Mappings for Data Augmentation to Improve Low-Resource Speech Recognition"*, Interspeech 2023.

(Jaitly and Hinton, 2013, Ko et al., 2015, Park et al., 2019). However, very limited work has been done for audio data augmentation in terms of altering the text rather than the audio signal (Datta et al., 2020, Thomas et al., 2020).

This chapter is organised as follows. In Section 8.1, literature is revisited to highlight the importance of e2e ASR models as a pretext for extending the mapping approach for e2e ASR models. Recent approaches to improve e2e low-resource speech recognition models and data augmentation are also discussed. Section 8.2 describes the details of mapping models for e2e and challenges in extending the mapping model approach for end-to-end ASR models. The mapping model approach is exploited for data augmentation to improve the low-resource speech recognition which is discussed in Section 8.3. The experimental setup is described in Section 8.4 and experimental results are presented and discussed in Section 8.5.

8.1 Background

In hybrid DNN-HMM systems, neural networks are employed as a part of an HMM-based acoustic model. Output from the DNN model is formulated as the emission probabilities of the HMM states. Training of such systems usually needs frame-level alignments which are achieved using a pre-trained (usually GMM-HMM) acoustic model. Additionally, the requirement of a lexicon is also a challenge in training a hybrid DNN-HMM acoustic model. With the advancement in building end-to-end models for different tasks in vision and language processing, efforts have been made to build an e2e automatic speech recognition system. Raw waveform or spectrogram is usually fed as input to such models and the output is the orthographic transcriptions. Graves et al. (Graves and Jaitly, 2014) have employed an RNN model with a speech spectrogram as input which outperformed the baseline DNN-HMM models without using an explicit language model. End-to-end speech recognition systems are considered stronger speech recognition systems because of the implicitly trained language model. Thus, e2e speech recognition systems have more language information than HMM-based speech recognition systems. Since then, numerous works have been done to improve e2e speech recognition models (Chan et al., 2016, Watanabe et al., 2017) because of the several following reasons;

- Contrary to DNN-HMM models, no prior alignments are needed to start the training.
- No (phonemic) pronunciation dictionary is required which is usually created manually in case of HMM models training.
- The Acoustic model has implicitly learnt language information and works well even without an explicit language model.
- Usually a raw waveform or spectrogram is used as input features which saves preprocessing time.

In recent years, a lot of efforts have been made to improve e2e speech recognition models and use them for multilingual speech recognition (Hou et al., 2020, Hsu et al., 2020, Pratap et al., 2020a). One of the motivations is to build a unified model for a large number of languages which have been discussed in Section 3.2. Several works have been done for low-resource languages which are briefly described in the following paragraphs.

Training of end-to-end models usually requires a lot more data than needed for training of an HMM model. However as discussed earlier, a lot of languages do not have enough data to build such models. This has drawn the attention of the researchers to exploit the cross-lingual resources in efforts to improve the e2e speech recognition of low-resource languages (Ragni et al., 2014, Thomas et al., 2020).

With the emergence of sequence modelling approaches, encoder-decoder architectures have also been applied to improve low-resource speech recognition in a multilingual setup (Toshniwal et al., 2018). Zhou et al. have shown that a multilingual transformer model improves ASR performance for participating languages if the corresponding language tags have been appended at the start of the transcriptions (Zhou et al., 2018). Furthermore, a lot of massively multilingual speech recognition models have been developed recently which are also based on end-to-end modelling approaches (Hou et al., 2020, Li et al., 2022a, Pratap et al., 2020a).

Though the earlier proposed approach of exploiting cross-lingual resources for low-resource speech recognition has been proven to work for phoneme-based hybrid DNN-HMM models, the mapping model approach is extended for end-to-end speech recognition models. However, training mapping model on top of e2e ASR models is challenging for several reasons;

- Different writing scripts across the languages (thus different character sets) make very diverse representations across the languages. It makes it challenging for a model to learn cross-lingual projections.
- Some languages have writing scripts that are very rich with a character set of 70-80 alphabets though they have fewer phonemes. A larger number of output tokens further make the task hard for mapping models.

Given the aforementioned challenges, a question arises if the mapping model approach is applicable on the top of e2e ASR models as well. Though each language has a very diverse character set, the underlying acoustic units are still the same so a model is expected to learn projections among the cross-lingual characters or sub-word units. With that intuition, the previously used mapping model approach is extended for e2e speech recognition models.

As described earlier, mapping models for e2e speech recognition are further employed for data augmentation. The work done for data augmentation so far is briefly described in the following section.

8.1.1 Data augmentation for low-resource ASR

End-to-end acoustic modelling techniques require a lot of training data for reliable parameters estimation. However, more than half of the world's population speaks only 23 languages out of more than 7000 languages being spoken across the globe (Ethnologue). Thus only a few languages have sufficient data resources, and a lot of languages are still under-resourced to build an ASR system. For such languages, multilingual speech recognition systems have stolen the limelight over the past decade (Abate et al., 2020, Besacier et al., 2014, Imseng et al., 2014, Karafiát et al., 2016, Tachbelie et al., 2020a, Vu and Schultz, 2013) which have been used for feature extraction (Ghoshal et al., 2013, Grézl et al., 2014, Veselý et al., 2012) or directly for transfer learning (Huang et al., 2013, Tong et al., 2018).

Data augmentation is another approach to increase the training data of a low-resource language. Commonly used data augmentation techniques include extending training data by making perturbed copies either by adding noise (Gales et al., 2009, Hannun et al., 2014), varying speed and tempo of original speech (Ko et al., 2015), Vocal Tract Length Perturbation (VTLP) (Cui et al., 2015, Jaitly and Hinton, 2013), SpecAugment (Park et al., 2019) and combinations of these methods (Ragni et al., 2014). All these techniques are based on audio data augmentation.

Jaitly and Hinton (Jaitly and Hinton, 2013) laid the foundation of audio data augmentation for speech recognition systems. In their proposed VTLP technique, perturbed copies of original audio data have been produced by mapping a frequency f in an audio signal to a new frequency f' using a randomly generated frequency warping factor. The range of the warping factor has been limited from $[0.9, 1.1]$ to avoid unrealistic distortions. Ko et al. (Ko et al., 2015) proposed a similar approach but in the time domain. Audio data is augmented by making three copies of the audio data with perturbed speeds of 0.9, 1.0 and 1.1 of the original audio speed. LVCSR trained for four different tasks have shown a relative improvement of more than 4% compared to VTLP (Ko et al., 2015). Ragni *et al.* (Ragni et al., 2014) combined two data augmentation techniques that are, unsupervised data and VTLP for training of low-resource ASR. Unsupervised data augmentation (Evermann and Woodland, 2000) is a conventional approach where unlabelled audio data is decoded through an existing baseline ASR. A certain threshold is set for the confidence score of the model and the utterances with a confidence score below that threshold are rejected while the rest of the utterances are augmented for retraining of the ASR.

In (Hannun et al., 2014), noisy data is synthesised to train a noise-robust speech recognition system. Multiple short noises are collected from various videos. These collected noisy clips are superimposed with a clean speech utterance to generate a noisy speech utterance (i.e. $\hat{x}^{(i)} = x^{(i)} + \xi_1^{(i)} + \xi_2^{(i)} + \cdots$ where x is a clean speech utterance and ξ represents a segment of noise). Multiple random noises are superimposed rather than adding a single noise to refrain the model from learning a single noise and subtracting

that from noisy speech. Any noise with an average frequency range significantly different from real-time noises has been discarded. This synthesised noisy data is used as augmented data to train an ASR which is more robust to noise than a system just trained on clean speech.

SpecAugment (Park et al., 2019) has also been proposed for data augmentation that is directly applied to speech spectrogram rather than raw audio or frequency. Time warping, time masking and frequency masking have been applied to a spectrogram to generate data for augmentation. The proposed method proves to outperform all other augmentation techniques for transformer-based LVCSR and achieved a word error rate of 2.5% and 5.8% on test-clean and test-other data sets of LibriSpeech.

In the recent past, a few studies have been done to augment data by processing text rather than speech (Datta et al., 2020, Emond et al., 2018, Thomas et al., 2020). Transcripts from different languages have been transliterated to Latin script to train a multilingual system (Datta et al., 2020). Emond et al. (Emond et al., 2018) and Datta et al. (Datta et al., 2020) transliterated code-switched and multilingual transcriptions to a uniform writing script. The transliteration has been done using a composition of Weighted Finite State Transducers (WFST). In $I \circ P \circ O$ composed WFST of (Emond et al., 2018), I maps an input unicode symbol to a paired language model symbol, P is a bi-gram language model that maps input in one writing script to the other and O maps the pair language model symbol to the target language symbol. However, these transliteration techniques require paired data (a word in the original script and its transliteration in Latin) for each language.

Thomas *et al.* (Thomas et al., 2020) have proposed to transliterate source language data to the target language without using parallel data. A multilingual ASR model is trained where all the layers are shared except the language-dependent output layers as shown in Figure 8.1. During training, gradients are calculated and updated for shared layers and only specific output layers for a batch of language L_i. During decoding, however, transcriptions from all the output layers are extracted for the data of language L_i. Though when i^{th} language-specific output layer outputs the transcription of the input audio data, the rest of the output layers generate pseudo transcriptions in corresponding languages. As those layers are not trained with this objective, the generated output is not expected to be an exact transliteration. The pseudo transcription from j^{th} language output layer along with the input audio of i^{th} language is used for data augmentation to train the ASR of j^{th} language. This is an interesting idea but an out-of-domain ASR has no knowledge of input language and thus is not expected to generate a good transliteration.

Motivated by the idea of Thomas et al. (Thomas et al., 2020) and the fact that the output layers of other languages are not trained with the objective they are being used for, the work on e2e cross-lingual speech recognition is extended for data augmentation employing the mapping models. In the previous chapters, mapping models have been

discussed as a technique to learn cross-lingual acoustic-phonetic similarities on phoneme level (Chapter 6), for multilingual and cross-lingual acoustic model fusion (Chapter 7). For data augmentation in this work, the ASR systems of source languages followed by a source-target mapping model for each source-target pair are used to transliterate source data into the target language script. Though both the components are trained on task-specific data and are expected to generate better output labels, transliteration of a source language audio data into the target language is still unintelligible, especially for unrelated languages and thus called *ciphered* data. So, the key contribution of this work is to generate ciphered text for a target language data augmentation using source languages ASR and <*source-target*> mapping models.

8.2 Mapping models

As discussed in Chapter 5, the training and functionality of mapping models are not dependent on ASR architecture or output tokens i.e. phonemes or characters. So, the mapping models are trained on the top of e2e ASR models exactly in a similar way as they have been trained for phoneme-based hybrid DNN-HMM acoustic models. For the experimentation in this chapter, the latest MESD mapping models are trained for each target language (details in Section 5.3.3). As either the output posterior distribution

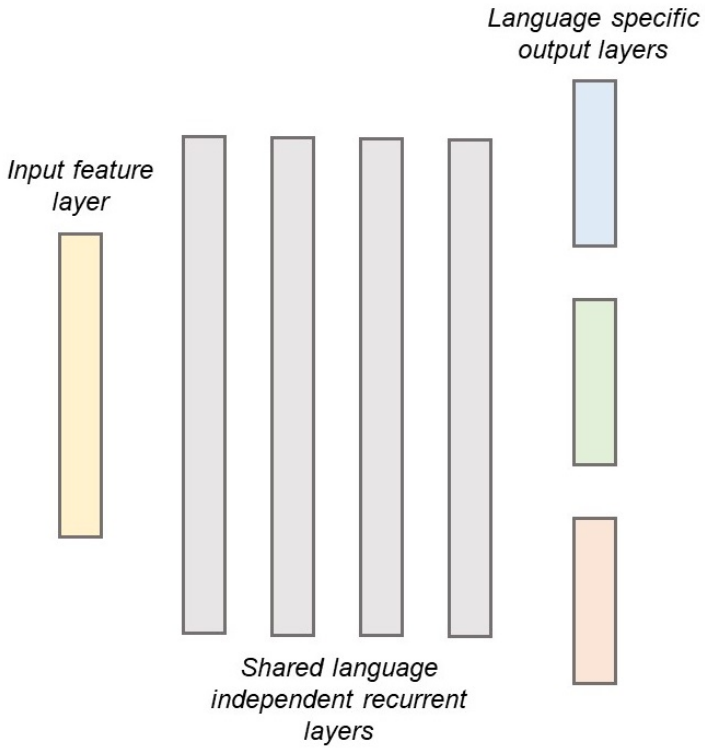


FIGURE 8.1: Architecture of multilingual acoustic model proposed by (Thomas et al., 2020). For training, all the layers are shared for all the languages except the language-dependent output layer. Transcriptions are obtained from all the output layers for each language for the proposed data augmentation technique to train the monolingual ASR models of each language.

or the alignments from the target language ASR model are used as targets to train a mapping model, the alignments (one-hot) have been used previously for the training of the MESD model in the Chapter 5. However, the only difference in MESD models here is that the models are trained using posterior distribution from monolingual ASR models as targets rather than the alignments. Training data for the mapping model is decoded through in-domain pre-trained monolingual model and the frame-level posterior distributions are used as the targets. Though the training data of the mapping model is already seen by the monolingual acoustic model during their training, decoding that data from the pre-trained monolingual acoustic model still poses the chances of mis-recognition. It raises the chances of noisy data for the mapping models' training which is likely to impact their performance. So, the loss function for training of a mapping model is KL divergence loss rather than the cross-entropy loss (reduced form of the KL loss). The loss function of each encoder-decoder module in a MESD model is then given as Equation 5.1

$$\mathcal{L}_{S_iA}(\theta) = \sum_{t=1}^{T}\sum_{k=1}^{d_A} p_{t,k}^{A} \cdot (\log p_{t,k}^{A} - \log p_{t,k}^{S_iA})$$

A MESD model is trained using a rank-weighted sum loss of all the source languages encoder-decoder modules (Equation 5.3).

As discussed earlier, mapping models are trained on frame-level data i.e. posterior distributions from each frame are an example of the training. Since the phoneme-based TDNN acoustic models are trained on frame level using MFCC features as the input, the number of output frames is also theoretically equal to the number of input frames. However, the e2e speech recognition models (described in detail in the Section 8.4.2) are trained using stacked filterbank features of the whole utterance as the input. The dimensionality of these 2D stacked features is reduced in some initial CNN layers which are then fed to a RNN encoder. During the training, the auto-regressive decoder is fed with the outputs of the encoder and an embedding layer is trained to generate the embeddings of the target output tokens. The decoder then generates the output sequence, typically consisting of text tokens (such as characters, words, or subwords), one token at a time. During decoding, the decoder takes the outputs of the encoder and generates a hypothesis in an auto-regressive fashion where the embeddings of the previous output token are fed to the next state. However, the output frames are reduced by compressing the output tokens. Furthermore, the sentence is terminated as soon as an end-of-sentence token is encountered. So, the number of output tokens is fewer than the number of input tokens. Since the number of training examples for a mapping model is dependent on the number of output frames which is lesser than the input frames, the training data for mapping models is also way less than that used to train mapping models on the top of phoneme-based DNN-HMM hybrid models. Though one can consider a lot of examples from those systems as copies of the same examples (since multiple frames

represent one phoneme), posterior distributions of all the frames of one phoneme are not necessarily identical. Therefore, such examples still help to improve the training of a neural network.

Though the details of e2e speech recognition models used for experimentation in this chapter are described in detail in the following sections, the reduction in training examples due to CNN layers is briefly described here. For an ASR system, usually the input features are extracted using a window of 10ms which gives 100 frames per second. However, the neural networks of the hybrid DNN-HMM models are Kaldi's LF-MMI models which use the reduced frame rate i.e. about only 33 frames per second. Assuming an utterance of t milliseconds, the number of output frames (or training examples for mapping models) is given as $t/30$. However, in the case of e2e models, input features are extracted using a window of 10ms and then stacked in 2D. A simple convolution layer of rectangular $kernel = (h, w)$ (assuming unit dilation and stride, and no padding) reduces the dimensionality of an input (H_{in}, W_{in}) as following;

$$H_{out} = H_{in} - 2$$

$$W_{out} = W_{in} - 2$$

where H and W are the number of input frames and filterbank dimensions respectively. We are only interested in H (number of frames) here. In e2e ASR models here, two such CNN layers are applied with a time pooling (layer of kernel size 2) after each layer. Finally, a time pooling layer (of kernel size 4) is applied. With all this configuration, the final number of output frames is given as

$$\begin{aligned} &= (\frac{\frac{(t/10)-2}{2} - 2}{2}) \cdot \frac{1}{4} \\ &\approx \frac{t-30}{160} \end{aligned}$$

which is quite smaller than the $t/30$ of hybrid acoustic models. However, zero padding is applied to speech signals in a batch to keep a uniform size of all the examples. So, technically this is the lower limit of the number of frames but appended frames are considered as non-speech frames (zero padding) and do not contribute towards the training of a mapping model.

8.3 Mapping models for data augmentation

As described in Section 8.1.1, *Thomas et al.* ((Thomas et al., 2020)) have used target language ASR for transliteration of source language audio data (architecture shown in Figure 8.1). Then this source language audio data with its transliterations from a source language ASR have been used for target language data augmentation. An ASR for the target language is retrained with augmented data. However, the target language ASR

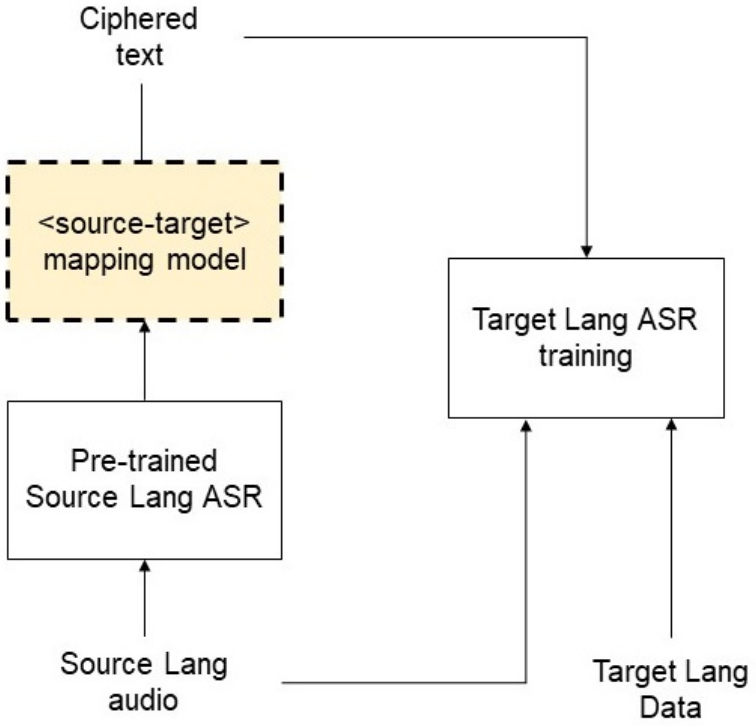


FIGURE 8.2: Flow of generating data for augmentation and retraining of target language ASR

layer has not been trained on source language data and thus transliterations of source language data from a target language layer are not expected to be rational.

In this work on the contrary, source language audio data is decoded using in-domain ASR (M_{S_i}) and then the output posterior distributions (P^{S_i}) are transformed to the target language posterior distributions (P^{S_iA}) using the source-target mapping model ($\mathcal{M}_{S_iA}$). Since an in-domain acoustic model is used for speech recognition and the output posterior distributions are then mapped using a source-target mapping model, both the components are trained for the tasks they are being used during inference. Mapped posteriors from the mapping models are then used to generate transliterated transcriptions (alternatively referred to as *ciphered* text or transcriptions) using greedy decoding. Though the transliterations still might not be exact transliterations (thus called *ciphered* text), both the components involved in the process are trained using the task-specific data and are expected to perform better.

A source language audio data and their ciphered transcriptions for the target language are then used as augmented data to retrain a target language ASR. The flow of the data augmentation and retraining is shown in the Figure 8.2.

TABLE 8.1: Details of Babel data sets used for training e2e ASR models

Lang	Train		Eval	
	# hours	# spks	# hours	# spks
Tamil *(tam)*	59.11	372	7.8	61
Telugu *(tel)*	32.94	243	4.97	60
Cebuano *(ceb)*	37.44	239	6.59	60
Javanese *(jav)*	41.15	242	7.96	60

8.4 Experimental Setup

8.4.1 Data set

As this work extends the previous work, experiments here are done on the same four languages of the Babel data set as used in the previous Babel experiments. Full language packs of four low-resource languages from IARPA Babel speech corpus (Gales et al., 2014) (Tamil (*tam*), Telugu (*tel*), Cebuano (*ceb*) and Javanese (*jav*)) are used for baseline ASR training and evaluation. However, only utterances with a duration of less than 20 seconds are included in the training due to computational limitations. Furthermore, all the non-speech utterances are discarded. So, the overall training data for e2e ASR models is less than the one used to train phoneme-based hybrid DNN-HMM ASR systems. The details of the data sets are tabulated in Table 8.1.

For training of the mapping models, a subset of 30 hours is randomly selected from each language pack. This data is further split into 29 hours of train set and 1 hour of dev set. The summary of training examples for the mapping model of each target language is given in Table 8.2.

If compared the data of Table 8.2 with the training data of mapping models on top of phoneme-based hybrid DNN-HMM models (Table 5.2), it can be seen that the same amount of training data (29 hours) produces around 1.5 million training examples for e2e models compared to 3.2 million examples from the of hybrid models. Similarly, the examples in the eval set for e2e systems are about one-third of the data produced in the case of hybrid models. The reason is that a lot of utterances in the Babel data sets carry no speech at all which have been discarded in the case of e2e systems. It has been discussed in detail for phoneme-based hybrid acoustic models in Section 5.6.5 and is discussed for e2e models in Section 8.5.1.

TABLE 8.2: Examples (in millions) for training of mapping models for each target language. Train set: 29 hours; Dev set: 1 hour; Eval set is same as for the ASR

Language	Train	Dev	Eval
Tamil *(tam)*	1.881	0.065	0.506
Telugu *(tel)*	1.611	0.078	0.358
Cebuano *(ceb)*	1.471	0.051	0.379
Javanese *(jav)*	1.479	0.049	0.400

8.4.2 Speech recognition systems

Hybrid CTC/attention architecture (Kim et al., 2017) is used to train all speech recognition models which consists of three modules that are; a shared encoder, an attention decoder and a CTC module. The input to the model is 40 filterbanks and the output of the model is the BPE tokens. Monolingual ASRs are trained for 100 BPE tokens for each language while the output of multilingual ASR is 400 tokens. SentencePiece library (Kudo, 2018) is used for tokenisation. The acoustic model is a sequence-to-sequence encoder-decoder model where the encoder is a combination of 2 convolutional blocks, 4 LSTM layers and 2 fully connected layers while the decoder consists of a single GRU layer. As described earlier, each convolution block consists of one CNN layer followed by a time pooling layer. There is a time pooling layer as well after the two blocks. The encoder of a model outputs an embedding of dimension 128 which is fed to the decoder. A combination of CTC and cross-entropy losses is used as a training criterion. CTC loss is calculated after a fully connected layer on the top of the encoder and CE loss is estimated on the output of the decoder. The training process jointly optimises the weighted sum of CTC and attention losses.

$$\mathcal{L}_{ASR} = \alpha\mathcal{L}_{CTC} + (1-\alpha)\mathcal{L}_{att} \tag{8.1}$$

During decoding, the final prediction is made based on a weighted sum of log probabilities from both the CTC and attention components. Given a speech input X, the final prediction $\hat{Y}$ is given by;

$$\hat{Y} = \arg\max_{Y\in\mathcal{Y}}\{\lambda \log P_{CTC}(Y|X) + (1-\lambda)\log P_{att}(Y|X)\} \tag{8.2}$$

where λ is a hyper-parameter. The values of α and λ are kept same for all ASR systems. SpeechBrain toolkit (Ravanelli et al., 2021) is used for training all ASR systems.

8.4.3 Mapping models

Multi encoder single decoder models are trained with the same configuration as for phoneme-based hybrid DNN-HMM models (5.3.3). A MESD model is trained for each target language. In an MESD model, there are three encoders and only one attention decoder. Each encoder and single decoder consists of one bidirectional RNN layer. For each target language, the mapping model size is only 2.59 million parameters. The performance of these mapping models is measured in terms of accuracy as given in Algorithm 2.

8.5 Results and Discussion

8.5.1 Mapping models

As discussed earlier, the posterior distributions of target language data from an in-domain monolingual acoustic model are used as targets to train the mapping models.

TABLE 8.3: Performance of e2e monolingual ASR models on train set of each language in terms of %CER

Language	% Character Error Rate
tam	28.57
tel	33.72
ceb	20.98
jav	22.72

Though mapping models for phoneme-based hybrid systems have been trained using alignments from the acoustic model, it is hard to get alignments from an e2e model trained with CTC loss. However, using posterior distributions from an ASR model as a target for training poses the challenge of noisy training data. Even if the model is trained using the same data, the model does not guarantee to recognise training data perfectly. To analyse the performance of monolingual models on their own training data, the %CER is shown in Table 8.3 for all the target languages. It can be seen that the error rate of these monolingual models is still not very good on the training data which means that there are noisy examples for the training of mapping models.

Accuracies of mapping models, trained to map posterior distribution from a source language ASR to the target language ASR, are tabulated in Table 8.4. Analysis shows that the correct target class is still among the top n mapped classes if not the most probable one. So, the mapping models' accuracy is calculated for different values of n where n represents the number of most probable classes. Though the accuracy increases with increasing value of n, the rate of change is not as much as observed in the case of phonemes (Section 5.6.2) which implies that the performance of the mapping model in the case of phoneme based hybrid DNN-HMM systems has been better than that for

TABLE 8.4: Accuracy of baseline mapping models considering top n mapped classes. The accuracy of a source-target mapping model is calculated using Algorithm 2.

Target Lang	Source Lang	*Mapping model* accuracy			
		n=1	*n=2*	*n=5*	*n=10*
tam	*tel*	47.46	54.58	66.31	77.06
	ceb	45.98	52.88	64.25	74.65
	jav	46.97	54.02	65.63	76.26
tel	*tam*	48.88	56.20	67.80	78.28
	ceb	46.22	53.27	64.97	75.96
	jav	47.40	54.78	66.76	77.54
ceb	*tam*	60.53	66.32	74.79	82.31
	tel	48.32	51.43	56.49	62.53
	jav	65.04	71.39	80.06	86.58
jav	*tam*	62.24	68.40	77.00	83.76
	tel	54.64	57.92	62.29	67.69
	ceb	65.51	71.85	80.30	86.65

e2e systems. The reasons include the lesser training data and the detrimental effect of noisy data sets.

Since there are no consistent trends in the performance of mapping models are observable here (similar to phoneme-based hybrid ASR systems), an analysis is carried out to understand the reasons behind it. As the analysis for phoneme-based hybrid acoustic models has shown that there is still a lot of silence/non-speech in utterances, the analysis is started from the same point. Though the non-speech utterances are already dropped in the case of e2e models, a lot of utterances with a little speech can still have a lot of non-speech frames. Silence and non-silence frames have been visualised in 5.6.5 for phoneme-based hybrid DNN-HMM models, alignments cannot be obtained from end-to-end ASR models and spectrograms cannot be visualised here. The reason mainly is that the number of output frames is not based on the number of input frames and an output frame cannot inform which speech segment it belongs to. Having the output posterior distributions from acoustic models, speech and non-speech frames (mapping model training examples) are counted and summarised in the Table 8.5.

As it is evident from Table 8.5, data sets used for training and evaluation of the mapping model consist of a huge amount of silence or zero-padded frames and thus the accuracy of mapping models in Table 8.4 is highly influenced by the performance of mapping models on the silence examples. So, the accuracy of the mapping model is calculated again ignoring the silence examples and tabulated in Table 8.6.

Though the amount of audio data for the training of the mapping model is the same for all the languages, the mappings for *ceb* and *jav* target language are better than *tam* and *tel* . It is even clear from the Table 8.5 that the non-silence examples for *ceb* and *jav* are less than those for *tam* and *tel* , and accuracies of mapping models for these two languages are still better. As the monolingual ASR models of *ceb* and *jav* perform better than *tam* and *tel* on their training data sets (Table 8.3), it is evident that the cleaner data helps to achieve better mapping model performance. Even for *ceb* and *jav* target languages, the accuracy of mappings from *tel* source language is very low in comparison to other source languages. Though both *tam* and *tel* are from a different language family, their higher CER on training data can also be a factor which implies

TABLE 8.5: Statistics of speech and non-speech examples (in millions) for training and evaluation of mapping models. *total:* Number of total examples, *NS:* Number of non-speech frames (examples), *%:* Percentage of speech examples in the data set

Lang	Train			Dev			Eval		
	total	NS	%	total	NS	%	total	NS	%
Tamil *(tam)*	1.88	1.02	45.88	0.07	0.04	43.64	0.51	0.27	45.91
Telugu *(tel)*	1.61	0.85	47.41	0.08	0.05	41.33	0.36	0.19	46.10
Cebuano *(ceb)*	1.47	0.75	49.22	0.05	0.03	49.06	0.38	0.20	48.27
Javanese *(jav)*	1.48	0.78	47.21	0.05	0.03	48.64	0.40	0.21	47.58

TABLE 8.6: Accuracy of MESD mapping models with and w/o considering non-speech (NS) frames for top-1 ($n = 1$) mapped class. The accuracy of a source-target mapping model is calculated using Algorithm 2

Target Lang	Source Lang	*Mapping model* accuracy	
		w/ NS	w/o NS
tam	*tel*	47.46	44.76
	ceb	45.98	41.89
	jav	46.97	43.80
tel	*tam*	48.88	47.19
	ceb	46.22	43.62
	jav	47.40	45.19
ceb	*tam*	60.53	44.04
	tel	48.32	41.24
	jav	65.04	52.52
jav	*tam*	62.24	52.05
	tel	54.64	41.62
	ceb	65.51	57.36

that the training of mapping models is not only affected by the noisy data of the target language but also that of a source language.

8.5.2 Speech recognition performance

Monolingual systems (*mono*) are the language-dependent acoustic and language models which are trained on target language-specific data sets. The train sets of all the languages are then mixed to train a multilingual system (*multi*). The language model for a multilingual system is also trained using a mix of corpora of individual languages. The results of speech recognition systems are shown in Table 8.7. The first row contains the monolingual ASR result without using LM for a later comparison while the rest of the results are ASR decoding with LM.

From Table 8.1, *tam* has more training data than the rest of the languages but the %CER of *tam* and *tel* is worse than *ceb* and *jav* . One potential factor is that the number of BPE tokens is restricted to 100 for all the languages. *ceb* and *jav* with only

TABLE 8.7: Speech recognition performance in terms of %CER. *mono*: monolingual ASR models without explicit LM, *+LM*: an explicit in-domain language model is integrated for all the rest of the results, *multi*: multilingual ASR model, *multiAug*: data augmentation from all the languages for retraining the monolingual ASR model, *crossAug*: data augmentation from only the closest language

Lang	tam	tel	ceb	jav
mono	44.6	58.24	39.40	42.42
+ LM	39.25	52.68	31.25	32.11
multi	41.15	54.38	38.91	42.65
multiAug	41.90	56.10	32.30	32.86
crossAug	**38.83**	**52.06**	**29.94**	**30.47**

TABLE 8.8: Cross-lingual speech recognition performance (in terms of %CER) on top of mapping models. Greedy decoding is applied on mapped posteriors from each source language for a target language

Target Lang	*Source Languages*			
	tam	*tel*	*ceb*	*jav*
tam	*44.60*	49.34	49.21	49.03
tel	63.19	*58.24*	64.33	63.65
ceb	48.10	65.31	*39.40*	40.94
jav	46.72	56.92	40.88	*42.42*

19 and 26 characters respectively and have good context coverage in 100 BPE tokens. However, the BPE tokens extracted for *tam* and *tel* , which have far more characters, do not cover context very well. As *tel* has the smallest training data among all the languages and also suffers from the BPE tokens size problem, the %CER on the training set and evaluation set is the worst. Though the training data for *ceb* and *jav* is still less than *tam* , they have good context coverage in 100 BPE tokens and the performance is better than the rest of the two languages. Furthermore, both *ceb* and *jav* are written in Latin script and thus have a full overlap of characters and are even acoustically close which helps them to train better mapping models as well. On the other hand, even though both *tam* and *tel* belong to the same Dravidian family, their writing scripts are different which makes it difficult for a model to learn mappings with a limited number of BPE tokens.

For a given target language, cross-lingual speech recognition results are also computed on top of mapping models after decoding target language data using source language acoustic models. Greedy decoding is applied on mapped posteriors and the results are shown in Table 8.8. CER on the diagonal is the same as the first row of Table 8.7. Though these results are from source language ASR followed by a source-target mapping model and do not use language-dependent ASR, it performs better than monolingual ASR in the case of *jav* . Results are comparable for other languages but fairly depend on the mapping model's performance. It is evident from these results that a source language acoustic model can be used for decoding a target language followed by a mapping model trained on a limited amount of data.

8.5.3 Data augmentation

For a given target language, audio data of all the source languages is decoded using language-dependent ASR systems and the source posterior distributions are then mapped to the target language distributions using the mapping models $\mathcal{M}_{S_iA}$. Greedy decoding is carried out on these output posterior distributions to generate ciphered transcriptions for the target language. The language model is not integrated at this stage to avoid LM's effect on transliterations. As this stage solely depends on mapping models, the quality of ciphered text depends on mapping models' accuracy (for $n = 1$). The analysis of ciphered transcriptions shows that the transliteration is fairly good for

Actual *(ceb)*	hello
Ciphered *(tam)*	இஹலோ
Ciphered *(tel)*	ఎలో న్ ్ో
Ciphered *(jav)*	hlo
Actual *(ceb)*	singkuwinta
Ciphered *(tam)*	சுயலாா
Ciphered *(tel)*	ర సిం ్ాం
Ciphered *(jav)*	sing pull ka

FIGURE 8.3: Examples of ciphered transcriptions

shorter utterances but gets worse for longer utterances. A few examples of ciphered transcriptions are shown in Figure 8.3.

Ciphered transcriptions are generated from all the source languages for a target language using mapping models as described in Section 8.3. Then the audio data of source languages and the ciphered transcriptions are used together as augmented data for retraining of target language ASR (*multiAug*). As described earlier, the quality of ciphered transcriptions depends on the performance of mapping models, using ciphered transcriptions data augmentation from all the source languages includes very low-quality transcriptions and has a detrimental effect on retraining of target language ASR. So, the augmentation is then restricted to use ciphered data from only the closest language (*crossAug*). For a target language, the source language with the highest mapping model accuracy is chosen as the closest language. By augmenting this data for retraining of a target language, a relative gain of up to 5% and 28.5% is achieved in terms of CER (*crossAug*) when compared with monolingual and multilingual ASR performance.

8.6 Summary

In this chapter, the approach of mapping models (from previous chapters) has been discussed and extended for e2e speech recognition systems. In the last few chapters, mapping models have been discussed to be trained on top of phoneme-based hybrid DNN-HMM acoustic models. However, the learning of mapping models for phoneme-based hybrid acoustic models is expected to be less challenging compared with those trained on e2e models. So, the mapping models have been trained on top of e2e speech recognition models. The speech recognition models are encoder-decoder models with a few convolutional blocks to process the input before feeding to an RNN encoder. These

convolution blocks reduce the frame rate of the input and consequently, we have fewer output frames compared to input frames. Since mapping models are trained on output posteriors, the output frame rate reduction reduces the training data for mapping models in this case compared to those trained on hybrid models. These mapping models have been employed for a novel data augmentation technique to improve low-resource speech recognition. Audio data of an out-of-domain language is decoded through its ASR and then mapped to the target language using the pre-trained mapping model. Then, this audio data along with its transcriptions in the target language has been augmented for retraining of the ASR of the target language.

The experimentation in this chapter has been done for four low-resource languages from the IARPA Babel corpus i.e. Tamil, Telugu, Cebuano and Javanese. The mapping models trained on top of e2e give an average accuracy of 46.27%. The proposed data augmentation approach reduces the character error rate by a relative of 28.5% and 5% compared to multilingual and monolingual speech recognition baselines.

Chapter 9

Multilingual knowledge distillation

Student-teacher learning or Knowledge Distillation (KD) (Hinton et al., 2015) has been previously used to improve the performance of speech recognition systems for the languages which show degradation from multilingual ASR systems (Leal et al., 2021). The loss used in training of knowledge distillation is usually the KL-divergence between posterior distributions from student and teacher model (Hinton et al., 2015, Leal et al., 2021). It implies that both posterior distributions, from the teacher and the student models, should be the same in terms of output tokens. Formally stated, a limitation of KD training is that the student model classes must be a proper or improper subset of the teacher model classes. This limitation prevents knowledge distillation from even acoustically similar languages if the character sets are not the same.

In this chapter, a novel approach is proposed to bridge the gap in distilling knowledge from teachers of diverse languages to a target language student model. The work on multilingual knowledge distillation here is motivated by my collaborated work on knowledge distillation for domain-adaptation (Ahmad et al., 2023) for English data which is presented in Section 9.2. The work on domain adaptation shows that the Out-of-Domain (OOD) data helps to improve the performance of a student domain without using labelled data. This approach could be helpful for low-resource languages and motivates to use knowledge distillation for low-resource languages. However, as mentioned in the literature revisited in Section 9.1.2, there is a gap which hinders the use of cross-lingual

This chapter is based on the publication in IEEE ASRU 2023;
M. U. Farooq, R. Ahmad, T. Hain, *"MUST: A Multilingual Student-Teacher Learning approach for low-resource speech recognition"*, ASRU 2023.

This work is motivated by the collaborated work published in ICASSP 2023
R. Ahmad, M. A. Jalal, **M. U. Farooq**, A. Ollerenshaw and T. Hain, *"Towards Domain Generalisation in ASR with Elitist Sampling and Ensemble Knowledge Distillation"*, ICASSP 2023

ASR systems as teacher models to train a low-resource student model. In Section 9.3, mapping models are employed to bridge that gap.

This chapter is structured as follows; in Section 9.1, literature is revisited for both, knowledge distillation for domain adaptation and multilingual knowledge distillation. The collaborated work on knowledge distillation for domain adaptation is presented in Section 9.2. The proposed approach to bridge the gap in exploiting cross-lingual ASR systems for knowledge distillation is described in Section 9.3.

9.1 Previous work

Knowledge distillation is a popular technique to distil the knowledge from a model (referred to as the teacher model) to train another (student) model. The teacher model can be a single model or knowledge can be transferred from multiple teacher models to train a single student model. The motivation for training a student model from ensemble teacher models has been to replace these computationally expensive multiple models with only one model (Hinton et al., 2015). The model ensemble has been a widely used technique where various models are trained for one task and their ensemble is used to predict the output for a given evaluation example (Sagi and Rokach, 2018). Since multiple models are needed to be trained and used for inference, the computational complexity is very high. So, the motivation of knowledge distillation has been to train a single model which could showcase the abilities of all different models. In knowledge distillation, various teacher models distil their knowledge to a single student model. However, it is also used to train a smaller student model distilling knowledge from a single bigger teacher model (Li et al., 2014b). The idea is to train a smaller capacity model from a bigger model without the loss of validity. As smaller models are computationally less expensive, they can be deployed on less powerful hardware devices such as edge or handheld devices.

Let's assume the knowledge from a teacher model M_T is distilled to train a student model M_S. Given a training example, M_T and M_S output the posterior distributions P_T and P_S respectively and y_{true} are the true labels (synonymously hard labels). The loss of the student model is calculated as the weighted sum of the two losses.

$$\mathcal{L} = (\lambda)\mathcal{L}_{seq}(P_S, y_{true}) + (1-\lambda)\mathcal{L}_{KD}(P_S, P_T) \tag{9.1}$$

where $\mathcal{L}_{KD}$ is KL-divergence loss and $\mathcal{L}_{seq}$ is a cross-entropy loss. Losses in the above equation are synonymously regraded as supervised loss and KL loss respectively. λ is a hyper-parameter to weigh both the losses.

Knowledge can also be transferred from teachers' hidden layers to the student's hidden layer (Romero et al., 2015) but usually, a student model is trained using soft labels from teachers' outputs (posterior distribution or output of the neural network before softmax layer) (Gou et al., 2021). A common technique is to transfer the knowledge on

the output layer using teachers' posterior distribution (Gou et al., 2021). Usually, the student model is trained with the teacher's posteriors (soft labels) along with the original training labels (hard labels). Therefore, the total loss is the weighted sum of supervised and KL losses. KD is used for many tasks such as domain adaptation (Asami et al., 2017, Meng et al., 2019, Zhu et al., 2020), domain generalisation (Fang et al., 2021, Kim et al., 2021, Wang et al., 2021), and model compression (Chebotar and Waters, 2016, Kim et al., 2019, Takashima et al., 2018).

In automatic speech recognition, knowledge distillation is performed over either frame-level (Hinton et al., 2015) or sequence-level (Huang et al., 2018, Wong and Gales, 2016) span. Initially, knowledge distillation has been proposed in speech recognition models trained with cross-entropy loss. However, sequence loss-trained acoustic models have outperformed conventional cross-entropy trained acoustic models (Graves, 2012). However, as given in Equation 9.1, KD loss is the sum of cross-entropy (supervised) and KL loss which is calculated on the frame level. So, it has been a challenge to distil knowledge from the teachers who are trained with sequence loss. Sequence level KD has been first proposed for neural machine translation (Kim and Rush, 2016) where the teacher models provide sequence-level probability distribution over the whole sample space for better knowledge transfer. Following the sequential nature of the speech recognition task, a sequence-level transfer is shown to be a better approach in (Wong and Gales, 2016) where the authors have proposed knowledge distillation from teachers' models to train a student model using maximum mutual information sequence loss. The concept has been later extended to use joint CTC/attention loss for the student model training (Yoon et al., 2021). In the domain adaptation part of this work, knowledge is distilled from multiple teacher models to train a domain-generalised student model. In the case of multiple teacher models, it is crucial to optimise the sampling strategy to obtain the best possible output. Various ensemble techniques along with a proposed Elitist Sampling (ES) technique are experimented with.

Most of the work done in the knowledge distillation domain has been done where the teachers and students have the same output tokens (Hinton et al., 2015, Huang et al., 2018, Wong and Gales, 2016). Since KL divergence is computed between student and teacher models' posteriors, the output tokens of the student model should be a proper or improper subset of the teacher models' output tokens. It hinders the distillation of knowledge across the acoustically closer languages if they have different sets of alphabets. In this work, a novel approach is proposed to bridge this gap by exploiting mapping models.

In the following sections, the literature is revisited for domain adaptation and multilingual knowledge distillation.

9.1.1 Domain adaptation

This section describes the work on knowledge distillation for domain adaptation as a precursor of the multilingual knowledge distillation work. As described earlier, the main

motivation of knowledge distillation has been its application for model compression to reduce the computational complexity (Chebotar and Waters, 2016, Kim et al., 2019, Takashima et al., 2018), it has also been used for domain adaptation and generalisation for speech applications. Previous work has shown that an acoustic model trained using speech data recorded for a specific domain does not perform well on out-of-domain data (Likhomanenko et al., 2021). For instance, Likhomanenko et al. (Likhomanenko et al., 2021) have shown that an acoustic model trained on LibriSpeech gives a word error rate of 2.8% on in-domain dev-clean data set but the error rate jumps to 30% for CommonVoice test set. It implies that an acoustic model trained on speech data recorded in one domain does not guarantee the same performance for the data set from a different domain even though the data is of the same language (i.e. English in this case). It calls for some techniques to build a more generalised model or ways to adapt one model for OOD data with minimal effort. This would be more challenging in the case of multilingual speech recognition where the languages are different and might not share linguistic characteristics.

Various approaches have been tried to overcome the aforementioned problem including retraining of a limited number of parameters (Gemello et al., 2007, Neto et al., 1995), learning hidden unit contributions (Swietojanski and Renals, 2014a) and use of global domain capturing auxiliary features (Abdel-Hamid and Jiang, 2013, Delcroix et al., 2016). In limited parameters retraining, a model trained for some specific task is retrained using an out-of-domain train set. Retraining can be done in two ways i.e. retraining all the parameters or updating only a few hidden and final layers (Huang et al., 2013, Vu and Schultz, 2013). LHUC introduces domain-specific hidden units after each or some selected layers for which the weights are learnt during training (Swietojanski and Renals, 2014a). At inference time, only the domain-specific hidden unit is activated. Another popular approach is to use regularise optimisation in retraining a model to avoid catastrophic forgetting during retraining (Yu et al., 2013). The regularisation term in the objective function controls the gradient updates so that the pre-trained model also retains its previous information while learning the new task. L2 or KL divergence regularisation is usually used between a trained and the source model. Knowledge distillation is based on regularisation approaches which regularise the distillation of the knowledge from a teacher model to a a student model during its training. Student models trained with the KD approach prove to generalise better than the conventional regularisation approaches (Asami et al., 2017).

In (Asami et al., 2017) and many other knowledge distillation approaches, both hard and soft labels are used for loss calculation during the training of a student model. A weighted sum of both, hard and soft label losses, is used to calculate the total loss. Usually, the cross-entropy loss is used for hard labels and the KL divergence loss is computed for soft losses. Meng et al. (Meng et al., 2019) proposed an unsupervised approach to train a student model using only the soft labels without relying on the hard

labels. KL divergence between teacher (target) and student (hypothesis) model posterior distributions are calculated and used as a loss function for the student training.

Since the original KD work has shown that a student trained with an ensemble of multiple teachers outperforms a student trained using a single teacher model (Hinton et al., 2015), Gao et al (Gao et al., 2021) has shown that a student acoustic model trained using multiple teacher models of a different domain, generalises well. Various ensemble techniques have been experimented and the results have been compared. In the presence of multiple teachers, a very conventional method is to take an average of output posterior distributions of all the teachers' models at a given time frame (usually referred to as teacher averaging) (Chebotar and Waters, 2016, Fukuda et al., 2017). Another approach is to select the best model among all the teacher models to distil the knowledge (Gao et al., 2021). The best teacher can be selected on an utterance level or a frame level. The output posterior distribution of the best teacher is used to calculate the KL divergence loss for the student model training. Rather than choosing the best teacher model, the average of the top 'K' teacher model has also been explored. Furthermore, a dynamic weighting of teachers has also been experimented with where each teacher is assigned a weight based on the error of each teacher for a given input audio utterance (Gao et al., 2021). Ground-truth transcriptions are used to measure the error of each teacher which assumes that the hard labels are available beforehand. However, this approach is not helpful in those scenarios where the goal is to adapt unseen out-of-domain data. Hence, an unsupervised sampling strategy is necessary to reduce the uncertainty of multiple teacher outcomes and select the best output. The results of (Gao et al., 2021) have shown that none of these schemes generalises well. In this work, an elitist sampling strategy is proposed for the teachers' ensemble to select the best-decoded utterance generated by completely out-of-domain teacher models for generalising unseen domains. An OOD student model is trained in a fully unsupervised fashion.

Furthermore, the performance of a student model fairly depends on the underlying teacher models. A teacher model is important for the generalised representation of the acoustic samples. Representations from various models such as BERT(Futami et al., 2020) and BLSTMs (Kurata and Audhkhasi, 2018) with different context windows have been used in ASR knowledge distillation tasks. Self-supervised training paved the path to learning the general data representation through unsupervised pre-training. Such a model is trained with masked spans for generalised contextual latent representation of speech (Baevski et al., 2020). These self-supervised models have been observed to be quickly adaptable to new domains or cross-domain tasks (Hsu et al., 2021a).

9.1.2 Multilingual knowledge distillation

Modern automatic speech recognition systems nowadays require huge amounts of data for training. However, only 23 out of 7000 languages are spoken by more than half of the world's population (Ethnologue). Thus a large number of languages lack enough data

resources to train a modern ASR system. Multilingual and cross-lingual systems have got a lot of attention in recent years to exploit resources of other languages to overcome the data scarcity issue for training of speech technologies for low-resource languages (Abate et al., 2020, Besacier et al., 2014, Karafiát et al., 2016, Tachbelie et al., 2020a). Although multilingual ASR systems are considered to perform better when compared with their monolingual counterparts of low-resource languages, the performance of these systems often degrades due to mixing of unrelated languages (Gaur et al., 2021, Hou et al., 2020, Pratap et al., 2020a). This has given rise to various studies intending to improve a monolingual ASR using multilingual or cross-lingual resources rather than training a unified model (Klejch et al., 2022, Morshed and Hasegawa-Johnson, 2022, Xu et al., 2022). Recently, some efforts have been made towards multilingual knowledge distillation where multilingual models are used for knowledge distillation to train a language-specific student ASR model (Leal et al., 2021). Such efforts have been discussed briefly in Section 3.3.3 and are recapped here.

As described earlier, student-teacher training or knowledge distillation (Hinton et al., 2015) has widely been used to distil the knowledge from either single or multiple teacher models (Huang et al., 2023) to train a student model. This technique of transferring a teacher's knowledge to a student model either at output layer (Hinton et al., 2015) or at intermediate stages (Romero et al., 2015) has been used for many tasks such as model compression (Huang et al., 2023, Kim et al., 2019) and domain generalisation (Fang et al., 2021, Kim et al., 2021, Wang et al., 2021). The student model is trained with a combined objective of minimising the KL-divergence loss for the prediction of the teacher's posteriors (soft labels) and a classification loss with the original training labels (hard labels).

Since KL-divergence loss is used as KD loss between a teacher's soft labels and the student model posteriors (Hinton et al., 2015), the output classes of the student model should be a proper subset of the teacher model. The reason is that the KL divergence loss is calculated between two posterior distributions and makes sense only if the posterior distributions represent the probabilities of the same classes from two different models. However, multilingual knowledge distillation studies have used teacher models where student model output classes are an improper subset of the teacher model classes (Leal et al., 2021). In that case, to train a student model of a specific language, the posterior distribution of a multilingual model can be used dropping the probabilities of the classes that do not belong to the target language. The updated posterior distribution is normalised before calculating the KL divergence loss. Works have been done to use multiple multilingual teacher models to distil knowledge to train a monolingual student model (Shen et al., 2023). Nevertheless, it still constrains a teacher model to cover all the student classes yet a lot of languages have diverse character sets and writing scripts.

Tan et al. (Tan et al., 2019) proposed to train a multilingual machine translation system by distilling data from monolingual teacher models. Posteriors of pre-trained

monolingual machine translation models have been used to match the posteriors of the multilingual student model. At each training step, a mini-batch of only one bilingual pair is sampled for student training and the posteriors from the corresponding teacher model are used to calculate the KL divergence loss. Though work has been done to train a student model of a specific language using knowledge of various teacher models trained on features from multilingual acoustic models (Cui et al., 2017), the attempt was not to train or directly distil knowledge from a multilingual ASR model. Xu et al. (Xu et al., 2019) have used multilingual and language-specific monolingual ASR models jointly to train a student model of a specific language.

In a recent work (Fukuda and Thomas, 2021), output posterior distributions from individual monolingual acoustic models have been used to train a multilingual ASR model. The student multilingual model has some shared and a few language-specific layers. The objective of this research is to train a multilingual model with some language-agnostic shared layers. These shared layers have been used for the initialisation of a model of an unseen language. Leal et al. (Leal et al., 2021) trained a monolingual student model using multiple cross-lingual teacher models. However, the set of languages has been chosen in a way that the languages share the same set of tokens. However, if the languages are chosen in a way that the output tokens are very diverse, KL divergence loss cannot be calculated between student and teacher models.

As it happens, several languages which are acoustically similar or belong to the same language families are written in different scripts such as Turkish and Kazakh (Turkic), Urdu and Hindi (Indo-Aryan), and Greek and Armenian (Indo-European). It prevents various languages from distilling their knowledge for training of a closer language ASR model. Previous work on knowledge distillation has been done for the domains where the student and teachers are from the same language and have the same output classes. However, no work has been done for either cross-lingual knowledge distillation or to overcome the aforementioned problem. This work on multilingual knowledge distillation presents a step towards overcoming the obstacle of applying KD in cross-lingual settings.

9.2 Knowledge distillation for domain adaptation

The generalisation problem in KD is two-fold. The first problem is optimising the teacher model's learned representation, and the second is choosing the appropriate distribution for a student from the teacher model. These problems become challenging when dealing with OOD data between teacher and student. In this work, the first problem is tackled with pre-trained wav2vec2.0 teacher models where each model is fine-tuned with an in-domain corpus. The second problem is tackled with a posterior-based elitist sampling strategy which selects the best utterance decoded by the teachers. In summary, an ensemble of teacher models is trained on completely OOD data compared to the data which student needs to adapt. An inference is run on the unlabelled OOD data to generate the soft labels from the teachers. Finally, an elitist sampling strategy based on the output posteriors is used to select the best-decoded utterance from the teachers

to train a student model. More specifically, data sets from read speech: WSJ (LDC catalogue LDC93S6A, LDC94S13A), LibriSpeech (Panayotov et al., 2015) and meeting: AMI (Carletta et al., 2006) are used to train three state-of-the-art teacher models. These models are used to decode the Switchboard (Godfrey et al., 1992) corpus, and a student model is trained. The results show that with such selection on the ensemble of teacher outputs, the student model performs better compared to the baselines and all the individual teacher models.

9.2.1 Distillation using CTC loss

In teacher-student training, knowledge is usually transferred from the teacher model in terms of the posterior distribution. The teacher model is already trained on a given labelled dataset. While training the student, its output distribution is tried to become closer to the teacher's posterior distribution. This is usually achieved by using KL divergence (Joyce, 2011) loss as:

$$KL(\hat{y}_{tea}||\hat{y}_{stu}) = \sum_{i=1}^{N} \hat{y}^i_{tea} \log \frac{\hat{y}^i_{tea}}{\hat{y}^i_{stu}} \tag{9.2}$$

where $\hat{y}_{tea}$ and $\hat{y}_{stu}$ represent the teacher and student output posteriors, and N is the total number of examples. Frame-wise KL divergence is usually calculated to match the two distributions. The total loss for the student model is the weighted sum of supervised and KL loss:

$$\mathcal{L}_{total} = \alpha\mathcal{L}_{sup} + (1-\alpha)\mathcal{L}_{KL} \tag{9.3}$$

In sequential tasks such as ASR, the studies show that knowledge can be transferred effectively via sequence-level information instead of frame-level. Therefore, CTC (Graves, 2012) loss has been used in this study.

The teacher and student models for KD are based on wav2vec2 (Baevski et al., 2020) pre-trained with libriVox (LV-60K). Dense neural network layers are connected at the output of wav2vec2 and CTC loss is applied. The raw input to the model is represented by sequence $\mathbf{x} = [x_1, ...x_{T_n}] \in X$ with each sequence of length T_n and X represents all the training sequences. The wav2vec2 model outputs the context features represented by $\mathbf{c} = [c_1, ...c_{T_n}]$. Context vector $\mathbf{c}$ given to the fully connected dense layers produces the output represented by $\mathbf{h} = [h_1, ...h_{T_y}]$, where T_y is the output length with the vocabulary of $G = [g_1, ...g_z]$, where g_z represents grapheme. Teacher models are trained with original labels using CTC loss as:

$$\mathcal{L}_{CTC} = -\sum_{B} \log p(\mathbf{y}|\mathbf{h}) \tag{9.4}$$

where $\mathbf{y}$ is the output label sequence and $B = \{X, Y\}$ is the training dataset.

In a standard distillation setting, the student model is trained with the original (hard) labels and soft labels provided by the teacher model. The total loss for the student model is the weighted sum of hard and soft label losses as presented in Equation 9.3. However, for the domain adaptation problem here, it is assumed that the original labels of the target domain are not available.

The OOD teacher models run the inference on the target data and provide the soft labels to train the student. Hence, the value of α in equation 9.3 becomes zero.

For the distillation loss, the work of (Huang et al., 2018, Takashima et al., 2018) is followed which has proposed to use sequence-level loss rather than frame-level for the ASR task. Therefore, $\mathcal{L}_{KL}$ in Equation 9.3 is replaced by soft CTC-KD loss. This is represented as follows:

$$\mathcal{L}_{CTC-KD} = -\sum_{B} p_{tea}(\hat{\mathbf{y}}|\mathbf{h_{tea}}) \log p_{stu}(\hat{\mathbf{y}}|\mathbf{h_{stu}}) \tag{9.5}$$

where $p_{tea}(\hat{\mathbf{y}}|\mathbf{h_{tea}})$ and $p_{stu}(\hat{\mathbf{y}}|\mathbf{h_{stu}})$ are posteriors from teacher and student models. In this case, the teacher posteriors come from the selection criteria from multiple teachers which is explained in Section 9.2.2.

9.2.2 Utterance selection strategies

This work utilises multiple teacher models to generalise the out-of-domain data for the student model. It is known that the ensemble models usually provide better predictions compared to the single model. One of the most straightforward approaches to distil the knowledge from the teacher models is to average the frame-wise posterior of the teachers following the work of (Chebotar and Waters, 2016, Fukuda et al., 2017) as:

$$\mathbf{p_{tea}} = \frac{1}{K} \sum_{k} \mathbf{p_k} \tag{9.6}$$

were K is the total number of teachers, and $\mathbf{p_k}$ is the posteriors of the whole utterance for teacher k. The resulting $\mathbf{p_{tea}}$ is the frame-wise average posteriors of each utterance. However, this strategy may not perform well because if some teachers perform poorly then the overall average becomes worse. Therefore, a sampling technique is devised which selects the best-decoded utterances from the teacher models.

In (Gao et al., 2021), authors have proposed three distillation strategies, i.e. weighted, Top-1 and Top-K. The authors have proposed to use error rate as a metric of selection/weighting in each strategy. For example, in the weighted technique, the weights for each teacher are determined with the average error on the mini-batch, which can only be calculated if labelled data is present. Moreover, in their technique, the student learns on the same data which teachers have already been trained on. However, this is not the case in here, where the student learns on the data unseen by the teachers and without original labels. Eventually, this technique cannot be directly adapted.

For the unlabelled data, each teacher model may make different mistakes while transcribing the same utterance, and one teacher would perform better than the others. Therefore, it has been proposed to select the utterance from that particular teacher model which best transcribes it. This could be achieved by ranking the decoded posteriors from the teachers. For the proposed selection strategy in this work, the average of the posteriors is computed of the whole utterance from each teacher as follows;

$$q_k = \frac{1}{T_y} \sum_{i=1}^{T_y} \max p_{ik} \tag{9.7}$$

where T_y is the length of utterance and k is the teacher. This average is calculated for each utterance which is decoded by the teachers. The best teacher is selected with maximum average posterior as:

$$b = \arg\max_k q_k \tag{9.8}$$

With such a selection strategy, each utterance is selected from the best-performing teacher b, and its posteriors are used in Equation 9.5 to train the student model. The intuition behind the selection strategy is elitist, i.e. assuming if a group of graphemes projects the highest confidence as a group, they are also best fitted as individuals. The grapheme-phoneme correspondence is optimal when the confidence score in graphemes is maximised as a group. If the posteriors are higher for each label, it refers to the model's confidence. Taking the average of all the frames would provide the model's confidence for the whole utterance. So the best utterance among the three models would have the highest average posterior.

9.2.3 Experimental setup

9.2.3.1 Data

Four different corpora have been used in the experiments comprising read and conversational speech. AMI, LS and WSJ are used to train the teacher models, while SwithBoard is for the student. LS, WSJ and SB are recorded by US native English speakers, while AMI is recorded in three different institutes including Edinburgh, Idiap and TNO, covering both native and non-native English speakers. Among these corpora, AMI and SB belong to the conversational speech and WSJ, and LS belong to the read speech. All of these corpora have different distributions in terms of recording conditions, speaking style etc.

The total duration of LS, WSJ, AMI and SB are 960h, 272h, 100h and 300h. As LS is the largest in duration, only the 360h subset (referred to as LS360) is considered. Moreover, all datasets have a sampling rate of 16Khz except SB. Hence, SB is up-sampled from 8KHz to 16KHz rate. To analyse the performance of the models, the standard test sets from each corpus are considered. LS-test-clean and LS-test-other are clean and noisy

test sets of LS. One test from WSJ is considered, i.e. WSJ-Eval92-20K, which consists of 20K unique words. From AMI, the full corpus ASR test set AMI-FCASC is taken. From SB, two test sets; eval00-CallHome (LDC97S42) represented by CH and eval00-Switchboard (SB) are considered. The eval00 combines both of these CH and SB test sets.

9.2.3.2 Experimental details

The base model for teachers and students is wav2vec2-large pre-trained with LV-60k. The output of wav2vec2 is connected with two fully connected layers, and CTC loss is applied. During training, wav2vec2 is also fine-tuned for both teachers and students. Moreover, wav2vec2 requires audio to be sampled at a 16KHz rate, so only SB audio is upsampled. The grapheme-based tokeniser is used for all the models.

In the experiments, independent teacher models are first trained on AMI, WSJ and LS360 datasets. These models are evaluated on multiple test sets including in-domain and out-of-domain sets as listed in Table 9.1. This evaluation helps to understand the performance of the model on completely out-of-domain distribution to analyse cross-domain adaptation. The language model (LM) used in the decoding is a 3-gram model which is trained on the training transcripts of all the teacher models.

Teacher models are used to decode the SB data and based on the selection strategy discussed in Section 9.2.2 the best decoded-utterance is selected to train the student model. While training the student model, it does not know the original training labels considering that they are not available. Hence, the student only generalises based on the posteriors from the teachers. Moreover, similar to the acoustic models the LM is also an out-of-domain model for the SB data. This is selected to maintain the consistency in this study to just evaluate the unseen domain adaptation. The results of the proposed technique are compared with individual teacher models and two baselines. For the first baseline, the frame-wise average of the posteriors of all the teachers is computed and the resulting posteriors are used to train the student model. For the second baseline, the maximum frame-wise posteriors are selected among the teachers to make the resulting distribution. The resulting posteriors are then used to train the student. The frame-wise maximum value of the posterior represents the confidence of the model for that particular label. Hence, frame-wise highly confident soft-labels are selected. All of these results are discussed in Table 9.2 and Table 9.3, where the first baseline is ensemble using teachers averaging method (TA) and the second one with frame-wise maximum method (FWM). The results using the proposed elitist sampling technique are in the ES column.

9.2.4 Results & Discussion

The teacher models trained in individual corpora were initially evaluated on in-domain and out-of-domain test sets. Table 9.1 shows the WER of the individual test set used for the evaluation. Each column (AMI, LS360 and WSJ) represents the model trained

TABLE 9.1: WER(%) evaluation of the teacher models on in-domain and out-of-domain test sets with and without LM. The three individual teacher models (fine-tuned wav2vec2.0) are trained on AMI, LibriSpeech 360 hours (LS360) and Wall Street Journal (WSJ) datasets. '✗': without using OOD LM, '✓': with using OOD LM. The shaded areas are the results of in-domain test sets.

	AMI		**LS360**		**WSJ**	
LM / **Test sets**	✗	✓	✗	✓	✗	✓
AMI-FCASC	15.85	14.41	51.69	47.42	60.60	57.32
LS-test-clean	14.61	12.41	3.51	3.04	10.14	8.39
LS-test-other	29.69	26.32	10.76	9.51	27.19	24.43
WSJ-Eval92-20K	15.87	13.11	9.60	7.14	2.39	1.47
eval00	52.04	49.21	45.86	42.59	66.43	64.20

TABLE 9.2: Evaluation of WER(%) of student and teacher models on Switchboard (SWBD) test sets without a language model. Utterance election strategies for ensemble: Teacher Averaging (TA), Frame-Wise Max (FWM) and Elitist Sampling (ES).

	Teacher models (w/o LM)			**Student models** (w/o LM)		
				TA	**FWM**	**ES**
sets	**AMI**	**LS360**	**WSJ**	**SWBD**		
eval00	52.04	45.86	66.43	58.11	51.71	**37.38**
CH	56.74	50.55	73.48	62.68	54.63	**41.15**
SB	47.09	40.96	59.07	53.33	48.67	**33.45**

TABLE 9.3: Evaluation of WER(%) of student and teacher models on Switchboard test sets with a 3-gram language model trained on AMI, LS360 and WSJ transcripts. Utterance election strategies for ensemble: Teacher Averaging (TA), Frame-Wise Max (FWM) and Elitist Sampling (ES).

	Teacher models (with OOD LM)			**Student models** (with OOD LM)		
				TA	**FWM**	**ES**
sets	**AMI**	**LS360**	**WSJ**	**SWBD**		
eval00	49.21	42.59	64.20	46.53	38.09	**32.00**
CH	54.03	47.18	71.70	51.84	42.16	**35.72**
SB	44.17	37.80	56.37	40.99	33.84	**28.13**

on that specific corpus and each test set is evaluated with and without LM. It can be seen that each model produces state-of-the-art results on its own test sets represented by shaded cells. For both LS test sets the best cross-corpora WER was produced by WSJ and then AMI. This is because WSJ is also read speech data and has native English speakers. However, both WSJ and LS have different recording and environmental conditions. Therefore, the WSJ model produced more than 2.5 times worse WER for LS-test-clean and LS-test-other with and without LM. On the other hand, AMI is a conversational speech and also has non-native English speakers. Additionally, the WSJ and LS are relatively more clean speech compared to AMI as it has many environmental

and human-generated noises. Similar behaviour is being noticed for the WSJ-Eval92-20K test set. For the AMI test set, LS360 and WSJ produced worse results compared to the AMI model itself. Again the reason is the speaking styles and short spontaneous utterances present in the conversational speech. Finally, the eval00 test, which belongs to the SB data, is best transcribed by LS360. This is due to the large training data (360h) and more speakers compared to the AMI and WSJ. One can argue that the AMI should perform better due to the conversational speech. However, the AMI model is only trained on 86h and has both native and non-native English speakers. While SB only has native English speakers.

Table 9.2 shows the WER of the individual teacher models and student models on SB test sets without LM. Among the student models the teacher averaging does not seem to be a good strategy for ensemble models as the WER is much worse compared to two teachers (AMI and LS360) and also other student models. The main reason is that one of the teachers i.e. WSJ is performing worse, thus deteriorating the average of the teachers' posterior which eventually affected the student models training. Whereas, Fw_max is better than the majority of the teachers and Tea_avg due to the reason that in this strategy maximum posterior is selected among the teachers which corresponds to the high confident prediction. The downside of this technique is that it affects the sequential nature of the output because it selects the posteriors from different teacher models.

Furthermore, other than the proposed the best model among the teachers and students which produced the lowest WER for all the test sets is LS360. However, it cannot be concluded that the LS360 model decodes all the utterances of the SB better than AMI and WSJ to train the students. This is depicted by the evaluation of the proposed student model which has the lowest WER compared to all the teachers and baseline student models. For the CH test set, the WER of the proposed student model is 41.15%, which is almost 9.4% better than the LS360, 21.5% better than Tea_avg and 13.4% better than Fw_max in an absolute difference. Similarly, SB achieved a minimum improvement of about 7.5% compared to the LS360 model. Collectively, for eval00 the proposed student model is almost 8.4% better than the best-performing model LS360.

Moreover, Table 9.3 show that better results are achieved when OOD LM is used for both teachers and student models. Compared to the results in Table 9.2 the proposed student model is improved up to 5.3% in absolute difference using the OOD LM making WER of eval00 to 32.00%. It has also been observed that the results of TA and FWM models have improved significantly eventually making FWM better than all the teachers and TA. This shows that frame-wise posterior sampling seems to be a better candidate if decoding is applied with an LM. Overall, results show the significance of the ensemble of teachers and selection strategy to train the student. As the teacher models are trained on completely out-of-domain data, the knowledge can still be distilled effectively from ensemble models to adapt better student. Therefore, the student performs much better

than the baselines and individual teachers.

9.2.5 Summary

To generalise the unseen out-of-domain data, this work proposes to use ensemble knowledge distillation with an elitist sampling technique. The sampling is performed based on output posteriors, and the best utterance is selected from the teachers to train a student model. With the proposed technique the results show that the student model generalises well on the out-of-domain data compared to the teacher models and baselines. Only the out-of-domain language model is used to study the effectiveness of cross-domain acoustic model variability. Furthermore, intermediate neural representations are analysed across different models and layers to understand the relationship between acoustic data and transcription. In future work, one can improve the sampling strategy and pseudo-label correction using iterative pseudo-labelling and in-domain language models.

9.3 Multilingual knowledge distillation

As described in Section 9.1.2, Student-teacher learning or knowledge distillation has been previously used to address data scarcity issues for the training of speech recognition systems. However, a limitation of KD training is that the student model classes must be a proper or improper subset of the teacher model classes. It prevents distillation from even acoustically similar languages if the character sets are not the same. In this section, the aforementioned limitation is addressed by proposing a Multilingual Student-Teacher (MUST) learning which exploits a posteriors mapping approach. A pre-trained mapping model is used to map posteriors from a teacher language to the student language ASR. These mapped posteriors are used as soft labels for KD learning. As an ensemble of multiple teachers proves to perform better than a single teacher in the work for domain generalisation, various teacher ensemble schemes are experimented with to train an ASR model for low-resource languages.

Earlier, the mapping model approach (described in Chapter 5) has been employed for multilingual and cross-lingual model fusion for speech recognition on phonemes level Chapter 7 and end-to-end ASR systems Chapter 8. In this chapter, a source (teacher) language ASR model followed by a source-target (teacher-student) mapping model is used to act as a teacher for student model training. Source and target are synonymously used as teacher and student respectively for the rest of the discussion in this chapter. For N languages, one mapping model is trained for each target language to map posteriors from other $N-1$ source languages ASR models to the posteriors of the target language ASR. Mapped posteriors from a source language ASR are used as soft labels for knowledge distillation. Having multiple teachers from $N-1$ source languages, different existing weighting schemes along with a proposed Self-Adaptive Weighting (SAW) are experimented with for teachers' ensemble to generate soft labels. The key contribution of MUST learning is to overcome the limitation of multilingual KD and use teachers from diverse languages for multilingual knowledge distillation.

9.3.1 Multilingual Student-Teacher (MUST) Learning

As described at the start of this chapter, output classes of the student model are required to be a proper or improper subset of the teacher model classes for knowledge distillation. It prevents a teacher language to distil its knowledge to train a student model if writing scripts or character sets are not the same. In this work, mapping models are employed to overcome this issue and distil knowledge from diverse teacher languages to train a student model of a low-resource language.

For a given target language (L_A), an encoder-decoder sequence-to-sequence monolingual acoustic model is trained using hybrid CTC loss as given in Equation 9.9.

$$\mathcal{L}_{ASR} = \alpha\mathcal{L}_{CTC} + (1-\alpha)\mathcal{L}_{seq} \tag{9.9}$$

where $\mathcal{L}_{CTC}$ is applied on top of the encoder after an affine projection layer. $\mathcal{L}_{seq}$ is cross-entropy loss which is applied to the decoder's output.

For MUST learning, soft labels from a single teacher or an ensemble of multiple teacher models are used to distil knowledge for the training of a model for low-resource language. $\mathcal{L}_{seq}$ loss in Equation 9.9 is modified as

$$\mathcal{L}_{seq}(\theta) = \lambda\mathcal{L}_{KD} + (1-\lambda)\mathcal{L}'_{seq} \tag{9.10}$$

where $\mathcal{L}'_{seq}$ is still a cross-entropy loss and $\mathcal{L}_{KD}$ is the knowledge distillation loss which is an ensemble of multiple teachers and given as

$$\mathcal{L}_{KD} = \sum_{K} \mathcal{W}_k \mathcal{L}_{T_k} \tag{9.11}$$

$\mathcal{L}_{T_k}$ is KL-divergence loss between posteriors from k^{th} teacher model and the student model.

$$\mathcal{L}_{T_k} = \sum_{B} p^{T_k} \log \frac{p^s}{p^{T_k}} \tag{9.12}$$

where p^{T_k} and p^s are the posterior distributions from k^{th} teacher and the student model respectively. A teacher model is a source language ASR model followed by a source-target language mapping model $\mathcal{M}_{S_iA}$ as shown in Figure 9.1. α and λ in Equations 9.9 and 9.10 are hyper-parameters and different teacher weighting strategies are experimented for $\mathcal{W}$ in Equation 9.11.

Given an utterance u of a target language, it is decoded through all the source language ASR systems (M_{S_i}) which generate posteriors for their output classes (P^{S_i}). Then a pre-trained target language mapping model ($\mathcal{M}_{S_iA}$) is used to map the output posteriors from source language ASR systems to the target language ASR (P^{S_iA}). Output posteriors from the mapping model are used as soft targets for student model training.

So, the ASR of each source language along with a target language mapping model acts as a teacher model for MUST learning. For the target language student learning, experiments are conducted using an ensemble of multiple teachers (source languages) and a single teacher to generate soft labels for KD training.

9.3.1.1 Self-adaptive weighting

Performance of the ensemble teacher models depends on the choice of $\mathcal{W}$ in the Equation 9.11 for each teacher loss. A straightforward approach is the Teacher averaging (TA) where all the teachers are assigned equal weights. However, all the teachers have different relationships with the student task and thus impact differently. In the case of multilingual systems, all teacher languages are not equally similar and assigning equal weights does not prove to be an optimal way.

In this work, a self-adaptive weighting scheme is proposed. Motivated by the domain adaptation work which makes use of posterior distributions, teacher models get relative weights based on their confidence in soft-labels. Furthermore, rather than assigning the same weights for a batch, teachers' weights are calculated on the fly for each utterance. Given an utterance u of T frames, mean of $max(p_t)\forall t \in \{0, 1, \cdots, T\}$ is calculated where p_t is the posterior distribution at time t.

$$\mu_k = \frac{1}{T}\sum_t \max(p_t^{T_k})$$

Then the weight of each teacher is set to

$$\mathcal{W}_k = \frac{\tau^{\mu_k}}{\sum_K \tau^{\mu_k}} \tag{9.13}$$

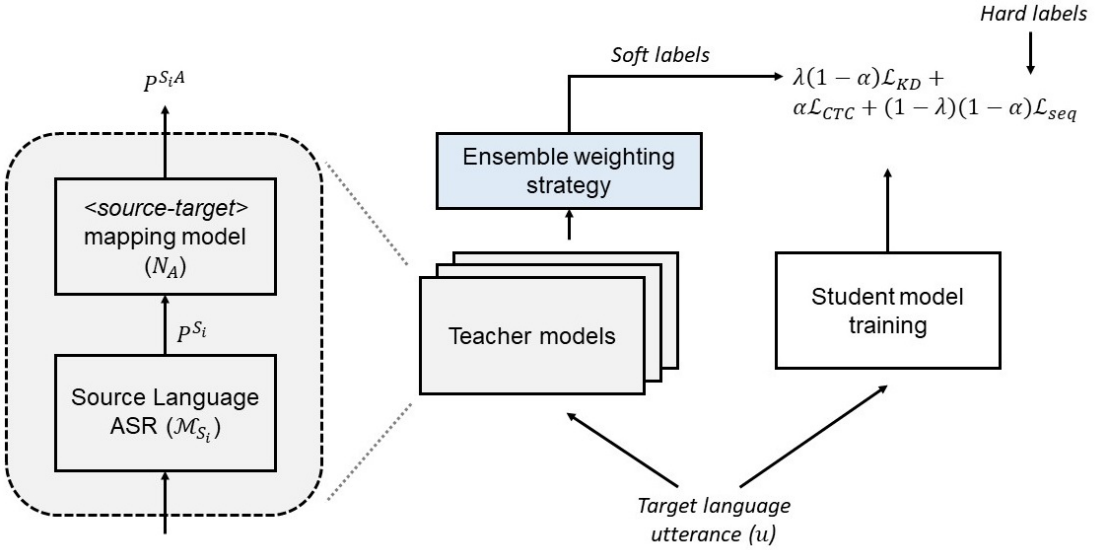


FIGURE 9.1: Architecture of Multilingual Student-Teacher (MUST) learning

TABLE 9.4: Details of Babel data sets used to train e2e speech recognition models

Lang	Train		Eval	
	# hours	# spks	# hours	# spks
Tamil (*tam*)	59.11	372	7.8	61
Telugu (*tel*)	32.94	243	4.97	60
Cebuano (*ceb*)	37.44	239	6.59	60
Javanese (*jav*)	41.15	242	7.96	60

where $\sum_K \mathcal{W} = 1$ and τ is a hyper-parameter for the sake of statistically significant weight distribution across the teachers. Increasing the value of τ increases the deviation of the teachers' weights from the mean weight.

9.3.2 Experimental setup

9.3.2.1 Data set

In this work, all the experiments are conducted using the same data sets as in the previous work on mapping models (as described in Section 7.3.1). Four low-resource languages (Tamil (*tam*), Telugu (*tel*), Cebuano (*ceb*) and Javanese (*jav*)) from the IARPA Babel speech corpus (Gales et al., 2014) with their full language packs are used for ASR training and evaluation. Most of the Babel data sets consist of conversational telephone speech with real-time background noises and are quite challenging because of conversation styles, limited bandwidth, environment conditions and channel. All the utterances without any speech are discarded. The details of the data sets are given in Table 9.4.

For training of the mapping models, a subset of 30 hours is randomly selected from each Babel language pack. This data is further split into 29 hours of train set and 1 hour of dev set.

9.3.3 Student and teacher models

As described earlier, Hybrid CTC/attention architecture (Kim et al., 2017) is used to train all e2e speech recognition models which consists of three modules that are; a shared encoder, an attention decoder and a CTC module. The training process jointly optimises the weighted sum of CTC and attention model as given in Equation 9.9 but $\mathcal{L}_{seq}$ is a cross-entropy loss for the training of teacher models which implies that $\mathcal{L}_{seq}$ in Equation 9.9 is same as $\mathcal{L}'_{seq}$ of Equation 9.10.

The input to the model is 40 filterbanks and the output of the model is byte-pair encoded tokens. All the models are trained for 100 BPE tokens for each language and SentencePiece library (Kudo, 2018) is used for tokenisation. Both student and teacher models are of the same capacity ($\sim$ 170.9 million) throughout the experimentation.

During decoding, the final prediction is made based on a weighted sum of log probabilities from both the CTC and attention components. Given a speech input X, the final

prediction $\hat{Y}$ is given by;

$$\hat{Y} = \arg\max_{Y \in \mathcal{Y}} \{\gamma \log P_{CTC}(Y|X) + (1-\gamma) \log P_{seq}(Y|X)\} \tag{9.14}$$

where γ is a hyper-parameter.

For the speech recognition task, results are reported in terms of percent character error rate. The SpeechBrain toolkit (Ravanelli et al., 2021) is used for training all ASR systems.

9.3.3.1 Mapping models

A multi-encoder single decoder model is trained for each target language. In an MESD model, there are three encoders and only one attention decoder. Each encoder and single decoder consists of one bidirectional RNN layer. For each target language, the mapping model size is only 2.59 million parameters. The performance of these mapping models is measured in terms of accuracy as given in Algorithm 2.

9.3.3.2 MUST learning

For the experimentation in this work, different values of α and λ of Equation 9.9 and 9.10 are varied between the range of $[0, 1]$ and the numbers are reported with the best configuration. The values of α and γ are kept constant for all the experimentation while λ may vary for different languages. For teachers' ensemble, various weighting strategies are experimented to assign the weights ($\mathcal{W}$). Conventional teacher averaging is compared with proposed self-adaptive weighting. In teacher averaging, all the teachers get equal weights and do not change during the whole training. Frame-wise maximum (FWM) selects posteriors from a different teacher for each frame of a given utterance. For each frame, the teacher having a maximum value of posteriors among all the teachers is selected. In domain adaptation experiments, the proposed elitist sampling technique has outperformed TA and FWM weighting strategies for speech recognition domain generalisation task. ES takes a mean of maximum posterior values of all the frames for a given utterance. Then the soft labels of the teacher having the highest value are used for that given utterance.

As discussed in Section 7.4, posterior distributions from all the mapping models have been fused as an acoustic model which has outperformed the monolingual acoustic models. However, the fused weights have been fine-tuned for the test set. An experiment is also done here by assigning fine-tuned weights (FTW) for the test set. These weights are manually fine-tuned and might be a sub-optimal solution. Lastly, a comparison is shown by using only one teacher model for knowledge distillation rather than an ensemble of all the teachers. The objective is to analyse the gap in performance by reducing the teacher models to reduce the computational complexity. In the case of single-teacher (ST) distillation, only the teacher from the closest language is selected. 'Closest' language is defined in terms of mapping model accuracy. For a target language, the source

TABLE 9.5: Accuracy of the pre-trained mapping models from Table 8.4 of Chapter 8

Source Lang.	*Target/Student languages*			
	tam	*tel*	*ceb*	*jav*
tam	-	48.88	60.53	62.24
tel	47.46	-	48.32	54.64
ceb	45.98	46.22	-	65.51
jav	46.97	47.40	65.04	-

language with maximum mapping model accuracy is selected as the teacher model.

9.3.4 Results and Discussion

9.3.4.1 Teacher models

As described in Section 9.3.1, a teacher model for MUST learning is a combination of a teacher language ASR and a student-teacher mapping model. Since most of the languages included in this study have different scripts and character sets, the ASR of a language cannot be used for decoding the data of the other one. Pre-trained mapping models are used for each student-teacher pair in this work. Performance of the pre-trained mapping models is tabulated in Table 9.5 in terms of accuracy. For each target language, the accuracy of the mapping model is shown for all the source-target mapping modules.

9.3.4.2 MUST learning

For multilingual student-teacher learning, various teacher ensemble strategies are explored. Before training a student model, the ensemble strategies are applied for teachers' model fusion. For a given target language, outputs from all the teacher models are fused in a weighted sum. CER is calculated by applying greedy search on fused teacher outputs. All the discussed ensemble strategies (in Section 9.3.3.2) including teacher averaging (TA), frame-wise max (FWM), elitist sampling (ES), self-adaptive weighting (SAW) and fine-tuned weights (FTW) are experimented with and results are tabulated in Table 9.6.

TABLE 9.6: MUST teachers' performance in terms of %CER with different ensemble approaches. *ES*: elitist sampling of Section 9.2.2, *FWM*: frame wise maximum, *SAW*: proposed self-adaptive weighting, *TA*: teacher averaging, *FTW*: manually fine-tuned weights

MUST teachers	*Target/Student languages*				
	tam	*tel*	*ceb*	*jav*	*avg*
ES	57.24	83.23	72.09	75.93	72.12
FWM	57.39	82.54	62.45	70.14	68.13
SAW	57.34	84.31	61.99	67.32	67.74
TA	57.38	84.31	61.98	67.26	67.73
FTW	57.34	83.36	60.03	59.45	65.04

TABLE 9.7: Performance (%CER) of student model trained using MUST learning. *mono*: baseline monolingual model, *ST*: single teacher (the closest language)

MUST teachers	*Target/Student languages*				
	tam	*tel*	*ceb*	*jav*	*avg*
mono	44.28	56.18	31.26	40.90	43.16
TA	44.72	57.02	32.43	42.61	44.20
SAW	44.59	56.14	31.87	39.11	42.93
FTW	44.42	55.79	30.80	37.50	42.13
ST	**43.77**	**55.56**	**29.43**	**36.98**	**41.44**

Analysis shows that the trend of student model performance with different ensemble strategies is the same as the trend for model fusion. So, the student models here are trained using only the top three best-performing teachers' ensemble techniques in Table 9.6 which are SAW, TA and FTW. %CER of student model are shown in Table 9.7. The first row is the %CER from a baseline monolingual ASR using an explicit RNNLM trained on the limited text of train set transcriptions. Although the performance of TA and SAW is almost the same for model fusion (in Table 9.6), student models trained with SAW ensemble reduce average CER by 1.27% relative if compared with the models trained with TA weighting (Table 9.7). With the weights fine-tuned for the test set, average CER is reduced to 42.13% from 42.93% of SAW-trained models which is a relative improvement of 2.4% compared to monolingual models.

In another experiment, knowledge from only a single teacher model is distilled for student model training (ST in Table 9.7). For each target language, the closest language is chosen as a teacher model. As described earlier, the closest source language for a target language is the one which has the highest mapping model accuracy for the target language. Student models trained using the single teacher outperform all other students for all the languages by an average improvement of 4% in the performance of the monolingual model. For *jav* target language, a relative improvement of 9.5% is observed.

Both *ceb* and *jav* yield more gains in performance than *tam* and *tel* because the mapping models' accuracies are higher for these two languages. It is evident that the gain for each language depends directly on the performance of the corresponding mapping model. Student training with ST does not have any test set information and performs even better than FTW which has fine-tuned weights for the test set. The results are in line with the performance of mapping models and the results reported for data augmentation using the mapping models in Section 8.5.2. Since some source-target mapping models do not perform very well for some of the teacher languages, the teacher knowledge introduces noise in student training and makes it hard for a student to learn. Knowledge distillation from only a single student not only improves ASR performance but also reduces computational complexity.

9.3.5 Summary

This chapter discusses two experiments i.e. collaborative work to improve out-of-domain speech recognition using knowledge distillation and bridging a gap in using knowledge distillation approach across the languages. In the work to improve OOD speech recognition, several self-supervised teacher models (wav2vec2.0) have been fine-tuned with speech data from different domains i.e. meetings (AMI) and read speech (LibriSpeech360 and WSJ). Then, a student model is trained in an unsupervised way distilling knowledge from the teachers' models. A novel 'elitist sampling' approach is introduced to select the best utterance among the teacher outputs for the student training. The results have shown that the individual teacher models give a higher error rate on an out-of-domain (telephonic) data set (Switchboard). AMI, LibriSpeech360 and WSJ give a word error rate of 52.04%, 45.86% and 66.43% respectively which is reduced to 51.71% using a frame-wise max ensemble approach to train a student model. A student model trained with the proposed elitist sampling approach reduces the error further to 37.38% without using any explicit language model.

The promising results of OOD work intrigued us to study the feasibility of a knowledge distillation approach for low-resource speech recognition improvement. However, a limitation of knowledge distillation is that the output tokens of the student model must be a proper or improper subset of the output tokens of the teacher models. It is not possible for different languages which have different writing scripts yet acoustically very similar. In this work, we addressed this limitation by employing previously proposed mapping models. In MUST learning, a teacher model is a combination of a source language ASR followed by a source-target mapping model. Pre-trained mapping models are used to map posteriors from a source language ASR to those of the target language ASR (Table 9.5). Various weighting strategies are explored for teachers' ensembles (Table 9.6). Student models are trained for each language with top-performing ensemble strategies. A student model trained with MUST learning proves to outperform baseline monolingual ASR by a relative gain of up to 9.5%.

Chapter 10

Conclusions and Future Work

The work in this thesis has been motivated by the fact that many languages with significant phonemes overlap pose speech recognition performance degradation from multilingual ASR systems. A novel data-driven approach has been devised to estimate cross-lingual acoustic-phonetic similarities. Mapping models have been trained for each source-target languages pair and these models are trained to map the posteriors from source language ASR to the target language ASR. Given a speech signal of a target language, posteriors from the target language acoustic models and mapped posteriors from the source language acoustic models are analysed to study the cross-lingual similarities. The mapping models have also been exploited to propose an acoustic model fusion approach in a multilingual setup to improve the performance of low-resource speech recognition. The approach is also further extended for e2e models. Furthermore, a long-standing gap between using knowledge distillation technique to distil knowledge from diverse languages has also been bridged in this work.

10.1 Contributions

10.1.1 Cross-lingual acoustic-phonetic similarities

In Chapter 4, an analysis has been done as a pretext for this research. The analysis has shown that the multilingual ASR systems might cause to increase phoneme error rate of some languages compared with the monolingual ASR systems. This can happen even in the case where the languages are close. Closeness of languages is usually estimated by the size of overlapping IPA-representations based phonemes of the languages. Furthermore, the analysis has shown that when a speech signal of a target language is decoded through a source language ASR, learnable patterns exist in the outputs from the target language ASR system and the source-language ASR system outputs.

In light of the insights into the existence of learnable cross-lingual mappings, the mapping models have been trained (discussed in detail in Chapter 5) for source-target languages. These mapping models have been used to analyse cross-lingual acoustic phonetic similarities to evaluate if the number of overlapping IPA-representations-based phonemes is

a good criterion to estimate the closeness of languages. The analysis has shown that a language might be acoustically closer to a target language even if has a lesser phoneme overlap compared to some other languages. Phonemes with different IPA representations across languages might be closer acoustically and IPA representation does not well inform on languages' closeness in this case. These properties are well-captured by learnt mapping models and the experiments here have shown that the KL-divergence between mapped posteriors (from a source-target language mapping model) and ground-truth posteriors (from target language ASR) can better inform about language similarities.

10.1.2 Improving low-resource speech recognition

Previous work on monolingual speech recognition has shown that fusing outputs from different models (i.e. models with different architectures for the same downstream task) helps improve speech recognition. Output posterior distributions from different models given an identical input signal can be fused if the posterior distributions are the same (in terms of the set of tokens and their identifiers). However, monolingual models of diverse languages have never been fused in a multilingual setup because all the languages have different token sets (i.e. phonemes or graphemes), and thus the output classes of the monolingual model of each language are different from the others. ASR systems of other languages can be helpful for a low-resource speech recognition system as an additional knowledge source especially when the languages are closer.

In Chapter 7, the mapping models have been leveraged to bridge the gap for the fusion of diverse monolingual models to help improve speech recognition of a target low-resource speech recognition system. A source language ASR has been used to output posterior distributions for a target language input speech signal. These posteriors are then mapped to the target language posterior space using a pre-trained source-target mapping model. Remember that these mapping models have the property to be trained on very limited data. The experiments have shown that the fusion of mapped posteriors along with the posteriors from the target-language acoustic model outperforms multilingual as well as monolingual ASR systems.

The approach does not only work for phoneme-based hybrid DNN-HMM ASR systems but also helps improve the performance of end-to-end ASR models (Chapter 8). Mapping models can also be used to generate transliterated data for augmentation which can be used to retrain a low-resource speech recognition system. Additional experiments in Chapter 8 show that the models retrained with adding augmented data outperform monolingual ASR systems.

10.1.3 Cross-lingual knowledge distillation

Many works have been done where a student model is trained using distilled knowledge from another pre-trained (teacher) model. Knowledge distillation or KD loss is computed as a KL divergence between posteriors from the student and teacher models given an input speech utterance. However, for this loss calculation, the output classes of the

student model must be a proper or improper subset of the teacher model. This prevents a teacher model to distil knowledge for a student if the student is of some other language (even a closer one but with different tokens). In Chapter 9, this gap is bridged through the proposed MUST approach. MUST exploits the mapping models to train student ASR systems using knowledge from ASR systems of diverse languages. The student models trained from MUST have been shown to reduce speech recognition error rates if compared with the monolingual ASR systems.

10.2 Future work

10.2.1 Enhancing mapping model capabilities

In this work, the cross-lingual mapping models have been proposed and their potential has been shown. For several approaches, the performance depends on the accuracy of mapping models and the performance of mapping models is dependent on the baseline acoustic models. The mapping models have been kept quite simple due to several resource limitations and there is room to enhance their capacity to perform even better which would lead to further gains in improving low-resource speech recognition. Furthermore, the mapping models have been trained to map posterior distributions. However, many other ways can be explored and might be helpful i.e. training a sequence mapping model to output target language transcription (rather than posteriors) taking posteriors from a source-language ASR as an input.

10.2.2 Exploiting rich resources for low-resource ASR improvement

For low-resource speech recognition improvement, resources of several other low-resource languages have been exploited. The idea has been to show the capabilities of mapping models trained with limited data. However, the work can be extended to improve low-resource speech recognition using acoustic models of resource-rich languages. In this work, both source and target languages are low-resource languages and hence their acoustic models are also weak. A limited amount of data has been used to train mapping models. If a source language is a high-resource language, its acoustic model would be strong and less prone to error. Incorporating the future work of the last section (Section 10.2.1), a mapping model trained to learn mappings from source posteriors to target text (rather than posteriors) would alleviate the problem of training a target language ASR.

Appendix A

Babel data sets tags

As discussed in Section 3.5.2, the various speech and non-speech events in the recordings have been annotated using various tags in the Babel data set. The Babel data set is used for several experiments in this work from training ASR models to training mapping models on top of them. In this appendix, the different tags used in Babel annotation and their handling for this work are described.

TABLE A.1: Tags used to annotate different speech and non-speech events in the Babel data set

Tag	**Description**
	Non-speech events
sta	on start of medium/loud background noise
lipsmack	lip smacks, tongue clicks
breath	inhalation and exhalation between words, yawning
cough	coughing, throat clearing, sneezing
laugh	laughing, chuckling
click	machine or phone click
ring	telephone ring
dtmf	noise made by pressing telephone keypad
int	any other intermittent foreground noise
no-speech	such as silence or breath
overlap	two speakers speaking simultaneously
prompt	electronic voice/automated recording
male-to-female	male-to-female speaker change during recording
female-to-male	female-to-male speaker change during recording
	Speech events
hes	hesitations in speaking
word	mispronounced word
-word/word-	fragments (mid-word stumbling)
(())	unintelligible (for annotator)
word/word /(())	recording truncation
foreign	words from some non-native language

TABLE A.2: Duration of utterances having the speech or non-speech events tags in Tamil data set of Babel

Tag	**Duration (hours)**
Non-speech events	
sta	38.12
lipsmack	0.66
breath	2.44
cough	0.62
laugh	4.33
click	0.61
ring	0.12
dtmf	0.09
int	7.03
no-speech	83.90
overlap	0.06
prompt	0.07
Non-speech events	
word	1.72
-word/word-	4.6
(())	13.45
word/word /(())	0.34
foreign	7.5

Most of the speech utterances in the Babel data set consist of conversational telephonic speech with a significant portion recorded in noisy environments. So, there are many speech and non-speech events such as cough, laughing, ringing sounds, hesitation in speaking and uttering some foreign language words etc. Such events make data very challenging and noisy. All the tags with their brief descriptions have been tabulated in Table A.1.

As a case study, the Tamil data set in Babel is analysed and the duration of utterances having these tags in their transcriptions is computed. A significant amount of utterances contain these tags. In particular, there are a lot of utterances which do not have any speech at all (consisting of only a 'no-speech' tag). The duration of utterances having each of these tags has been shown in Table A.2. There are utterances of the duration of about 84 hours which do not contain any speech. Many other tags also constitute a large number of utterances. Given the volume of utterances having these tags, all such utterances cannot be simply dropped as it would come with the price of losing a huge amount of data. The same analysis have been done then for other Babel languages' data sets as well and the same magnitude of utterances with these tags have been found.

As a simple rule to handle speech and non-speech events in the Babel data set, it is expected that an ASR model can learn most of the non-speech events except for those which cause confused speech. It includes 'overlap' and 'prompt' tags where the speech of the speaker is overlapped with another speech recording. However, speech events are

TABLE A.3: Handling of speech and non-speech events tags in Babel data sets transcriptions. *Remove tag*: remove tag from annotation and retain the segment, *Remove segment*: drop the utterance from the data set

Tag	**Duration (hours)**
Non-speech events	
sta	Remove tag
lipsmack	Remove tag
breath	Remove tag
cough	Remove tag
laugh	Remove tag
click	Remove tag
ring	Remove tag
dtmf	Remove tag
int	Remove tag
no-speech	Remove tag
overlap	Remove segment
prompt	Remove segment
male-to-female	Remove tag
female-to-male	Remove tag
Non-speech events	
hes	As it is
word	Remove segment
-word/word-	Remove segment
(())	Remove segment
word/word /(())	Remove segment
foreign	Remove segment

a more serious challenge as the annotation tags do not represent what is being spoken. So, the utterances with most of the speech tags are removed from the data set. Only the utterances with 'hes' tags are retained as the Babel lexicons come with annotations of 'hes' tags to corresponding phonemes. The summary of handling of all tags is tabulated in Table A.3.

References

Solomon Teferra Abate, Martha Yifiru Tachbelie, and Tanja Schultz. Multilingual acoustic and language modeling for ethio-semitic languages. In *Proc. Interspeech*, pages 1047–1051, 2020.

Ossama Abdel-Hamid and Hui Jiang. Fast speaker adaptation of hybrid nn/hmm model for speech recognition based on discriminative learning of speaker code. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7942–7946, 2013. doi: 10.1109/ICASSP.2013.6639211.

Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(10):1533–1545, 2014. doi: 10.1109/TASLP.2014.2339736.

Ahmed Hussen Abdelaziz. Comparing fusion models for dnn-based audiovisual continuous speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(3):475–484, 2018.

Rehan Ahmad, Md Asif Jalal, Muhammad Umar Farooq, Anna Ollerenshaw, and Thomas Hain. Towards domain generalisation in asr with elitist sampling and ensemble knowledge distillation. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023. doi: 10.1109/ICASSP49357.2023.10095746.

Dario Amodei, Rishita Anubhai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Jingdong Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Erich Elsen, Jesse Engel, Linxi Fan, Christopher Fougner, Tony Han, Awni Hannun, Billy Jun, Patrick LeGresley, Libby Lin, Sharan Narang, Andrew Ng, Sherjil Ozair, Ryan Prenger, Jonathan Raiman, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Yi Wang, Zhiqian Wang, Chong Wang, Bo Xiao, Dani Yogatama, Jun Zhan, and Zhenyao Zhu. Deep speech 2: End-to-end speech recognition in english and mandarin, 2015.

Guillermo Aradilla, Jithendra Vepa, and Herve Bourlard. An acoustic model based on kullback-leibler divergence for posterior features. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, volume 4, pages IV–657–IV–660, 2007. doi: 10.1109/ICASSP.2007.366998.

Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. Common voice: A massively-multilingual speech corpus. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4218–4222, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4.

Taichi Asami, Ryo Masumura, Yoshikazu Yamaguchi, Hirokazu Masataki, and Yushi Aono. Domain adaptation of dnn acoustic models using knowledge distillation. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5185–5189, 2017. doi: 10.1109/ICASSP.2017.7953145.

International Phonetic Association. International phonetic alphabet. *IPA Handbook*, 1999. URL https://www.internationalphoneticassociation.org/content/ipa-handbook.

Arun Babu, Changhan Wang, Andros Tjandra, Kushal Lakhotia, Qiantong Xu, Naman Goyal, Kritika Singh, Patrick von Platen, Yatharth Saraf, Juan Pino, Alexei Baevski, Alexis Conneau, and Michael Auli. XLS-R: Self-supervised Cross-lingual Speech Representation Learning at Scale. In *Proc. Interspeech 2022*, pages 2278–2282, 2022. doi: 10.21437/Interspeech.2022-143.

A. Baevski, Y. Zhou, A. Mohamed, and M. Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33:12449–12460, 2020.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2016.

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, 1986. doi: 10.1109/ICASSP.1986.1169179.

Laurent Besacier, Etienne Barnard, Alexey Karpov, and Tanja Schultz. Automatic speech recognition for under-resourced languages: A survey. *Speech Communication*, 56:85–100, 2014.

A. Bhuvaneswari, J. Timothy Jones Thomas, and P. Kesavan. Embedded bi-directional gru and lstmlearning models to predict disasterson twitter data. *Procedia Computer Science*, 165:511–516, 2019. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.

2020.01.020. 2nd International Conference on Recent Trends in Advanced Computing ICRTAC -DISRUP - TIV INNOVATION , 2019 November 11-12, 2019.

Jayadev Billa. ISI ASR System for the Low Resource Speech Recognition Challenge for Indian Languages. In *Proc. Interspeech 2018*, pages 3207–3211, 2018. doi: 10.21437/Interspeech.2018-2473.

Hervé A. Bourlard and Nelson Morgan. *The Hybrid HMM/MLP Approach*, pages 155–183. Springer US, Boston, MA, 1994. doi: 10.1007/978-1-4615-3210-1_7.

Lukáš Burget, Petr Schwarz, Mohit Agarwal, Pinar Akyazi, Kai Feng, Arnab Ghoshal, Ondřej Glembek, Nagendra Goel, Martin Karafiát, Daniel Povey, Ariya Rastrow, Richard C. Rose, and Samuel Thomas. Multilingual acoustic modeling for speech recognition based on subspace gaussian mixture models. In *ICASSP*, pages 4334–4337, 2010.

J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, W. Kraaij, M. Kronenthal, G. Lathoud, M. Lincoln, A. Lisowska, I. McCowan, W. Post, D. Reidsma, and P. Wellner. The ami meeting corpus: A pre-announcement. In *Machine Learning for Multimodal Interaction*, pages 28–39. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-32550-5.

Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, Jul 1997. ISSN 1573-0565. doi: 10.1023/A:1007379606734.

William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4960–4964, 2016. doi: 10.1109/ICASSP.2016.7472621.

Xuankai Chang, Wangyou Zhang, Yanmin Qian, Jonathan Le Roux, and Shinji Watanabe. End-to-end multi-speaker speech recognition with transformer, 2020.

Y. Chebotar and A. Waters. Distilling knowledge from ensembles of neural networks for speech recognition. In *Interspeech*, pages 3439–3443, 2016.

Dongpeng Chen and Brian Kan-Wing Mak. Multitask learning of deep neural networks for low-resource speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(7):1172–1183, 2015. doi: 10.1109/TASLP.2015.2422573.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL https://aclanthology.org/D14-1179.

Kwanghee Choi and Hyung-Min Park. Distilling a Pretrained Language Model to a Multilingual ASR Model. In *Proc. Interspeech 2022*, pages 2203–2207, 2022. doi: 10.21437/Interspeech.2022-716.

Jan Chorowski and Navdeep Jaitly. Towards better decoding and language model integration in sequence to sequence models, 2016.

Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. End-to-end continuous speech recognition using attention-based recurrent nn: First results, 2014.

Sharada V Chougule, Mahesh S Chavan, and M S Gaikwad. Filter bank based cepstral features for speaker recognition. In *2014 IEEE Global Conference on Wireless Computing & Networking (GCWCN)*, pages 102–106, 2014. doi: 10.1109/GCWCN.2014.7030857.

Ekapol Chuangsuwanich. *Multilingual Techniques for Low Resource Automatic Speech Recognition*. PhD thesis, 2016.

Alexis Conneau, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed, and Michael Auli. Unsupervised Cross-Lingual Representation Learning for Speech Recognition. In *Proc. Interspeech*, pages 2426–2430, 2021.

Alexis Conneau, Min Ma, Simran Khanuja, Yu Zhang, Vera Axelrod, Siddharth Dalmia, Jason Riesa, Clara Rivera, and Ankur Bapna. Fleurs: Few-shot learning evaluation of universal representations of speech, 2022. URL https://arxiv.org/abs/2205.12446.

Florian Coulmas. *The Blackwell Encyclopedia of Writing Systems*. Blackwell Publishers, Malden, MA, 1996.

Jia Cui, Brian Kingsbury, Bhuvana Ramabhadran, George Saon, Tom Sercu, Kartik Audhkhasi, Abhinav Sethy, Markus Nussbaum-Thom, and Andrew Rosenberg. Knowledge distillation across ensembles of multilingual models for low-resource languages. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4825–4829, 2017.

Xiaodong Cui, Vaibhava Goel, and Brian Kingsbury. Data augmentation for deep neural network acoustic modeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(9):1469–1477, 2015. doi: 10.1109/TASLP.2015.2438544.

Arindrima Datta, Bhuvana Ramabhadran, Jesse Emond, Anjuli Kannan, and Brian Roark. Language-agnostic multilingual modeling. In *ICASSP*, pages 8239–8243, 2020.

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. doi: 10.1109/TASSP.1980.1163420.

Marc Delcroix, Keisuke Kinoshita, Atsunori Ogawa, Takuya Yoshioka, Dung T. Tran, and Tomohiro Nakatani. Context Adaptive Neural Network for Rapid Adaptation of Deep CNN Based Acoustic Models. In *Proc. Interspeech 2016*, pages 1573–1577, 2016. doi: 10.21437/Interspeech.2016-203.

Li Deng and Xuedong Huang. Challenges in adopting speech recognition. *Commun. ACM*, 47(1):69–75, jan 2004. ISSN 0001-0782. URL https://doi.org/10.1145/962081.962108.

John Dines and Mathew Magimai Doss. A study of phoneme and grapheme based context-dependent asr systems. In Andrei Popescu-Belis, Steve Renals, and Hervé Bourlard, editors, *Machine Learning for Multimodal Interaction*, pages 215–226, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. ISBN 978-3-540-78155-4.

Jesse Emond, Bhuvana Ramabhadran, Brian Roark, Pedro Moreno, and Min Ma. Transliteration based approaches to improve code-switched speech recognition performance. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 448–455, 2018. doi: 10.1109/SLT.2018.8639699.

Ethnologue. Languages of the world. https://www.ethnologue.com/guides/how-many-languagesm. Accessed: 2022-03-27.

G. Evermann and P.C. Woodland. Large vocabulary decoding and confidence estimation using word posterior probabilities. In *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.00CH37100)*, volume 3, pages 1655–1658 vol.3, 2000. doi: 10.1109/ICASSP.2000.862067.

Gongfan Fang, Yifan Bao, Jie Song, Xinchao Wang, Donglin Xie, Chengchao Shen, and Mingli Song. Mosaicking to distill: Knowledge distillation from out-of-domain data. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 11920–11932. Curran Associates, Inc., 2021.

Muhammad Umar Farooq and Thoams Hain. Learning Cross-lingual Mappings for Data Augmentation to Improve Low-Resource Speech Recognition. In *Proc. Interspeech 2023*, 2023.

Muhammad Umar Farooq and Thomas Hain. Investigating the Impact of Crosslingual Acoustic-Phonetic Similarities on Multilingual Speech Recognition. In *Proc. Interspeech 2022*, pages 3849–3853, 2022. doi: 10.21437/Interspeech.2022-10916.

Muhammad Umar Farooq, Farah Adeeba, Sarmad Hussain, Sahar Rauf, and Maryam Khalid. Enhancing large vocabulary continuous speech recognition system for urdu-english conversational code-switched speech. In *IEEE O-COCOSDA*, pages 155–159, 2020.

Muhammad Umar Farooq, Darshan Adiga Haniya Narayana, and Thomas Hain. Non-linear pairwise language mappings for low-resource multilingual acoustic model fusion, 2022.

Muhammad Umar Farooq, Rehan Ahmad, and Thomas Hain. MUST: A Multilingual Student-Teacher Learning Approach for Low-Resource Speech Recognition. In *ASRU*, 2023.

Siyuan Feng, Piotr Żelasko, Laureano Moro-Velázquez, Ali Abavisani, Mark Hasegawa-Johnson, Odette Scharenborg, and Najim Dehak. How phonotactics affect multilingual and zero-shot asr performance. In *ICASSP*, pages 7238–7242, 2021.

W. M. Fisher, G. Doddington, and K. Goudie-Marshall. The darpa speech recognition research database: specifications and status. In *Proceedings of the DARPA Workshop on Speech Recognition*, pages 93–99, 1986.

David Freedman and Persi Diaconis. On the histogram as a density estimator:l2 theory. *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete*, 57(4):453–476, Dec 1981. ISSN 1432-2064. doi: 10.1007/BF01025868. URL https://doi.org/10.1007/BF01025868.

T. Fukuda, M. Suzuki, G. Kurata, S. Thomas, J. Cui, and B. Ramabhadran. Efficient knowledge distillation from an ensemble of teachers. In *Interspeech*, pages 3697–3701, 2017.

Takashi Fukuda and Samuel Thomas. Knowledge Distillation Based Training of Universal ASR Source Models for Cross-Lingual Transfer. In *Proc. Interspeech 2021*, pages 3450–3454, 2021. doi: 10.21437/Interspeech.2021-796.

H. Futami, H. Inaguma, S. Ueno, M. Mimura, S. Sakai, and T. Kawahara. Distilling the knowledge of bert for sequence-to-sequence asr. *arXiv preprint arXiv:2008.03822*, 2020.

M. J. F. Gales, A. Ragni, H. AlDamarki, and C. Gautier. Support vector machines for noise robust asr. In *2009 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 205–210, 2009.

Mark J. F. Gales, Kate M. Knill, Anton Ragni, and Shakti P. Rath. Speech recognition and keyword spotting for low-resource languages: Babel project research at CUED. In *Workshop on Spoken Language Technologies for Under-Resourced Languages (SLTU)*, pages 16–23, 2014.

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

M.J.F. Gales. Semi-tied covariance matrices for hidden markov models. *IEEE Transactions on Speech and Audio Processing*, 7(3):272–281, 1999. doi: 10.1109/89.759034.

Y. Gao, T. Parcollet, and N. D Lane. Distilling knowledge from ensembles of acoustic models for joint ctc-attention end-to-end speech recognition. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 138–145. IEEE, 2021.

Neeraj Gaur, Brian Farris, Parisa Haghani, Isabel Leal, Pedro J. Moreno, Manasa Prasad, Bhuvana Ramabhadran, and Yun Zhu. Mixture of informed experts for multilingual speech recognition. In *ICASSP*, pages 6234–6238, 2021.

J.L. Gauvain, L.F. Lamel, G. Adda, and M. Adda-Decker. The LIMSI continuous speech dictation systemt. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*, 1994. URL https://aclanthology.org/H94-1064.

Roberto Gemello, Franco Mana, Stefano Scanzio, Pietro Laface, and Renato De Mori. Linear hidden transformations for adaptation of hybrid ann/hmm models. *Speech Communication*, 49(10):827–835, 2007. URL https://www.sciencedirect.com/science/article/pii/S0167639306001786.

Abraham P. George and Warrren B Powell. Adaptive stepsizes for recursive estimation with applications in approximate dynamic programming. *Machine Learning*, 65:167–198, 2006.

Arnab Ghoshal, Pawel Swietojanski, and Steve Renals. Multilingual training of deep neural networks. In *ICASSP*, pages 7319–7323, 2013.

James Glass, Giovanni Flammia, David Goodine, Michael Phillips, Joseph Polifroni, Shinsuke Sakai, Stephanie Seneff, and Victor Zue. Multilingual spoken-language understanding in the mit voyager system. *Speech Communication*, 17(1):1–18, 1995. ISSN 0167-6393. doi: https://doi.org/10.1016/0167-6393(95)00008-C.

J.J. Godfrey, E.C. Holliman, and J. McDaniel. Switchboard: telephone speech corpus for research and development. In *IEEE ICASSP*, volume 1, pages 517–520 vol.1, 1992. doi: 10.1109/ICASSP.1992.225858.

Zhuo Gong, Daisuke Saito, Sheng Li, Hisashi Kawai, and Nobuaki Minematsu. Can we train a language model inside an end-to-end ASR model? - investigating effective implicit language modeling. In Xianchao Wu, Peiying Ruan, Sheng Li, and Yi Dong, editors, *Proceedings of the Second Workshop on When Creative AI Meets Conversational AI*, pages 42–47, Gyeongju, Republic of Korea, October 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.cai-1.6.

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 vol.2, 1998. doi: 10.1109/ICASSP.1998.675351.

J. Gou, B. Yu, S. J Maybank, and D. Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzman, and Angela Fan. The flores-101 evaluation benchmark for low-resource and multilingual machine translation, 2021. URL https://arxiv.org/abs/2106.03193.

A. Graves. Connectionist temporal classification. In *Supervised sequence labelling with recurrent neural networks*, pages 61–93. Springer, 2012.

Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1764–1772, Bejing, China, 22–24 Jun 2014. PMLR.

Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610, 2005. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2005.06.042. IJCNN 2005.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 369–376, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933832. doi: 10.1145/1143844.1143891. URL https://doi.org/10.1145/1143844.1143891.

Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. Hybrid speech recognition with deep bidirectional lstm. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 273–278, 2013a. doi: 10.1109/ASRU.2013.6707742.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks, 2013b.

Frantisek Grézl, Martin Karafiát, and Karel Veselý. Adaptation of multilingual stacked bottle-neck neural network structure for new language. In *ICASSP*, pages 7654–7658, 2014.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O.K. Li, and Richard Socher. Non-autoregressive neural machine translation. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1l8BtlCb.

Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. Deep speech: Scaling up end-to-end speech recognition, 2014.

William J. Hardcastle, John Laver, and Fiona E. Gibbon. *The Handbook of Phonetic Sciences*. Wiley-Blackwell, 2010.

Mark Hasegawa-Johnson, Leanne Rolston, Camille Goudeseune, Gina-Anne Levow, and Katrin Kirchhoff. Grapheme-to-phoneme transduction for cross-language asr. In Luis Espinosa-Anke, Carlos Martín-Vide, and Irena Spasić, editors, *Statistical Language and Speech Processing*, pages 3–19. Springer International Publishing, 2020.

G. Heigold, V. Vanhoucke, A. Senior, P. Nguyen, M. Ranzato, M. Devin, and J. Dean. Multilingual acoustic models using distributed deep neural networks. In *ICASSP*, pages 8619–8623, 2013.

Hynek Hermansky. Perceptual linear predictive (PLP) analysis of speech. *The Journal of the Acoustical Society of America*, 87(4):1738–1752, 04 1990. ISSN 0001-4966. doi: 10.1121/1.399423.

Wolfgang J. Hess, Klaus J. Kohler, and Hans-Günther Tillmann. The Phondat-verbmobil speech corpus. In *Proc. 4th European Conference on Speech Communication and Technology (Eurospeech 1995)*, pages 863–866, 1995. doi: 10.21437/Eurospeech.1995-197.

G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015. URL http://arxiv.org/abs/1503.02531.

Geoffrey Hinton, Li Deng, Dong Yu, George E. Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and Brian Kingsbury. 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. doi: 10.1109/MSP.2012.2205597.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Takaaki Hori, Shinji Watanabe, Yu Zhang, and William Chan. Advances in Joint CTC-Attention Based End-to-End Speech Recognition with a Deep CNN Encoder and RNN-LM. In *Proc. Interspeech 2017*, pages 949–953, 2017. doi: 10.21437/Interspeech.2017-1296.

Takaaki Hori, Jaejin Cho, and Shinji Watanabe. End-to-end speech recognition with word-based rnn language models. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 389–396, 2018. doi: 10.1109/SLT.2018.8639693.

Wenxin Hou, Yue Dong, Bairong Zhuang, Longfei Yang, Jiatong Shi, and Takahiro Shinozaki. Large-Scale End-to-End Multilingual Speech Recognition and Language Identification with Multi-Task Learning. In *Proc. Interspeech*, pages 1037–1041, 2020.

Jui-Yang Hsu, Yuan-Jui Chen, and Hung-yi Lee. Meta learning for end-to-end low-resource speech recognition. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7844–7848, 2020. doi: 10.1109/ICASSP40776.2020.9053112.

W.-N. Hsu, A. Sriram, A. Baevski, T. Likhomanenko, Q. Xu, V. Pratap, J. Kahn, A. Lee, R. Collobert, G. Synnaeve, and M. Auli. Robust wav2vec 2.0: Analyzing domain shift in self-supervised pre-training. *arXiv preprint arXiv:2104.01027*, 2021a.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units, 2021b.

Jui-Ting Huang, Jinyu Li, Dong Yu, Li Deng, and Yifan Gong. Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers. In *ICASSP*, pages 7304–7308, 2013.

Kuan Po Huang, Tzu-Hsun Feng, Yu-Kuan Fu, Tsu-Yuan Hsu, Po-Chieh Yen, Wei-Cheng Tseng, Kai-Wei Chang, and Hung-Yi Lee. Ensemble knowledge distillation of self-supervised speech models. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023. doi: 10.1109/ICASSP49357.2023.10096445.

M. Huang, Y. You, Z. Chen, Y. Qian, and K. Yu. Knowledge distillation for sequence model. In *Interspeech*, pages 3703–3707, 2018.

D H Hubel and T N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *J Physiol*, 160(1):106–154, January 1962.

B. Imperl, Z. Kacic, B. Horvat, and A. Zgank. Agglomerative vs. tree-based clustering for the definition of multilingual set of triphones. In *ICASSP*, volume 3, pages 1273–1276 vol.3, 2000.

David Imseng, Petr Motlicek, Herve Bourlard, and Philip Garner. Using out-of-language data to improve an under-resourced speech recognizer. *Speech Communication*, 56: 142–151, 01 2014.

Navdeep Jaitly and Geoffrey E Hinton. Vocal tract length perturbation (vtlp) improves speech recognition. In *Proc. ICML Workshop on Deep Learning for Audio, Speech and Language*, volume 117, page 21, 2013.

Daniel Jones. The history and meaning of the term "phoneme". *Le Maître Phonétique*, 35 (72):1–20, 1957. ISSN 1016832X. URL http://www.jstor.org/stable/44705495.

J. M. Joyce. *Kullback-Leibler Divergence*, pages 720–722. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-04898-2. doi: 10.1007/978-3-642-04898-2_327. URL https://doi.org/10.1007/978-3-642-04898-2_327.

Dan Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, chapter Automatic Speech Recognition and Text-to-Speech. Prentice Hall, 2008a.

Dan Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, chapter N-gram Language Models. Prentice Hall, 2008b.

Janez Kaiser, Bogomir Horvat, and Zdravko Kačič. Overall risk criterion estimation of hidden markov model parameters. *Speech Communication*, 38(3):383–398, 2002. ISSN 0167-6393. doi: https://doi.org/10.1016/S0167-6393(02)00009-2.

Anjuli Kannan, Arindrima Datta, Tara N. Sainath, Eugene Weinstein, Bhuvana Ramabhadran, Yonghui Wu, Ankur Bapna, Zhifeng Chen, and Seungji Lee. Large-scale multilingual speech recognition with a streaming end-to-end model, 2019.

S. Kanthak and H. Ney. Context-dependent acoustic modeling using graphemes for large vocabulary speech recognition. In *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages I–845–I–848, 2002. doi: 10.1109/ICASSP.2002.5743871.

Martin Karafiát, Murali Karthick Baskar, Pavel Matějka, Karel Veselý, František Grézl, and Jan Černocky. Multilingual blstm and speaker-specific vector adaptation in 2016 but babel system. In *Spoken Language Technology Workshop (SLT)*, pages 637–643, 2016.

Mirjam Killer, Sebastian Stuker, and Tanja Schultz. Grapheme based speech recognition. In *Proc. 8th European Conference on Speech Communication and Technology (Eurospeech 2003)*, pages 3141–3144, 2003. doi: 10.21437/Eurospeech.2003-785.

Byeonggeun Kim, Seunghan Yang, Jangho Kim, and Simyung Chang. Domain generalization on efficient acoustic scene classification using residual normalization, 2021.

Ho-Gyeong Kim, Hwidong Na, Hoshik Lee, Jihyun Lee, Tae Gyoon Kang, Min-Joong Lee, and Young Sang Choi. Knowledge distillation using output errors for self-attention end-to-end models. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6181–6185, 2019. doi: 10.1109/ICASSP.2019.8682775.

Suyoun Kim, Takaaki Hori, and Shinji Watanabe. Joint ctc-attention based end-to-end speech recognition using multi-task learning. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4835–4839, 2017. doi: 10.1109/ICASSP.2017.7953075.

Y. Kim and A. M Rush. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*, 2016.

Ondrej Klejch, Electra Wallington, and Peter Bell. Deciphering Speech: a Zero-Resource Approach to Cross-Lingual Transfer in ASR. In *Proc. Interspeech 2022*, pages 2288–2292, 2022. doi: 10.21437/Interspeech.2022-10170.

Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. Audio augmentation for speech recognition. In *Proc. Interspeech 2015*, pages 3586–3589, 2015. doi: 10.21437/Interspeech.2015-711.

J. Kohler. Multi-lingual phoneme recognition exploiting acoustic-phonetic similarities of sounds. In *Proceeding of Fourth International Conference on Spoken Language Processing (ICSLP)*, volume 4, pages 2195–2198 vol.4, 1996.

Krunoslav Kovac. Multitask learning for bayesian neural networks. Master's thesis, University of Toronto, 2005.

Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates, 2018.

S. Kullback and R. A. Leibler. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 22(1):79 – 86, 1951.

G. Kurata and K. Audhkhasi. Improved knowledge distillation from bi-directional to uni-directional lstm ctc for end-to-end speech recognition. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 411–417, 2018. doi: 10.1109/SLT.2018.8639629.

Ludwig Kürzinger, Dominik Winkelbauer, Lujun Li, Tobias Watzel, and Gerhard Rigoll. *CTC-Segmentation of Large Corpora for German End-to-End Speech Recognition*, page 267–278. Springer International Publishing, 2020. ISBN 9783030602765. doi: 10.1007/978-3-030-60276-5_27. URL http://dx.doi.org/10.1007/978-3-030-60276-5_27.

Peter Ladefoged. *The Sounds of the World's Languages*. Blackwell Publishers, Oxford, UK, 1996.

Lori Lamel, M. Adda-Decker, and Jean-Luc Gauvain. Issues in large vocabulary, multilingual speech recognition. In *Proc. 4th European Conference on Speech Communication and Technology (Eurospeech 1995)*, pages 185–188, 1995. doi: 10.21437/Eurospeech.1995-46.

Lori F. Larnel, Jean-Luc Gauvain, and Maxine Eskenazi. BREF, a large vocabulary spoken corpus for French. In *Proc. 2nd European Conference on Speech Communication and Technology (Eurospeech 1991)*, pages 505–508, 1991. doi: 10.21437/Eurospeech.1991-126.

Viet Bac Le, L. Besacier, and T. Schultz. Acoustic-phonetic unit similarities for context dependent acoustic model portability. In *ICASSP*, volume 1, 2006.

Isabel Leal, Neeraj Gaur, Parisa Haghani, Brian Farris, Pedro J. Moreno, Manasa Prasad, Bhuvana Ramabhadran, and Yun Zhu. Self-Adaptive Distillation for Multilingual Speech Recognition: Leveraging Student Independence. In *Proc. Interspeech 2021*, pages 2556–2560, 2021. doi: 10.21437/Interspeech.2021-614.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Hung-Shin Lee, Pin-Yuan Chen, Yao-Fei Cheng, Yu Tsao, and Hsin-Min Wang. Speech-enhanced and noise-aware networks for robust speech recognition, 2022.

Bo Li, Ruoming Pang, Yu Zhang, Tara N. Sainath, Trevor Strohman, Parisa Haghani, Yun Zhu, Brian Farris, Neeraj Gaur, and Manasa Prasad. Massively multilingual asr: A lifelong learning solution. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6397–6401, 2022a.

Jie Li, Xiaorui Wang, and Bo Xu. An empirical study of multilingual and low-resource spoken term detection using deep neural networks. In *Proc. Interspeech 2014*, pages 1747–1751, 2014a. doi: 10.21437/Interspeech.2014-399.

Jinyu Li, Rui Zhao, Jui-Ting Huang, and Yifan Gong. Learning small-size dnn with output-distribution-based criteria. In *Interspeech*, September 2014b.

Qiujia Li. *Attention-Based Encoder-Decoder Models for Speech Processing*. PhD thesis, Apollo - University of Cambridge Repository, 2022a. URL https://www.repository.cam.ac.uk/handle/1810/341636.

Qiujia Li. Attention-based encoder-decoder models for speech processing. 2022b. doi: 10.17863/CAM.89062. URL https://www.repository.cam.ac.uk/handle/1810/341636.

Xinjian Li, Siddharth Dalmia, Juncheng Li, Matthew Lee, Patrick Littell, Jiali Yao, Antonios Anastasopoulos, David R. Mortensen, Graham Neubig, Alan W Black, and Florian Metze. Universal phone recognition with a multilingual allophone system. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8249–8253, 2020. doi: 10.1109/ICASSP40776.2020.9054362.

Xinjian Li, Florian Metze, David R. Mortensen, Alan W Black, and Shinji Watanabe. ASR2K: Speech Recognition for Around 2000 Languages without Audio. In *Proc. Interspeech 2022*, pages 4885–4889, 2022b. doi: 10.21437/Interspeech.2022-10712.

Tatiana Likhomanenko, Qiantong Xu, Vineel Pratap, Paden Tomasello, Jacob Kahn, Gilad Avidov, Ronan Collobert, and Gabriel Synnaeve. Rethinking Evaluation in ASR: Are Our Models Robust Enough? In *Proc. Interspeech 2021*, pages 311–315, 2021. doi: 10.21437/Interspeech.2021-1758.

Hui Lin, Li Deng, Dong Yu, Yi-fan Gong, Alex Acero, and Chin-Hui Lee. A study on multilingual acoustic modeling for large vocabulary asr. In *ICASSP*, pages 4333–4336, 2009.

Liang Lu, Xingxing Zhang, and Steve Renais. On training the recurrent neural network encoder-decoder for large vocabulary end-to-end speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5060–5064, 2016. doi: 10.1109/ICASSP.2016.7472641.

Lu Lu. Dying relu and initialization: Theory and numerical examples. *Communications in Computational Physics*, 28(5):1671–1706, June 2020. ISSN 1991-7120. doi: 10.4208/cicp.oa-2020-0165. URL http://dx.doi.org/10.4208/cicp.OA-2020-0165.

Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In Lluís Màrquez, Chris Callison-Burch, and Jian Su, editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1166. URL https://aclanthology.org/D15-1166.

Srikanth Madikeri, Banriskhem K. Khonglah, Sibo Tong, Petr Motlicek, Hervé Bourlard, and Daniel Povey. Lattice-Free Maximum Mutual Information Training of Multilingual Speech Recognition Systems. In *Proc. Interspeech*, pages 4746–4750, 2020.

Sri Harish Mallidi and Hynek Hermansky. Novel neural network based fusion for multi-stream asr. In *ICASSP*, pages 5680–5684, 2016.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. volume 24 of *Psychology of Learning and Motivation*, pages 109–165. Academic Press, 1989. doi: https://doi.org/10.1016/S0079-7421(08)60536-8.

Erik McDermott, Hasim Sak, and Ehsan Variani. A density ratio approach to language model fusion in end-to-end automatic speech recognition. In *ASRU*, pages 434–441, 2019.

Zhong Meng, Jinyu Li, Yashesh Gaur, and Yifan Gong. Domain adaptation via teacher-student learning for end-to-end speech recognition. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 268–275, 2019. doi: 10.1109/ASRU46091.2019.9003776.

Brian C. J. Moore. *An Introduction to the Psychology of Hearing*. Brill, 6th edition, 2014. ISBN 978-9004252420.

Roger K. Moore and Ricard Marxer. Progress and Prospects for Spoken Language Technology: Results from Five Sexennial Surveys. In *Proc. INTERSPEECH 2023*, pages 401–405, 2023. doi: 10.21437/Interspeech.2023-235.

Steven Moran and Daniel McCloy, editors. *PHOIBLE 2.0*. Max Planck Institute for the Science of Human History, Jena, 2019. URL https://phoible.org/.

Mahir Morshed and Mark Hasegawa-Johnson. Cross-lingual articulatory feature information transfer for speech recognition using recurrent progressive neural networks. In *Proc. Interspeech 2022*, pages 2298–2302, 2022. doi: 10.21437/Interspeech.2022-11202.

David R. Mortensen, Siddharth Dalmia, and Patrick Littell. Epitran: Precision G2P for many languages. In *International Conference on Language Resources and Evaluation (LREC)*, May 2018.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807–814, Madison, WI, USA, 2010. Omnipress. ISBN 9781605589077.

Preetum Nakkiran. Learning rate annealing can provably help generalization, even for convex problems. *arXiv preprint arXiv:2005.07360*, 2020.

Joao Neto, Luís Almeida, Mike Hochberg, Ciro Martins, Luis Nunes, Steve Renals, and Tony Robinson. Speaker-adaptation for hybrid hmm-ann continuous speech recognition system. 1995.

Josef R. Novak, Nobuaki Minematsu, and Keikichi Hirose. WFST-based grapheme-to-phoneme conversion: Open source tools for alignment, model-building and decoding. In *International Workshop on Finite State Methods and Natural Language Processing*, pages 45–49, 2012.

Online. Common voice – datasets. https://commonvoice.mozilla.org/en/datasets. Accessed: 2024-01-01.

V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. Librispeech: An asr corpus based on public domain audio books. In *IEEE ICASSP*, pages 5206–5210, 2015. doi: 10.1109/ICASSP.2015.7178964.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition. In *Proc. Interspeech 2019*, pages 2613–2617, 2019. doi: 10.21437/Interspeech.2019-2680. URL http://dx.doi.org/10.21437/Interspeech.2019-2680.

Sree Hari Krishnan Parthasarathi, Björn Hoffmeister, Spyros Matsoukas, Arindam Mandal, Nikko Ström, and Sri Garimella. fmllr based feature-space speaker adaptation of dnn acoustic models. In *Interspeech 2015*, 2015. URL https://www.amazon.science/publications/fmllr-based-feature-space-speaker-adaptation-of-dnn-acoustic-models.

Neil Patel, Sheetal Agarwal, Nitendra Rajput, Amit Nanavati, Paresh Dave, and Tapan S. Parikh. *A Comparative Study of Speech and Dialed Input Voice Interfaces in Rural India*, page 51–54. Association for Computing Machinery, 2009.

Douglas B. Paul and Janet M. Baker. The design for the Wall Street Journal-based CSR corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*, 1992. URL https://aclanthology.org/H92-1073.

Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. A time delay neural network architecture for efficient modeling of long temporal contexts. In *Proc. Interspeech 2015*, pages 3214–3218, 2015. doi: 10.21437/Interspeech.2015-647.

Daniel Povey. Discriminative training for large vocabulary speech recognition. 01 2003.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. The kaldi speech recognition toolkit. In *ASRU*. IEEE Signal Processing Society, 2011.

Daniel Povey, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahremani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur. Purely Sequence-Trained Neural Networks for ASR Based on Lattice-Free MMI. In *Proc. Interspeech 2016*, pages 2751–2755, 2016.

Daniel Povey, Gaofeng Cheng, Yiming Wang, Ke Li, Hainan Xu, Mahsa Yarmohammadi, and Sanjeev Khudanpur. Semi-Orthogonal Low-Rank Matrix Factorization for Deep Neural Networks. In *Proc. Interspeech*, pages 3743–3747, 2018.

Rohit Prabhavalkar, Takaaki Hori, Tara N. Sainath, Ralf Schlüter, and Shinji Watanabe. End-to-end speech recognition: A survey, 2023.

Vineel Pratap, Anuroop Sriram, Paden Tomasello, Awni Hannun, Vitaliy Liptchinsky, Gabriel Synnaeve, and Ronan Collobert. Massively Multilingual ASR: 50 Languages, 1 Model, 1 Billion Parameters. In *Proc. Interspeech*, pages 4751–4755, 2020a.

Vineel Pratap, Qiantong Xu, Anuroop Sriram, Gabriel Synnaeve, and Ronan Collobert. Mls: A large-scale multilingual dataset for speech research. *ArXiv*, abs/2012.03411, 2020b.

L. Rabiner and B. Juang. An introduction to hidden markov models. *IEEE ASSP Magazine*, 3(1):4–16, 1986. doi: 10.1109/MASSP.1986.1165342.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision, 2022.

Anton Ragni, Kate M. Knill, Shakti P. Rath, and Mark J. F. Gales. Data augmentation for low resource languages. In *Proc. Interspeech 2014*, pages 810–814, 2014. doi: 10.21437/Interspeech.2014-207.

Mirco Ravanelli, Philemon Brakel, Maurizio Omologo, and Yoshua Bengio. Light gated recurrent units for speech recognition. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(2):92–102, April 2018. ISSN 2471-285X. doi: 10.1109/tetci.2017.2762739. URL http://dx.doi.org/10.1109/TETCI.2017.2762739.

Mirco Ravanelli, Titouan Parcollet, Peter Plantinga, Aku Rouhe, Samuele Cornell, Loren Lugosch, Cem Subakan, Nauman Dawalatabad, Abdelwahab Heba, Jianyuan Zhong, Ju-Chieh Chou, Sung-Lin Yeh, Szu-Wei Fu, Chien-Feng Liao, Elena Rastorgueva, François Grondin, William Aris, Hwidong Na, Yan Gao, Renato De Mori, and Yoshua Bengio. Speechbrain: A general-purpose speech toolkit, 2021.

Ilyes Rebai, Yessine BenAyed, Walid Mahdi, and Jean-Pierre Lorré. Improving speech recognition using data augmentation and acoustic model fusion. *Procedia Computer Science*, 112:316–322, 2017.

Steve Renals and Simon King. *Automatic Speech Recognition*, chapter 22, pages 804–838. John Wiley & Sons, Ltd, 2010. ISBN 9781444317251.

T. Robinson, J. Fransen, D. Pye, J. Foote, and S. Renals. Wsjcamo: a british english speech corpus for large vocabulary continuous speech recognition. In *1995 International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 81–84 vol.1, 1995. doi: 10.1109/ICASSP.1995.479278.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets, 2015.

Ewa Roszkowska. Rank ordering criteria weighting methods – a comparative overview. *Optimum. Studia Ekonomiczne*, 5(65):14–33, 2013. ISSN 1506-7637.

Mickael Rouvier, Grégor Dupuy, Paul Gay, Elie Khoury, Teva Merlin, and Sylvain Meignier. An open-source state-of-the-art toolbox for broadcast news diarization. In *Proc. Interspeech 2013*, pages 1477–1481, 2013. doi: 10.21437/Interspeech.2013-383.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, Oct 1986. ISSN 1476-4687. doi: 10.1038/323533a0. URL https://doi.org/10.1038/323533a0.

Omer Sagi and Lior Rokach. Ensemble learning: A survey. *WIREs Data Mining and Knowledge Discovery*, 8(4):e1249, 2018. doi: https://doi.org/10.1002/widm.1249.

Hardik B. Sailor and Thomas Hain. Multilingual Speech Recognition Using Language-Specific Phoneme Recognition as Auxiliary Task for Indian Languages. In *Proc. Interspeech*, pages 4756–4760, 2020.

Tanja Schultz. Globalphone: a multilingual speech and text database developed at karlsruhe university. In *Proc. 7th International Conference on Spoken Language Processing (ICSLP 2002)*, pages 345–348, 2002. doi: 10.21437/ICSLP.2002-151.

Tanja Schultz and Katrin Kirchhoff. Chapter 1 - introduction. In Schultz and Kirchhoff, editors, *Multilingual Speech Processing*, pages 1–4. Academic Press, 2006a.

Tanja Schultz and Katrin Kirchhoff. Chapter 4 - multilingual acoustic modeling. In Schultz and Kirchhoff, editors, *Multilingual Speech Processing*, pages 71–122. Academic Press, 2006b.

Tanja Schultz and Katrin Kirchhoff. Chapter 5 - multilingual dictionaries. In Schultz and Kirchhoff, editors, *Multilingual Speech Processing*, pages 123–168. Academic Press, 2006c.

Tanja Schultz and Alex Waibel. Language-independent and language-adaptive acoustic modeling for speech recognition. *Speech Commun.*, 35(1–2):31–51, 2001.

Tanja Schultz, Ngoc Thang Vu, and Tim Schlippe. Globalphone: A multilingual text & speech database in 20 languages. In *ICASSP*, pages 8126–8130, 2013.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units, 2016.

Changhao Shan, Chao Weng, Guangsen Wang, Dan Su, Min Luo, Dong Yu, and Lei Xie. Component fusion: Learning replaceable language model component for end-to-end speech recognition system. In *ICASSP*, pages 5361–5635, 2019.

C.E. Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1): 10–21, 1949. doi: 10.1109/JRPROC.1949.232969.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, 2017.

Zhijie Shen, Wu Guo, and Bin Gu. Language-universal adapter learning with knowledge distillation for end-to-end multilingual speech recognition, 2023.

J. Sherwani, N. Ali, S. Mirza, A. Fatma, Y. Memon, M. Karim, R. Tongia, and R. Rosenfeld. Healthline: Speech-based access to health information by low-literate users. In *International Conference on Information and Communication Technologies and Development*, pages 1–9, 2007.

Vishwas M. Shetty and Metilda Sagaya Mary N.J. Improving the performance of transformer based low resource speech recognition for indian languages. In *ICASSP*, pages 8279–8283, 2020.

Leslie N. Smith. Cyclical learning rates for training neural networks, 2015.

Samuel L. Smith, Pieter-Jan Kindermans, Chris Ying, and Quoc V. Le. Don't decay the learning rate, increase the batch size, 2017.

T F Smith and M S Waterman. Identification of common molecular subsequences. " *Journal of Molecular Biology,*, 147(1):195–197, 1981.

John S. Sobolewski. Data transmission media. In Robert A. Meyers, editor, *Encyclopedia of Physical Science and Technology (Third Edition)*, pages 277–303. Academic Press, New York, third edition edition, 2003. ISBN 978-0-12-227410-7. doi: https://doi.org/10.1016/B0-12-227410-5/00165-4.

American Mathematical Society. *Structure of Language and Its Mathematical Aspects*. Number v. 12; v. 20 in Proceedings of symposia in applied mathematics. American Mathematical Society, 1961. ISBN 9780821813126. URL https://books.google.co.uk/books?id=w0vHCQAAQBAJ.

Kenneth N. Stevens. On the quantal nature of speech. *Journal of Phonetics*, 17(1):3–45, 1989.

Andreas Stolcke. Srilm – an extensible language modeling toolkit. In *International Conference on Spoken Language Processing (ICSLP)*, pages 901–904, 2002.

Pawel Swietojanski and Steve Renals. Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 171–176, 2014a. doi: 10.1109/SLT.2014.7078569.

Pawel Swietojanski and Steve Renals. Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 171–176, 2014b. doi: 10.1109/SLT.2014.7078569.

Martha Yifiru Tachbelie, Solomon Teferra Abate, and Tanja Schultz. Development of multilingual asr using globalphone for less-resourced languages: The case of ethiopian languages. In *Proc. Interspeech*, pages 1032–1036, 2020a.

Martha Yifiru Tachbelie, Ayimunishagu Abulimiti, Solomon Teferra Abate, and Tanja Schultz. Dnn-based speech recognition for globalphone languages. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8269–8273, 2020b. doi: 10.1109/ICASSP40776.2020.9053144.

R. Takashima, S. Li, and H. Kawai. An investigation of a knowledge distillation method for ctc acoustic models. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5809–5813. IEEE, 2018.

Xu Tan, Yi Ren, Di He, Tao Qin, and Tie-Yan Liu. Multilingual neural machine translation with knowledge distillation. In *International Conference on Learning Representations*, 2019.

Samuel Thomas, Sriram Ganapathy, and Hynek Hermansky. Multilingual mlp features for low-resource lvcsr systems. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4269–4272, 2012. doi: 10.1109/ICASSP.2012.6288862.

Samuel Thomas, Kartik Audhkhasi, and Brian Kingsbury. Transliteration Based Data Augmentation for Training Multilingual ASR Acoustic Models in Low Resource Settings. In *Proc. Interspeech 2020*, pages 4736–4740, 2020. doi: 10.21437/Interspeech.2020-2593.

Andros Tjandra, Nayan Singhal, David Zhang, Ozlem Kalinli, Abdelrahman Mohamed, Duc Le, and Michael L. Seltzer. Massively multilingual asr on 70 languages: Tokenization, architecture, and generalization capabilities, 2022.

Sibo Tong. *Multilingual Training and Adaptation in Speech Recognition.* PhD thesis, 2020.

Sibo Tong, Philip N. Garner, and Hervé Bourlard. Cross-lingual adaptation of a ctc-based multilingual acoustic model. *Speech Communication*, 104:39–46, 2018.

Shubham Toshniwal, Tara N. Sainath, Ron J. Weiss, Bo Li, Pedro Moreno, Eugene Weinstein, and Kanishka Rao. Multilingual speech recognition with a single end-to-end model. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4904–4908, 2018. doi: 10.1109/ICASSP.2018.8461972.

Zoltán Tüske, Ralf Schlüter, and Hermann Ney. Multilingual hierarchical MRASTA features for ASR. In *Proc. Interspeech 2013*, pages 2222–2226, 2013. doi: 10.21437/Interspeech.2013-523.

Jörgen Valk and Tanel Alumäe. Voxlingua107: a dataset for spoken language recognition, 2020.

Karel Veselý, Martin Karafiát, František Grézl, Miloš Janda, and Ekaterina Egorova. The language-independent bottleneck features. In *Spoken Language Technology Workshop (SLT)*, pages 336–341, 2012.

Karel Veselý, Arnab Ghoshal, Lukáš Burget, and Daniel Povey. Sequence-discriminative training of deep neural networks. In *Proc. Interspeech 2013*, pages 2345–2349, 2013. doi: 10.21437/Interspeech.2013-548.

Oriol Vinyals, Suman V. Ravuri, and Daniel Povey. Revisiting recurrent neural networks for robust asr. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4085–4088, 2012. doi: 10.1109/ICASSP.2012.6288816.

Andrew J Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967.

Ngoc Thang Vu and Tanja Schultz. Multilingual multilayer perceptron for rapid language adaptation between and across language families. In *Proc. Interspeech*, pages 515–519, 2013.

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. doi: 10.1109/29.21701.

Yufei Wang, Haoliang Li, Lap-pui Chau, and Alex C. Kot. Embracing the dark knowledge: Domain generalization using regularized knowledge distillation. In *Proceedings of the 29th ACM International Conference on Multimedia*, MM '21, page 2595–2604, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450386517.

Shinji Watanabe, Takaaki Hori, Suyoun Kim, John R. Hershey, and Tomoki Hayashi. Hybrid ctc/attention architecture for end-to-end speech recognition. *IEEE Journal of Selected Topics in Signal Processing*, 11(8):1240–1253, 2017. doi: 10.1109/JSTSP.2017.2763455.

Paul J. Werbos. Generalization of backpropagation with application to a recurrent gas market model. *Neural Networks*, 1(4):339–356, 1988. ISSN 0893-6080. doi: https://doi.org/10.1016/0893-6080(88)90007-X.

Ronald J. Williams and David Zipser. *Gradient-Based Learning Algorithms for Recurrent Networks and Their Computational Complexity*, page 433–486. L. Erlbaum Associates Inc., USA, 1995. ISBN 0805812598.

J. HM Wong and M. Gales. Sequence student-teacher training of deep neural networks. 2016.

Jeremy H.M. Wong, Yashesh Gaur, Rui Zhao, Liang Lu, Eric Sun, Jinyu Li, and Yifan Gong. Combination of End-to-End and Hybrid Models for Speech Recognition. In *Proc. Interspeech 2020*, pages 1783–1787, 2020. doi: 10.21437/Interspeech.2020-2141.

Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network, 2015a.

Jingyi Xu, Junfeng Hou, Yan Song, Wu Guo, and Lirong Dai. Knowledge distillation from multilingual and monolingual teachers for end-to-end multilingual speech recognition. In *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 844–849, 2019. doi: 10.1109/APSIPAASC47483.2019.9023203.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In Francis Bach and David Blei, editors, *Proceedings*

of the 32nd International Conference on Machine Learning, volume 37 of *Proceedings of Machine Learning Research*, pages 2048–2057, Lille, France, 07–09 Jul 2015b. PMLR.

Qiantong Xu, Alexei Baevski, and Michael Auli. Simple and Effective Zero-shot Cross-lingual Phoneme Recognition. In *Proc. Interspeech 2022*, pages 2113–2117, 2022. doi: 10.21437/Interspeech.2022-60.

Ji Won Yoon, Hyeonseung Lee, Hyung Yong Kim, Won Ik Cho, and Nam Soo Kim. Tutornet: Towards flexible knowledge distillation for end-to-end speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:1626–1638, 2021. doi: 10.1109/TASLP.2021.3071662.

Dong Yu, Kaisheng Yao, Hang Su, Gang Li, and Frank 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, 2013. doi: 10.1109/ICASSP.2013.6639201.

Xiaohui Zhang, Jan Trmal, Daniel Povey, and Sanjeev Khudanpur. Improving deep neural network acoustic models using generalized maxout networks. In *ICASSP*, pages 215–219, 2014.

Long Zhou, Jinyu Li, Eric Sun, and Shujie Liu. A configurable multilingual model is all you need to recognize all languages. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6422–6426, 2022. doi: 10.1109/ICASSP43922.2022.9747905.

Shiyu Zhou, Shuang Xu, and Bo Xu. Multilingual end-to-end speech recognition with a single transformer on low-resource languages, 2018.

H. Zhu, J. Zhao, Y. Ren, L. Wang, and P. Zhang. Domain adaptation using class similarity for robust speech recognition. *arXiv preprint arXiv:2011.02782*, 2020.

Qifeng Zhu, Barry Chen, Nelson Morgan, and Andreas Stolcke. Tandem connectionist feature extraction for conversational speech recognition. In Samy Bengio and Hervé Bourlard, editors, *Machine Learning for Multimodal Interaction*, pages 223–231, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. ISBN 978-3-540-30568-2.

E. Zwicker. Subdivision of the audible frequency range into critical bands (frequenzgruppen). *The Journal of the Acoustical Society of America*, 33(2):248–248, 07 2005. ISSN 0001-4966. doi: 10.1121/1.1908630.

Piotr Żelasko, Laureano Moro-Velázquez, Mark Hasegawa-Johnson, Odette Scharenborg, and Najim Dehak. That Sounds Familiar: An Analysis of Phonetic Representations Transfer Across Languages. In *Proc. Interspeech*, pages 3705–3709, 2020.