EPFL

Thèse n° 7896

Multilingual Training and Adaptation in Speech Recognition

Présentée le 6 août 2020

à la Faculté des sciences et techniques de l'ingénieur
Laboratoire de l'IDIAP
Programme doctoral en génie électrique

pour l'obtention du grade de Docteur ès Sciences

par

Sibo TONG

Acceptée sur proposition du jury

Prof. P. Frossard, président du jury
Prof. H. Bourlard, Dr Ph. N. Garner, directeurs de thèse
Dr P. Bell, rapporteur
Prof. Y. Estève, rapporteur
Dr M. Salzmann, rapporteur

École polytechnique fédérale de Lausanne

2020

Abstract

State-of-the-art acoustic models for Automatic Speech Recognition (ASR) are based on Hidden Markov Models (HMM) and Deep Neural Networks (DNN) and often require thousands of hours of transcribed speech data during training. Therefore, building multilingual ASR systems or systems on a language with few resources is a challenging task. Multilingual training and cross-lingual adaptation are potential solutions. However, context-dependent states modeling creates difficulties for multilingual and cross-lingual ASR because of the large increase in context dependent labels arising from the phone set mismatch.

The goal of this thesis is to improve current state-of-the-art acoustic modeling techniques in general for ASR, with a particular focus on multilingual ASR and cross-lingual adaptation. We systematically exploited new training frameworks, from Maximum Likelihood Estimation, Connectionist Temporal Classification to Maximum Mutual Information, in the context of phoneme-based multilingual training. In order to minimize the negative effects of data impurity arising from language mismatch, we investigated language adaptive training approaches which help further improve the multilingual ASR performance. Through comprehensive experimental comparison we demonstrated that phoneme-based multilingual models are easily extensible to unseen phonemes of new languages, from which the cross-lingual adaptation yields significant improvement over traditional approaches on limited data. Finally, we proposed a semi-supervised training approach based on dropout to boost the performance in low-resourced languages using untranscribed data.

In the other part of the thesis, we conducted more theoretical analysis of techniques found to be useful in sequential multilingual training. More specifically, we revisited the recurrent architecture based on Bayes's theorem. This leads to a Bayesian recurrent unit dictated by the probabilistic formulation and naturally support a backward recursion. Experiments show that the proposed architecture exceeds the performance of conventional recurrent network.

Together, this thesis constitutes a thorough analysis of the current field. Through theoretical and experimental comparisons, the proposed approaches are shown to yield significant improvement over the conventional hybrid systems on multilingual speech recognition.

Keywords: speech recognition, multilingual training, cross-lingual adaptation, language adaptive training, semi-supervised training, end-to-end, Bayesian inference

Résumé

Les modèles acoustiques pour la reconnaissance automatique de la parole (RAP) les plus avancés à ce jour sont basés sur les modèles de Markov cachés ainsi que les réseaux de neuronaux profonds (DNN, pour deep neural network), et nécessitent souvent des milliers d'heures de discours transcrits pour l'entraînement. C'est pourquoi il est difficile de construire des systèmes multilingues de RAP ou des systèmes basés sur une seule langue avec peu de données. Des solutions potentielles sont l'entraînement multilingue et l'adaptation interlinguistique. Toutefois, la modélisation des états dépendants du contexte entraîne des difficultés pour la reconnaissance automatique de la parole multilingue et interlinguistique en raison d'une forte augmentation des étiquettes dépendantes du contexte, qui découle d'une non-concordance dans les phonèmes.

L'objectif de cette thèse est d'améliorer les techniques de modélisation acoustiques en général pour la RAP, avec un intérêt particulier pour la reconnaissance automatique de la parole multilingue et l'adaptation interlinguistique. Nous avons systématiquement exploité de nouveaux cadres d'entraînement, en passant de l'estimation du maximum de vraisemblance, à la classification temporelle connectionniste et la maximisation de l'information mutuelle, le tout dans le contexte d'un entraînement multilingue basé sur les phonèmes.Afin de minimiser l'impact négatif des données contenant des erreurs qui proviennent de la non-concordance entre les langues, nous avons exploré des approches d'entraînement linguistique adaptatif qui permettent d'améliorer la performance de la reconnaissance automatique de la parole multilingue. A travers une comparaison expérimentale approfondie, nous avons démontré que les modèles multilingues basés sur les phonèmes peuvent aisément s'adapter aux phonèmes inconnus de nouvelles langues, à partir desquels l'adaptation interlinguistique permet d'obtenir de bien meilleurs résultats qu'une approche traditionnelle avec des données limitées. Enfin, nous avons proposé un entraînement semi-supervisé basé sur le dropout afin d'améliorer la performance dans les langues avec peu de données en utilisant des ressources non-transcrites.

Dans l'autre partie de la présente thèse, nous avons procédé à une analyse plus théorique des techniques actuellement jugées utiles dans l'entraînement multilingue séquentiel. Plus précisément, nous avons repensé le réseau de neurones récurrent en nous appuyant sur le théorème de Bayes. Cela nous a mené à une unité bayésienne récurrente dictée par la formulation probabiliste. Cette unité favorise naturellement une récursion à rebours. Des

expérimentations montrent que l'architecture proposée est plus performante que le réseau récurrent classique.

En conclusion, cette thèse constitue une analyse en profondeur du domaine en question. A travers des comparaisons théoriques et expérimentales, les approches avancées dans notre thèse ont conduit à de meilleures performances que les systèmes hybrides traditionnels en matière de reconnaissance automatique de la parole multilingue.

Mots clés: reconnaissance de la parole, entraînement multilingue, adaptation interlinguistique, entraînement linguistique adaptatif, entraînement semi-supervisé, de bout en bout, inférence bayésienne

Acknowledgements

Firstly, I would like to express my gratitude to my enthusiastic advisers, Hervé Bourlard and Phil Garner. They gave me complete freedom of research topics, and were always willing and patient to hear any of my ideas. It was because of this academic freedom and encouragement that I was able work across diverse topics in speech recognition. They taught me how good machine learning research is being done. I could not have asked for a better adviser for my Ph.D research.

I would like to thank my thesis jury members: Pascal Frossard, Mathieu Salzmann, Peter Bell, and Yannick Estève for their insightful comments and encouragement.

I sincerely thank SUMMA project for supporting my research. My gratitude also goes to Amazon and my mentor at Amazon, Simon Wiesler, for giving me an internship opportunity. Internship at Amazon and the discussions with researchers there with diverse background enabled me to widen my research perspective.

Before coming to Idiap, I already had some research experience in speech processing as a masters student at Shanghai Jiao Tong University. I am thankful to Kai Yu for introducing me to the world of speech research.

The colleagues I have had at Idiap have been exceptional. Pranay Dighe, who showed how to really be focused on good research and stay out of distractions. Apoorv Vyas, whose insights and engineering skills resulted in a research collaboration. I would like to thank all the system and the administrative team at Idiap for their support. I thank my roommates and all the wonderful people that I met during my time at Idiap for their friendship and all the fun times. I cherish the daily lunch, dinners and all the board games we have played together.

Finally, I thank my family, especially my parents for their patience, encouragement and freedom to pursue whatever interests me.

Martigny, June 2020

Contents

List of Figures

List of Tables

Part I

In this first part of the thesis, we justify the structure of the thesis, introduce the background of the research, and present state-of-the-art speech recognition techniques and multilingual training approaches.

1 Introduction

1.1 Multilingual Speech Recognition

State-of-the-art Automatic speech recognition (ASR) systems usually consist of two major components: acoustic model and language model. We consider the whole system as a multilingual ASR system if at least one of the two components is multilingual. In this thesis, we mostly focus on improving the acoustic model and assume that the language model is given.

ASR systems have been improved dramatically in recent years. Although it has been shown that recognition accuracy can reach human parity on certain tasks [Xiong et al., 2017], building ASR systems with good performance requires a lot of training data. While sufficient data is available for languages like English, issues with data scarcity arise for under-resourced languages. While text data for training the language model is relatively easier to obtain, collecting transcribed audio data to train the acoustic model is costly.

A common solution is to explore universal phonetic structures among different languages by sharing the parameters of the acoustic model. The state-of-the-art acoustic model is typically built with the hybrid of hidden Markov models (HMMs) and deep neural networks (DNNs). In a DNN, the hidden layers can be considered as a universal feature extractor. Therefore, the hidden layers can be trained jointly using data from multiple languages to benefit each other. The target of the multilingual DNN can be either the universal International Phonetic Alphabet (IPA) based multilingual context-dependent states [e.g., Dupont et al., 2005; Lin et al., 2009; Vu et al., 2014] or a layer consisting of separate activations for each language [e.g., Scanzio et al., 2008; Huang et al., 2013; Ghoshal et al., 2013; Heigold et al., 2013]. The latter architecture has been shown to outperform the monolingual DNN but Lin et al. [2009] reported the performance of IPA-based multilingual DNN sometimes degrades. Although the universal model may share data among various languages, mixture of data creates more variations especially for those identical IPA symbols shared among different languages.

Another common approach for creating models for low-resourced languages is to transfer the knowledge learned from other well-resourced languages to the target language. The bottleneck

approach extracts features from a bottleneck layer of a multilingual model and uses bottleneck features as additional input to train the acoustic model of a target language [e.g., Thomas et al., 2012; Knill et al., 2013; Grézl et al., 2014]. Bottleneck features are believed to contain a minimal multilingual subspace, they generalize well even on new languages. Knowledge can also be transferred by replacing the output layer of a well trained model and re-training the model to predict the targets of a low-resourced language [e.g., Huang et al., 2013; Ghoshal et al., 2013]. The hidden layers are shared and transferred from rich-resourced languages to the target low-resourced language.

All of these models are based on a conventional DNN/HMM hybrid framework [Morgan and Bourlard, 1990; Bourlard and Morgan, 1994; Hinton et al., 2012]. In order to perform well, DNNs model context-dependent states to mitigate the error associated with the Markov assumption. Consequently, training Gaussian Mixture Model (GMM)/HMM hybrid systems and building decision trees to generate the clustered context-dependent states become a prerequisite procedure. However, it creates more challenges for multilingual and cross-lingual ASR because of the large increase in context dependent labels arising from the phone set mismatch. According to Schultz and Waibel [2000], for example, 85% monophones in Portuguese can be covered by German, but the triphones coverage drops to 57%. Although approaches to adapt decision trees have been proposed by Schultz and Waibel [2000], the simple and effective way is to build a language-specific decision tree for the target language and replace the whole output layer of a DNN with the new targets, or to train a completely new network using bottleneck features.

Recently, end-to-end approaches for automatic speech recognition have received a lot of attention. There is not yet a clear definition about the term "end-to-end". Connectionist Temporal Classification (CTC) [Graves et al., 2006] was the first attempt towards end-to-end ASR. The model can learn the pronunciation and acoustics together, but it is incapable of learning the language model well due to the conditional independence assumptions similar to HMMs and it must rely on a separate language model during decoding to obtain good performance. Therefore, CTC models the transformation from the feature end to the phoneme or character end. Alternative approaches, such as RNN-Transducers (RNN-T) [Chorowski et al., 2014] and attention-based methods [Chan et al., 2016] do not have conditional independence assumptions and can simultaneously learn all the components of a ASR system including the pronunciation, acoustics and language model. It models the mapping directly from the feature end to the sentence end. In this thesis, we use the term "end-to-end" to refer to the methods that train a neural network-based model in one stage without relying on prerequisite models, alignments or decision trees. In this context, we consider CTC, RNN-T and attention-based model all as end-to-end methods since no separate prerequisite model training is involved.

It has been shown that end-to-end models are able to achieve equal or better performance than DNN/HMM hybrid systems when large amount of data is available [Sak et al., 2015; Miao et al., 2016]. Multilingual ASR and cross-lingual adaptation can benefit more from the end-to-end training: language-specific prerequisite systems are no longer required; Cross-lingual

adaptation also becomes simpler and more straightforward because end-to-end models get around the problem of context-dependent state mismatch. To this end, the first part of the thesis will focus on building multilingual ASR systems in new frameworks and improving the multilingual ASR performance in general. In this pursuit, we investigated phoneme-based multilingual ASR systems using different training frameworks. Based on the observations from this research, we explored language adaptive training to mitigate the drawbacks arising from mixture of multilingual data. Then, built on the phoneme-based multilingual frameworks, cross-lingual adaptation is investigated to tackle the harder ASR problem where only very limited transcribed data is available. In the rest part of the thesis, we attempt to address the data scarcity problem from a different aspect and focus on more general acoustic modeling in the context of deep learning. We devised a novel semi-supervised training approach to utilize untranscribed data to overcome data scarcity for low-resource scenarios. In addition to practical work, we derived a novel recurrent architecture with probabilistic explanation. It not only results in better performance for ASR tasks but also leads to critical understanding of the recurrent neural network in general.

1.2 Motivation and Objective

This work was funded by the EU H2020 SUMMA project[1]. The goals of this project are to significantly improve media monitoring by creating a platform to automate the analysis of media streams across many languages, to aggregate and distill the content, to automatically create rich knowledge bases, and to provide visualisations to cope with this deluge of data. Robust, multilingual speech recognition across a broad variety of broadcast sources is central to the stream processing that we have undertaken in SUMMA.

One key problem of multilingual acoustic modeling and cross-lingual adaptation is how to utilize data from different languages to learn common properties and transfer them to low-resourced languages. The state-of-the-art addresses this problem by sharing parameters in the acoustic model. However, due to the phone set mismatch among different languages, the prerequisite language-specific GMM/HMM training makes the multilingual and cross-lingual frameworks complicated and constrains the complete utilization of multilingual resources. Moreover, effective cross-lingual adaptation, especially for under-resourced languages, is still an open research problem.

The initial goal of the thesis is to investigate multilingual systems in new frameworks to remove the prerequisite GMM/HMM training and reduce the number of modeling targets without performance loss, so that the multilingual model can be easily extended to new languages by using appropriate cross-lingual adaptation techniques. Later during my PhD, the goal evolved past multilingual ASR. The earlier research implies that the Bayesian approach is beneficial. Thus, in the second part of the thesis, we aim at investigating the underlying

[1] SUMMA stands for Scalable Understanding of Multilingual MediA. See https://www.idiap.ch/en/scientific-research/projects/SUMMA.

techniques for general acoustic modeling in the context of deep learning, which can in turn benefit multilingual ASR and cross-lingual adaptation.

1.3 Main Contributions

More specifically, the main contributions of this thesis can be summarized as follows:

- Investigation of different training frameworks for multilingual ASR. We started from Maximum Likelihood Estimation, comparing it with Connectionist Temporal Classification (CTC) training, and ended up with Maximum Mutual Information (MMI) training. Theoretical comparison is conducted to provide analysis of these training frameworks. Multilingual training performance is also evaluated and compared on commonly used datasets. [Tong et al., 2017a,b, 2019a].

- Exploiting language adaptive training to improve the state-of-the-art multilingual ASR. We explored various approaches for language adaptive training, including concatenating language embedding, Learning Hidden Unit Contribution, and Cluster Adaptive Training. Through theoretical analysis, we concluded that they can be considered as particular cases of Mixture of Experts. It was demonstrated that the multilingual ASR performance can be further improved by applying language adaptive training [Tong et al., 2017a].

- Improving cross-lingual adaptation based on phoneme-based ASR framework. We demonstrated that phoneme-based multilingual model is extensible to new phonemes during cross-lingual adaptation and outperforms conventional cross-lingual adaptation based on hybrid models [Tong et al., 2018a]. In addition, we developed a new approach to initialize the model parameters by incorporating phonological information. It was demonstrated that the proposed approach results in better and faster convergence in cross-lingual adaptation [Tong et al., 2018b].

- Development of a novel semi-supervised training approach. We first showed that using dropout during inference allows us to model acoustic model uncertainty [Vyas et al., 2019]. Based on this observation, we devised a novel framework which uses Dropout at the test time to sample from the posterior predictive distribution of word-sequences to produce unbiased supervision for semi-supervised training. Results on monolingual experiments show that the proposed approach can further improve the performance over the state-of-the-art method [Tong et al., 2019b].

- Theoretical analysis of recurrent neural network. Given a probabilistic interpretation of common feed-forward neural network components, we derived recurrent components in the same spirit. Such components are dictated by the probabilistic formulation and naturally support a backward recursion. Evaluation on state-of-the-art ASR task shows that the resulting architecture can perform as well as a bidirectional recurrent network

with the same number of parameters as a unidirectional one. Further, when configured explicitly bidirectionally, the architecture can exceed the performance of a conventional bidirectional recurrence [Garner and Tong, 2020].

1.4 Thesis Outline

This thesis is divided into three parts. We describe below the main organization of this thesis, briefly describing the main goal of each of its constituting parts and chapters. These chapters are mostly in chronological order.

The first part consists of the current chapter and Chapter 2, Background, where we present the key components of the ASR pipeline, state-of-the-art DNN-based acoustic modeling, multilingual training and cross-lingual adaptation approaches.

The second part is constituted by Chapter 3 and Chapter 4. The focus is investigating promising techniques for multilingual ASR and cross-lingual adaptation and improving the performance practically.

Chapter 3, Multilingual training and language adaptive training, investigates state-of-the-art multilingual training techniques and introduces various approaches to conduct language adaptation during multilingual training. These different approaches are compared both theoretically and practically.

Chapter 4, Multilingual training and cross-lingual adaptation in new frameworks, applies CTC and end-to-end MMI training to multilingual ASR and exploits output layer extension based on monophone acoustic model. We investigate different ways for cross-lingual adaptation and propose a novel parameter initialization approach by incorporating phonological information.

The last part of the thesis consists of Chapter 5 and Chapter 6. We study techniques for more general acoustic modeling in the context of deep learning.

Chapter 5, Semi-supervised training using dropout, presents a novel semi-supervised training approach. Theoretical background is provided and the proposed approach is evaluated on a commonly used ASR dataset.

Chapter 6, Bayesian recurrent unit, derives a novel recurrent architecture with probabilistic explanation. We show that Bayes's theorem leads to a recurrent unit with a prescribed feedback formulation and introduction of a context indicator leads to a variable feedback which is similar to the conventional recurrent units. Such unit naturally supports a backward recursion. Experimental evaluation is also provided on various ASR tasks.

Chapter 7, Conclusions and directions for future work, summarizes the main conclusions of this thesis and provides some possible directions for future work.

2 Background

In this chapter, we provide brief background on hidden Markov models (HMM) and the key components of an HMM-based ASR system in Section 2.1. For a more detailed reading on HMM, conventional HMM-based ASR, and the neural network based hybrid connectionist approach for ASR, we refer the reader to the following resources: [Rabiner, 1989; Jelinek, 1997; Bourlard and Morgan, 2012]. Section 2.2 and 2.3 provide more specific background for multilingual ASR and cross-lingual adaptation. Finally, Section 2.5 gives details of the datasets that were used for evaluating the methods proposed in this thesis.

2.1 Key Components in ASR

A typical ASR system consists of four major components: signal processing and feature extraction, acoustic model (AM), language model (LM), and hypothesis search. As illustrated in Figure 2.1, the signal processing component takes the audio signal as input, converts the input signal from time-domain to frequency-domain, and extracts suitable feature vectors for the following acoustic modeling. The acoustic model generates an acoustic score for the input of feature sequence by integrating knowledge about acoustics and phonetics. The language model then estimates the probability of a hypothesized word sequence by modeling the correlations between words from the training text and generates a language score. The last components, hypothesis search, combines the acoustic score and the language score, and outputs the most likely word sequence as the recognition result. In this thesis, we focus on the acoustic modeling. We did not explore modifications or improvements in the language modeling component in this thesis. In the next subsections, we will introduce the Hidden Markov Model (HMM)-based acoustic model.

2.1.1 Hidden Markov Models

Over the last several decades, hidden Markov models (HMM) have served as the backbone of almost all large-scale ASR systems. As a general framework, HMMs are often considered as

the "wheel" of sequence processing in general, and speech processing in particular. Here, we introduce the basics of HMMs.

A hidden Markov model is a Markov chain where each state generates an observable discrete symbol or a continuous-valued vector as per a state-conditional probability distribution function. While the emitted observations are visible to an observer, the underlying Markov process is hidden. The hidden state sequence is non-deterministic and can only be probabilistically estimated based on the observation sequence and the parameters of the model. Here, we consider only continuous density HMMs which emit real-valued multi-dimensional vectors as observations. The random variable denoting the observed sequence is defined as $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T\}$.

Thus, an HMM can be completely defined by following components:

- Set of states $\mathbb{Q} = \{q_1, q_2, ..., q_K\}$: Random variable s_t, denoting hidden state at time t, takes values from this set
- Set of observations, $\boldsymbol{X}$: Random variable $\boldsymbol{x}_t$, denoting the observation emitted at time t, takes a value $\boldsymbol{x}_t \in \mathbb{R}$
- Initial state distribution $\pi = \{\pi_1, ..., \pi_K\}$ that the Markov chain will start with a particular state.

$$\pi_k = P(s_1 = q_k) \quad s.t. \quad \pi_k \geq 0 \quad \forall k, \quad \sum_{k=1}^{K} \pi_k = 1 \tag{2.1}$$

- Transition probabilities: The probability that the Markov chain will go from one particu-

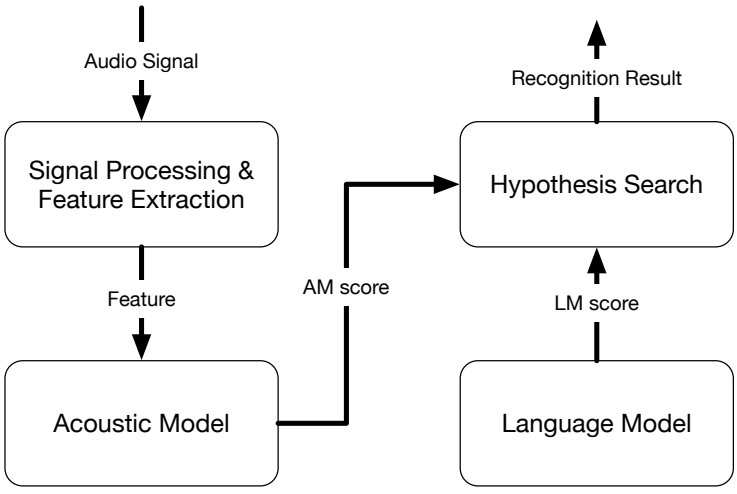


Figure 2.1: Architecture of ASR systems.

lar state to another.

$$a_{i,j} = P(s_t = q_j | s_{t-1} = q_i) \quad s.t. \quad a_{i,j} \geq 0 \quad \forall j, \quad \sum_{j=1}^{K} a_{i,j} = 1 \tag{2.2}$$

- Emission probabilities $b_k(\boldsymbol{x})$: Probability of an observation $\boldsymbol{x} \in \mathbb{R}$ being generated when the underlying hidden state is q_k.

$$b_k(\boldsymbol{x}) = P(\boldsymbol{x}|q_k) \tag{2.3}$$

An HMM based on a first-order Markov chain involves two important assumptions. The first assumption is the first-order Markovian assumption i.e. $P(s_t|\mathbf{s}_1^{t-1}) = P(s_t|s_{t-1})$. The second assumption, famously called HMM conditional-independence assumption, states that the observation emitted at time t is dependent only on the hidden state at time t , and is conditionally independent of the past hidden state as well as observations, i.e. $P(\boldsymbol{x}_t|\boldsymbol{x}_1^{t-1}, \mathbf{s}_1^t) = P(\boldsymbol{x}_t|s_t)$.

One of the most commonly used versions of continuous probability density HMMs is based on multivariate Gaussian Mixture Models (GMM). In a GMM/HMM, each hidden state q_k has a GMM associated with it. Employing HMMs for any task usually results in one or more of the following three standard problems - 1) finding the likelihood of an observation sequence given the HMM parameters, 2) finding the most likely hidden state sequence given an observation sequence and the HMM parameters, and 3) finding the parameters of the HMM given a set of observation sequences. Associated with addressing these three problems are the famous HMM-based algorithms - namely Forward-backward algorithm, Viterbi algorithm, and Baum-Welch algorithm respectively. We refer the reader to the work of Rabiner [1989] for complete details on these algorithms.

2.1.2 Mathematical Formulation of HMM-based ASR

In a typical HMM-based ASR framework, the hypothesized word sequence $\hat{\mathbf{W}}$ is estimated from the sequence of acoustic features $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_t, ..., \boldsymbol{x}_T\}$, where $\boldsymbol{x}_t$ denotes the acoustic feature at time t, as

$$\hat{\mathbf{W}} = \underset{\mathbf{W}}{\operatorname{argmax}} P(\mathbf{W}|\boldsymbol{X}) \tag{2.4}$$

$$= \underset{\mathbf{W}}{\operatorname{argmax}} \frac{p(\boldsymbol{X}|\mathbf{W})P(\mathbf{W})}{p(\boldsymbol{X})} = \underset{\mathbf{W}}{\operatorname{argmax}}\, p(\boldsymbol{X}|\mathbf{W})P(\mathbf{W}) \tag{2.5}$$

where $P(\mathbf{W})$ is the probability of word sequence $\mathbf{W}$ estimated from a language model and $p(\boldsymbol{X}|\mathbf{W})$ is the likelihood of the feature sequence conditioned on the word sequence, estimated from an acoustic model. In the last step, we ignore the denominator probability $p(\boldsymbol{X})$ as

it is independent of the word sequence **W** in the maximization argument. Assuming that the observation sequence $\boldsymbol{X}$ is generated by a hidden Markov model, the task at hand is to compute its probability by marginalizing over all possible hidden state sequences **S** that correspond to the word sequence **W** (i.e. using the Forward-Backward algorithm). Thus, $p(\boldsymbol{X}|\mathbf{W})$ is computed as

$$
\begin{aligned}
p(\boldsymbol{X}|\mathbf{W}) &= \sum_{\mathbf{S}} p(\boldsymbol{X}|\mathbf{S},\mathbf{W})P(\mathbf{S}|\mathbf{W}) && (2.6)\\
&\approx \max_{\mathbf{S}} p(\boldsymbol{X}|\mathbf{S},\mathbf{W})P(\mathbf{S}|\mathbf{W}) && (2.7)\\
&= \pi(s_1)\prod_{t=2}^{T} a_{s_{t-1},s_t}\prod_{t=1}^{T} p(\boldsymbol{x}_t|s_t) && (2.8)
\end{aligned}
$$

where $\hat{\mathbf{S}} = \{s_1, ..., s_T\}$ is the most probable hidden state sequence obtained from the Viterbi algorithm [Rabiner, 1989] for decoding with s_t taking value from the state set $\mathbb{Q}$ and $\pi(s_1)$, a_{s_{t-1},s_t} and $p(\boldsymbol{x}_t|s_t)$ have usual meanings in context of a HMM as described in the previous section. The marginalization over all possible hidden state sequences **S** is typically approximated just by using the most probable hidden states sequence.

2.1.3 DNN/HMM Hybrid Acoustic Models

In a DNN/HMM hybrid ASR system [Bourlard and Morgan, 1994; Hinton et al., 2012], the traditional GMM-based modeling of the state probability distribution functions is replaced by a deep neural network model. The DNN takes as input an acoustic feature vector and predicts the posterior probabilities of all state classes at the output layer. The mapping from the acoustic features to the state posterior probabilities is done through multiple layers of non-linear transformations.

In a Bayesian GMM/HMM system, the frame likelihood $p(\boldsymbol{x}_t|s_t)$ required in (2.8) can be directly computed using the state-specific GMMs. In case of DNN/HMM acoustic models, it has to be indirectly approximated as follows:

$$
p(\boldsymbol{x}|q_k) \sim \frac{p(\boldsymbol{x}|q_k)}{p(\boldsymbol{x})} = \frac{p(q_k|\boldsymbol{x})}{p(q_k)} \tag{2.9}
$$

where the state posterior probability $p(q_k|\boldsymbol{x})$ is obtained at the output of the DNN and $p(q_k)$ is the prior probability of the state q_k obtained from its frequency count in the training data. $p(\boldsymbol{x}|q_k)$ is estimated as the scaled likelihood $\frac{p(\boldsymbol{x}|q_k)}{p(\boldsymbol{x})}$.

The DNN acoustic model can be a simple Multi-Layer Perceptron (MLP) [Rumelhart and McClelland, 1986; Rumelhart et al., 1986] or any other neural network architectures like recurrent neural networks (RNNs) or convolutional neural networks (CNNs)[Schuster and Paliwal, 1997; Sainath et al., 2013]. RNN and its variants, especially Long Short Term Memory

(LSTM) [Hochreiter and Schmidhuber, 1997] and Gated Recurrent Unit (GRU) [Cho et al., 2014], are widely used in ASR because they are capable of capturing sequential dependency. In order to explicitly allow the network to take account of future observations, bidirectional RNNs (Bi-RNNs) [Graves and Schmidhuber, 2005] are commonly used and this type of network remains the state of the art in several fields. We refer the readers to the original papers for more detailed background and Chapter 6 for more discussions about RNNs.

Since the outputs of the neural network acoustic model represent probabilities of the HMM states, thus we use a softmax layer as the last layer of the network. The vector of state posterior probabilities at the output layer of the DNN is also called a DNN posterior.

DNN Training for ASR

Training of a DNN/HMM ASR system usually starts with training a GMM/HMM system first. For a typical ASR task, training the GMM/HMM system involves creating the set of context-dependent states using decision tree based state tying and learning the HMM parameters using the training data. Once the GMM/HMM system is learned, we force-align a sequence of states over the training utterances using their ground-truth text transcript under the Viterbi algorithm. Frame-wise state alignments of the training data provide us with outputs for training the DNN acoustic model.

A DNN acoustic model can be trained either towards the goal of minimizing the framewise state classification error or towards minimizing the sentence level error by using the error backpropagation algorithm [Rumelhart et al., 1986]. Framewise training of the DNN is typically done by minimizing a cross-entropy (CE) loss function. On a training example, if the target posterior vector is **t** and the DNN predicts a posterior vector **y**, then the cross entropy loss is given by:

$$\mathcal{L}_{CE} = -\sum_{k}^{K} t_k \log y_k \tag{2.10}$$

where t_k and y_k are the k^{th} components of DNN target and output vectors, respectively. By minimizing the CE loss over the whole training data, we minimize the Kullback-Liebler distance between the target probability distribution and the DNN output distribution. The network can be also trained using sequence-level criteria such as such as Maximum Mutual Information (MMI) [Bahl et al., 1986] or Minimum Bayes Risk (MBR) [Kingsbury, 2009] criterion. The former one will be discussed in more detail in Section 4.2.

2.2 Multilingual Speech Recognition

DNN/HMM has yielded state-of-the-art ASR performance in language-specific acoustic modelling when large amount of data for the target language is available. However, when multilin-

gual ASR is required, data collection and labelling may become too costly. A common solution is to explore shared phonetic structure among different languages by sharing the parameters in DNNs. By joint training a multilingual DNN using data from different languages, shared phonetic knowledge can be learned.

2.2.1 Multilingual Hybrid System

In hybrid configuration, multilingual DNN is used in the conventional DNN/HMM hybrid way. The target of the multilingual DNN can be either the universal International Phonetic Alphabet (IPA) based multilingual targets [Vu et al., 2014] or a layer consisting of separate activations for each language [Huang et al., 2013].

IPA-based universal DNN To train a multilingual DNN modelling universal multilingual context-dependent states, the monolingual phones are merged if they share the same symbol in the IPA table. The context-dependent states for the training of the multilingual DNN are obtained by training the multilingual GMM/HMM systems and building multilingual decision trees using data from different languages. During decoding, language-specific language models and lexicons are used for each language separately.

Shared-hidden-layer multilingual DNN (SHL-MDNN) The input and hidden layers are shared across all the languages in this architecture. The output layers, however, are not shared. Instead, each language has its own output layer to estimate the posterior probabilities of the context-dependent states specific to that language. Ghoshal et al. [2013] proposed to train DNNs on a sequence of target languages, progressively swapping the output layer with each new language. Huang et al. [2013] presented samples from all languages in an interleaved fashion during training, with the output layer swapped according to the target language being present.

The shared-hidden-layer multilingual DNN has been shown to outperform the monolingual DNN with a $3-5\%$ relative word error rate (WER) reduction but Lin et al. [2009] found the performance of IPA-based universal DNN is worse than the language-specific acoustic models. Although the universal model may share data among various languages, mixture of data creates more variation especially for those identical IPA symbols shared among different languages.

2.2.2 Problems and Motivations

Although the IPA-based multilingual modelling enjoys richer data resources, it has a larger set of units to model as well. Moreover, identical IPA symbols across languages may not correspond to acoustic similarity. Therefore, the IPA-based universal DNN sometimes performs

worse than monolingual acoustic model. Language adaptive training (LAT) is a potential solution, which, however, has not been fully explored in the literature. To address this problem, we may find inspiration in speaker adaptive training. The acoustic characteristic can also vary a lot across different speakers. Intensive approaches have been investigated for speaker adaptive training in the context of DNN-based acoustic model such as exploiting auxiliary features (e.g., i-vector proposed by Saon et al. [2013]) and model-based adaptation techniques (e.g., cluster adaptive training proposed by Tan et al. [2015] and learning hidden unit contribution from Swietojanski et al. [2016]). With the help of effective LAT methods, we hypothesize that, the multilingual DNN can model language specificity while keeping the advantage of data sharing across languages.

2.3 Cross-lingual adaptation

Quick delivery of ASR system for a new language is one of the challenges in the community. In order to perform well, neural networks need to be trained on large amount of data. It raises the question whether knowledge can be transferred from other rich-resourced languages. Some common approaches that utilize information from other languages are discussed in the following.

2.3.1 Tandem System

Tandem systems usually train a multilingual DNN with a bottleneck (BN) layer, to classify monophone states or triphone (context-dependent) states. The output layer can either model multiple sets of language-specific targets or one single universal multilingual target set. The output of the bottleneck layer is used as discriminative features for another GMM-based acoustic model [e.g. Veselỳ et al., 2012; Thomas et al., 2012] or DNN-based acoustic model [Knill et al., 2013]. Veselỳ et al. [2012] shows that multilingual BN features consistently outperform monolingual systems. Because the BN features are language-independent, it generalizes well even on new language. Grézl et al. [2014] also investigate stacked bottleneck architecture, in which the BN feature of the first DNN is used to train a second neural network. The output of the BN layer in the second DNN is used as discriminative feature. There has been extensive work demonstrating the advantage of such multilingual representation. Tandem configuration becomes a standard way of developing ASR system for a new low-resourced language. More recently, advanced architecture has been investigated. Sercu et al. [2017] explored extracting multilingual bottleneck features from CNN and LSTM network.

2.3.2 Phone Mapping

The simplest approach for cross-lingual adaptation is to define a deterministic mapping between source and target phoneme sets. For example, Sim and Li [2009] proposed a data-driven approach to estimate the phone mapping. However, this hard mapping results in

losing information of the target language acoustics that cannot be represented by a single source language phoneme. Imseng et al. [2012] proposed an alternative to learn a probabilistic mapping. The distribution of the target phonemes is expressed over a feature space comprising source language phoneme posterior probabilities, which is formulated as a Kullback-Liebler (KL)-HMM.

2.3.3 Regularisation Approaches

Regularisation approaches aim to improve neural network training for the target language using source language data, for instance, by better initialisation of the DNN or by parameter-sharing. Swietojanski et al. [2012] proposed to use restricted Boltzmann machine (RBM) pre-training on source languages to improve the initialization of the DNN parameters for the target language. Ghoshal et al. [2013] and Huang et al. [2013] utilized multilingual data to train hybrid DNNs, where the hidden layers are shared across language. During cross-lingual adaptation, the hidden layers are fixed and only the output layer is re-estimated for target language. The effect is to regularize the networks by sharing lower hidden layers. Note that, KL-HMM mentioned above can be considered as a special DNN/HMM in which an additional layer that evaluates KL-divergence is put on top of original DNN and the softmax layer serves as a bottleneck layer. Thus, it has a similar idea to regularisation approaches.

2.3.4 Problems and Motivations

There are several ways to address the data scarcity problem. On the one hand, cross-lingual adaptation can exploit knowledge learned from well resourced languages. However, one of the problems of the current cross-lingual adaptation technique is that it is hard to extend the multilingual DNN to new language. A completely different GMM/HMM system is required for the new language and the multilingual DNN has to be retrained. Alternative frameworks that are independent of HMMs (e.g. attention mechanism from Chan et al. [2016] and Connectionist Temporal Classification proposed by Graves et al. [2006]) can be considered. On the other hand, semi-supervised training is also beneficial for low-resourced languages since unlabeled data is less costly to obtain. Exploiting unlabeled data can mitigate data scarcity from a different aspect.

2.4 Evaluation Metric

It is common to measure the performance of different acoustic models with Phoneme Error Rate (PER) so that lexical and linguistic effect can be minimized. To evaluate the performance of different ASR systems as a whole, the word error rate (WER) is normally used. Therefore, Section 2.4.1 shows how we calculate PER and WER for an ASR system. To determine if there is a significant difference between the error rates measured for two different systems, we then use the significance test described in Section 2.4.2

2.4.1 Phoneme Error Rate and Word Error Rate

To calculate the error rate, the output of the system need to be aligned and compared with the original ground truth. Dynamic programming will then be performed to match the recognized and reference label sequences. After this matching procedure, the number of insertion errors (E_I), substitution errors (E_S), and deletion errors (E_D) can be calculated. The Error Rate is then defined as:

$$\text{Error Rate} = \frac{E_I + E_S + E_D}{N} \tag{2.11}$$

where N is the total number of labels in the reference transcription. Hence, word error rate is measured if the labels are words; phoneme error rate is calculated if the labels are phonemes.

2.4.2 Significance Test

We calculate the equal-tailed 95% credible interval for a beta assumption for the error rate to measure the statistical significance of any improvements. In addition, we also perform matched-pair t-test between systems, the test statistic being the utterance-wise difference in word (phoneme)-level errors normalized by the reference length.

2.5 Datasets

2.5.1 Globalphone Corpus - French, German, Portuguese, Spanish, Russian

GlobalPhone [Schultz et al., 2013] is a multilingual database of high-quality read speech with corresponding transcriptions and pronunciation dictionaries in 20 languages. GlobalPhone was designed to be uniform across languages with respect to the amount of data, speech quality, the collection scenario, the transcription and phone set conventions. With more than 400 hours of transcribed audio data from more than 2000 native speakers, the complete data corpus comprises (1) audio/speech data, i.e. high-quality recordings of spoken utterances read by native speakers, (2) corresponding transcriptions, (3) pronunciation dictionaries covering the vocabulary of the transcripts, and (4) baseline n-gram language models.

In this thesis, we used the French (FR), German (GE), Portuguese (PO), Russian (RU) and Spanish (SP) datasets from the GlobalPhone corpus. Each language has roughly 20 hours of speech for training and two hours for development and evaluation sets, from a total of about 100 speakers. Results on evaluation sets are reported. Development sets are used for tuning the hyper-parameters. The trigram language models that we used are publicly available[1]. The detailed statistics for each of the languages is shown in Table 6.1.

[1] http://www.csl.uni-bremen.de/GlobalPhone/

Table 2.1: Statistics of the subset of GlobalPhone languages used in this work: the amounts of speech data for training and evaluation sets are in hours.

Language	Vocab	PPL	#Phones	Train	Dev	Eval
FR	65k	324	38	22.7	2.1	2.0
GE	38k	672	41	14.9	2.0	1.5
PO	62k	58	45	22.7	1.6	1.8
RU	293k	1310	48	21.1	2.7	2.4
SP	19k	154	40	17.6	2.0	1.7

Table 2.2: Statistics of the BCN dataset: the amounts of speech data for training and evaluation sets are in hours.

	Type	Length	#Words	OOV(%)
Train	Radio	146.28	1209k	0.55
Dev	TV	8.93	84k	1.11
Eval	Radio	6.33	50k	0.36

2.5.2 Broadcast News - German

The Broadcast News (BCN) Corpus [Weninger et al., 2014] is a German dataset recorded at Duisburg University and consists of over 160 hours of German speech, including mainly radio broadcasts, but also news on television. The utterances from the clean speech parts are divided into a training, development, and test set, ignoring the low-quality segments (approx. 1.4% of the total utterance length). A trigram LM was trained based on the archive of the German newspaper *taz* (*die tageszeitung*), consisting of 633 611 articles from the years 1986-2000 with 185.9 million words in total. For the LM vocabulary, pronunciations were taken from a semi-automatically generated dictionary. The recording length, the number of utterances, and the number of out-of-vocabulary (OOV) entries of the language model for the three sets is shown in Table 2.2.

2.5.3 BREF - French

BREF [Lamel et al., 1991] is a French dataset designed to provide continuous speech data for the development of dictation machines, for the evaluation of continuous speech recognition systems (both speaker-dependent and speaker-independent), and for the study of phonological variations. The text to be read was selected from 5 million words of French newspaper, *Le Monde*. In total, 11000 texts were selected, with the selection criteria that emphasized maximizing the number of distinct triphones. Separate text materials were selected for training and testing corpora. The speech was recorded by 120 speakers, each providing between 5000 and 10000 words (approximately 40-70 minutes) of speech.

The speech data has been recorded at LIMSI. The talker, located in an acoustically isolated room, reads the text. The texts are presented in paragraph context when appropriate. Record-

Table 2.3: Statistics of the WSJ dataset: the amounts of speech data for training and evaluation sets are in hours.

	#Speakers	#Utterance	Length
Train	284	37416	81
Dev	10	503	1.1
Eval	8	333	1.1

ings are made in stereo using a close-talking, noise canceling microphone. The recorded sentences have an average duration of 15 seconds and a signal-to-noise ratio of about 60 dB. Each sentence was manually aligned with the transcription. In total, there is around 104 hours data for training, 10 hours data for development and 8.8 hours data for testing.

2.5.4 Wall Street Journal - English

The Wall Street Journal (WSJ) corpus [Paul and Baker, 1992] contains large amounts of read English speech material from a large number of speakers and has associated text material which can be used as a source for statistical language modeling. The training set consists of roughly 37 000 sentences from 284 speakers which is normally referred to as *SI-284*. The data labeled as *dev92* consists of 503 sentences of development data from ten different speakers. The data labeled as *eval93* consists of 333 sentences of test data from another eight different speakers. Detailed statistics is shown in Table 2.4.

2.5.5 Fisher - English

Fisher English Training Speech [Cieri et al., 2004] consists of two parts. Part 1 Transcripts represents the first half of a collection of conversational telephone speech (CTS) that was created at LDC in 2003. It contains time-aligned transcript data for 5 850 complete conversations, each lasting up to 10 minutes. In addition to the transcriptions, which are found under the trans directory, there is a complete set of tables describing the speakers, the properties of the telephone calls, and the set of topics that were used to initiate the conversations.

Fisher English Training Part 2 Speech represents the second half of a collection of conversational telephone speech (CTS) that was created at the LDC during 2003. It contains 5 849 audio files, each one containing a full conversation of up to ten minutes. The two parts together contains roughly 1500 hours transcribed speech for training, 3.3 hours data for development and 3.2 hours data for testing.

2.5.6 AMI - English

The AMI corpus [Carletta et al., 2005] contains recordings of spontaneous conversations between a group of participants in meeting scenarios. The meeting scenarios was designed

Table 2.4: Details of AMI database: the amounts of speech data for training and evaluation sets are in hours.

	#Words	#Utterance	Length
Train	802,604	108,221	81
Dev	94,914	13,059	9
Eval	89,635	12,612	9

such that the participants freely discuss and debate over some topics. The meetings were recorded in English, although the speakers were mostly non-native. The recordings were done in three different rooms with different acoustic environments across three geographical locations in the UK, the Netherlands, and Switzerland. AMI corpus is multi-modal and provides audio recordings from close-talk as well as far-field microphones. In this thesis, we only used the speech data recorded using close-talk microphones. Due to the conversational style and the fact that speakers frequently overlap and interrupt other speakers' speech, the AMI corpus has proved to be a challenging ASR task.

The close-talk microphone speech is termed as individual headset microphone (IHM) condition in AMI. The dataset is available at 16kHz sampling rate with nearly 100 hours of meeting recordings divided approximately as 81 hours train set, 9 hours development and 9 hours eval set. We used 10% of the training data for cross-validation during DNN training, whereas the development set was used for tuning the hyper-parameters of our proposed approaches. We used a pronunciation dictionary of 47k words and a trigram language model for decoding in our ASR experiments.

Part II

In the second part of the thesis, we investigate and apply promising techniques to multilingual ASR problems. More specifically, Connectionist Temporal Classification and Maximum Mutual Information are exploited in the context of multilingual training to remove the need of building GMM/HMM in conventional training pipeline and reduce the modeling targets without performance loss. Language adaptive training is explored to further improve the multilingual ASR performance. Moreover, cross-lingual adaptation based-on CTC training is systematically investigated. This part of the thesis provides thorough experimental investigations of promising techniques for multilingual ASR and cross-lingual adaptation.

3 Multilingual Training and Language Adaptive Training

This chapter describes the investigation of multilingual training using the state-of-the-art DNN/HMM hybrid framework. In multilingual DNN training, the hidden layers (possibly extracting bottleneck features) are usually shared across languages, and the output layer can either model multiple sets of language-specific context-dependent states or one single universal multilingual targets. Both architectures are investigated, exploiting and comparing different language adaptive training (LAT) techniques originating from successful DNN-based speaker adaptation. More specifically, speaker adaptive training methods such as Cluster Adaptive Training (CAT) and Learning Hidden Unit Contribution (LHUC) are considered. In addition, we show both CAT and LHUC can be considered as particular cases of Mixture of Experts (MoE). Experimental evaluation confirms that language adaptive training can further improve the performance of multilingual training. The work in this chapter was published as Tong et al. [2017a].

3.1 Maximum Likelihood Estimation

The conventional training of the hybrid acoustic model is based on Maximum likelihood estimation (MLE). As discussed in Section 2.1, given the acoustic feature vector $\boldsymbol{X}$, the ASR problem can be formulated as follows:

$$\hat{\mathbf{W}} = \underset{\mathbf{W}}{\operatorname{argmax}}\, p(\boldsymbol{X}|\mathbf{W})\, p(\mathbf{W}) \tag{3.1}$$

A dictionary is normally used to convert the words to smaller pronunciation units which are modeled by the acoustic model. The acoustic model estimates $p(\boldsymbol{X}|\mathbf{W})$, the probability of a sequence of acoustic features $\boldsymbol{X}$ conditioned on word sequence $\mathbf{W}$. The language model estimates the probability of the hypothesized word sequence $p(\mathbf{W})$. The decoder finally outputs the most likely word sequence by searching and comparing the hypothesized word sequences.

HMM-based hybrid model is broadly used as the acoustic model to estimate $p(\mathbf{X}|\mathbf{W})$. Let the input sequence $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_T\}$. The likelihood of observing an acoustic feature sequence $\boldsymbol{X}$ given the word sequence $\mathbf{W}$ with state sequences $\mathbf{S} = \{s_1, ..., s_T\}$ as hidden variable can be written as:

$$p(\boldsymbol{X}|\mathbf{W}) = \sum_{s_1^T:\mathbf{W}} p(\boldsymbol{X}|s_1^T, \mathbf{W}) P(s_1^T|\mathbf{W}) \tag{3.2}$$

where $s_1^T : \mathbf{W}$ denotes all the possible state sequences allowed by the word sequence $\mathbf{W}$. Assuming a first order Markov model, i.e. $p(s_t|s_{t-1}, ..., s_1) = p(s_t|s_{t-1})$ and the independence of acoustic observations given the state, (3.2) can be rewritten:

$$\begin{aligned}
\mathcal{L}(\theta) &= p(\boldsymbol{X}|\mathbf{W}, \theta) && (3.3)\\
&= \sum_{s_1^T:\mathbf{W}} \prod_{t=1}^{T} p(\boldsymbol{x}_t|s_t, \theta) p(s_t|s_{t-1}) && (3.4)
\end{aligned}$$

where θ stands for the parameters of the probability density function $p(\boldsymbol{x}_t|s_t, \theta)$ and $p(s_t|s_{t-1})$ are the transition probabilities defined in an HMM.

Using the maximum likelihood estimation (MLE) criterion, we obtain the following derivative:

$$\begin{aligned}
\frac{\partial}{\partial\theta}\log\mathcal{L}(\theta) &= \frac{1}{\mathcal{L}(\theta)}\frac{\partial}{\partial\theta}\mathcal{L}(\theta) && (3.5)\\
&= \frac{1}{\mathcal{L}(\theta)}\sum_{t,s}\frac{\partial\mathcal{L}(\theta)}{\partial p(\boldsymbol{x}_t|s_t,\theta)}\cdot\frac{\partial p(\boldsymbol{x}_t|s_t,\theta)}{\partial\theta} && (3.6)\\
&= \frac{1}{\mathcal{L}(\theta)}\sum_{t,s}\frac{\partial\mathcal{L}(\theta)}{\partial p(\boldsymbol{x}_t|s_t,\theta)}\cdot p(\boldsymbol{x}_t|s_t,\theta)\cdot\frac{\partial}{\partial\theta}\log p(\boldsymbol{x}_t|s_t,\theta) && (3.7)\\
&\overset{\text{model}}{=} \frac{1}{\mathcal{L}(\theta)}\sum_{t,s}\left[\sum_{s_1^T:\mathbf{W},s_t=s} p(s_1^T|\mathbf{W})\cdot\frac{\prod_{t'=1}^{T} p(\boldsymbol{x}_{t'}|s_{t'},\theta)}{p(\boldsymbol{x}_t|s_t,\theta)}\right]\cdot p(\boldsymbol{x}_t|s_t,\theta) && (3.8)\\
&\quad\cdot\frac{\partial}{\partial\theta}\log p(\boldsymbol{x}_t|s_t,\theta) && (3.9)\\
&= \sum_{t,s} q_t(s|\boldsymbol{x}_1^T, \mathbf{W}, \theta)\cdot\frac{\partial}{\partial\theta}\log p(\boldsymbol{x}_t|s_t,\theta) && (3.10)
\end{aligned}$$

with

$$q_t(s|\boldsymbol{x}_1^T, \mathbf{W}, \theta) = \frac{\sum_{s_1^T:\mathbf{W},s_t=s} p(\boldsymbol{x}_1^T, s_1^T|\mathbf{W}, \theta)}{\sum_{s_1^T:\mathbf{W}} p(\boldsymbol{x}_1^T, s_1^T|\mathbf{W}, \theta)}. \tag{3.11}$$

The quantity $q_t(s|\boldsymbol{x}_1^T, \mathbf{W}, \theta)$ can be efficiently computed using the Baum-Welch algorithm and is also known as the soft alignment. Alternatively, one can also do the maximum approximation and calculate a Viterbi alignment and encode $q_t(s|\boldsymbol{x}_1^T, \mathbf{W}, \theta)$ as a one-hot encoding of the Viterbi alignment.

For neural-network based models, the probability $p(\boldsymbol{x}_t|s_t, \theta)$ in the conventional hybrid modeling is modeled as

$$p(\boldsymbol{x}_t|s_t, \theta) \sim \frac{p(s_t|\boldsymbol{x}_t, \theta)}{p(s_t)} \tag{3.12}$$

For the gradient, $p(s_t|\boldsymbol{x}_t, \theta)$ is estimated as the neural network output. $p(\boldsymbol{x}_t|\theta)$ and $p(s_t)$ are constant w.r.t. θ. Thus, (3.10) becomes:

$$\frac{\partial}{\partial \theta} \log \mathcal{L}(\theta) = \sum_{s,t} q_t(s|\boldsymbol{x}_1^T, \mathbf{W}, \theta) \cdot \frac{\partial}{\partial \theta} \log p(s_t|\boldsymbol{x}_t, \theta) \tag{3.13}$$

In DNN/HMM hybrid systems, $q_t(s|\boldsymbol{x}_1^T, \mathbf{W}, \theta)$ can be also computed using the Baum-Welch algorithm. However, in practice, it is normally fixed and pre-calculated by a GMM/HMM model, known as the forced alignment. Then, the derivatives correlate to cross entropy (CE) criterion as in (2.10).

The forced alignment $q_t(s|\boldsymbol{x}_1^T, \mathbf{W}, \theta)$ for DNN training is calculated in an iterative way to steadily refine the alignment. It starts with a linear alignment and then trains increasingly powerful models (monophone GMM→ tied context-dependent (triphone) GMM) which are then used to gradually obtain better alignments. A neural network based acoustic model is then trained using the alignment to optimize the frame-wise cross entropy criterion, possibly followed by sequence discriminative training. Therefore, this DNN/HMM hybrid training is a particular approximation of maximum likelihood estimation. Although it is also possible to directly apply Baum-Welch training on a DNN/HMM hybrid framework, we consider this separate DNN/HMM hybrid training procedure as our baseline because it is the most widely used framework in practice and yields the state-of-the-art performance. The introduction of context-dependent states improves the recognition performance but it also results in more work and challenges for multilingual ASR. Prerequisite GMM/HMM training has to be done for each of the language and initialization of multilingual context-dependent models leads to an explosion of context-dependent states.

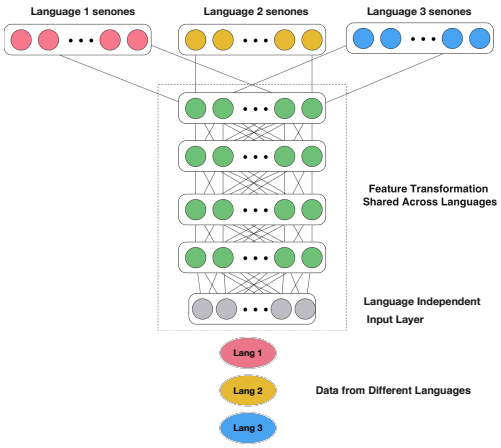


Figure 3.1: Architecture of the SHL-MDNN.

3.1.1 Multilingual Training

Universal Phone Set Multilingual DNN

The main goal of multilingual acoustic modeling is to share the acoustic data across multiple languages to cover as much as possible the contextual variation in all languages being considered. One way to achieve such data sharing is to define a common phonetic alphabet across all languages. This common phone set can be either derived in a data-driven way, or obtained from the International Phonetic Alphabet (IPA). In this thesis, we used the IPA and merge the monolingual phonemes if they share the same symbol in the IPA table. The context-dependent targets for training the multilingual DNN are obtained by training the multilingual GMM/HMM systems and building multilingual decision trees to generate tied-state alignments. During decoding, language-specific language models and lexicons are used for each language separately. This architecture is subsequently denoted as MUL-IPA.

Shared-Hidden-Layer Multilingual DNN

In addition to modelling one single universal multilingual senone set, the output layer can also model multiple sets of language-specific targets and hidden layers are shared across languages. Ghoshal et al. [2013] proposed to train DNNs on a sequence of target languages, progressively swapping the output layer with each new language. Whilst in the work of Huang et al. [2013] and Heigold et al. [2013], data from all languages is presented in an interleaved fashion during training, with the output layer swapped according to the target language being present. Here, only the architecture in the work of Huang et al. [2013] is discussed and we denote this shared-hidden-layer multilingual DNN as SHL-MDNN following Huang et al. [2013].

Figure 3.1 depicts the architecture used for multilingual ASR. The input and hidden layers

Table 3.1: Comparison between monolingual baseline systems and multilingual training in WER(%).

system	FR	GE	PO	RU	SP
monolingual baseline	23.2	16.6	19.9	28.8	9.0
MUL-IPA	23.3	18.5	19.3	30.4	9.8
SHL-MDNN	**23.0**	**15.6**	**18.9**	**28.3**	**8.6**

are shared across all the languages. The output layers, however, are not shared. Instead, each language has its own output layer to estimate the posterior probabilities of the context-dependent states specific to that language. Therefore, multiple GMM/HMM systems have to be trained for each language in order to generate the forced alignment for training the multilingual DNN. During recognition, language-specific prior and posterior probabilities are used for decoding.

3.1.2 Experimental Evaluation

In this section, we report state-of-the-art multilingual DNN/HMM hybrid training on GlobalPhone as described in Section 2.5.1. In this study, we used the French (FR), German (GE), Portuguese (PO), Russian (RU) and Spanish (SP) datasets from the GlobalPhone corpus. Each language has roughly 20 hours of speech for training and two hours for development and evaluation sets. Results on evaluation sets are reported as the development sets are used for hyper-parameter tuning. The trigram language models that we used are publicly available[1].

We conducted two different sets of experiments by varying the output layer. In the first set of experiments, the SHL-MDNN architecture was used where each language has its corresponding softmax output. Monolingual GMM/HMM systems were trained to obtain the language specific tied-state alignments. The second set of experiments was conducted using the IPA-based universal triphone output. To create the universal phone set, we merged all the monolingual phones which share the same symbol in the IPA table. Multilingual GMM/HMM system was trained and multilingual decision tree was built to generate tied-state alignments.

The Kaldi speech recognition toolkit [Povey et al., 2011] was used to build all the systems. For each language, we built GMM/HMM systems, using 39-dimensional MFCC features (C0-C12, with delta and acceleration coefficients). The number of context-dependent triphone states for each language is 3100 with a total of 50K Gaussians (an average of roughly 16 Gaussians per state). The number of the IPA-based multilingual context-dependent triphone states is 8000 with a total of 150K Gaussians. All the DNNs used in the experiments had 6 hidden layers, each consisting of 2,000 sigmoidal units and were trained from 11 consecutive frames after restricted Bolzmann machine (RBM) pretraining.

Here we present the comparison between different multilingual architectures and baseline

[1] http://www.csl.uni-bremen.de/GlobalPhone/

monolingual systems, which is listed in Table 4.5. It shows that SHL-MDNN achieves improvement over monolingual DNN baseline systems in all languages. However, the multilingual training using universal phone set does not show much improvement and in most cases it is even worse. The result is consistent with previous work [Huang et al., 2013; Lin et al., 2009; Chen and Mak, 2015]. Although the IPA-based multilingual modelling enjoys richer data resources, it has a larger set of units to model as well. Moreover, identical IPA symbols across languages may not correspond to acoustic similarity, which will also results in problems for clustering the context-dependent states. Applying language adaptive training is a potential solution as it encourages the multilingual network to capture language specificity with language-specific parameters while modeling language-independent characteristics using shared parameters. Language adaptive training is discussed in the following sections.

3.2 Language adaptive training

We have investigated the state-of-the-art multilingual training framework in the last section. Multilingual training allows a shared, language-independent speech representation to be more robustly learned in the shared layers of a neural network due to the increased training data presented. However, the performance of multilingual training is sometimes worse than the language-specific acoustic models especially when a universal multilingual output is used [Lin et al., 2009]. This has been also observed from our experiments in Section 3.1.2. Mixture of multilingual data leads to more variations especially for the shared phonemes or graphemes. Thus, the universal network may fail to model the language specificity.

Therefore, we present language adaptive training approaches in this section to address this problem by providing additional language information. These approaches are systematically compared from both theoretical and practical aspects.

3.2.1 Related Work

One of the most widely used approaches for language adaptation in multilingual training is to use an additional feature vector that represents the language identity. This vector can be a one-hot vector [e.g. Müller and Waibel, 2015] or a language embedding learned from another model [e.g. Miiller et al., 2018]. Such language vectors are normally concatenated to the input acoustic feature to provide additional language information, similar to the use of i-vector for speaker adaptive training.

Recently, various speaker adaptive training (SAT) approaches based on DNN have been proposed. Cluster adaptive training (CAT) was extended from GMM to DNN [Tan et al., 2015]. It factorizes the hidden layers in DNN into a set of canonical weight matrices and speaker-dependent interpolation parameters. Similar aproaches have been proposed independently by Delcroix et al. [2015] and Wu and Gales [2015]. Swietojanski and Renals [2014] have also introduced learning hidden unit contribution (LHUC). LHUC learns speaker-specific

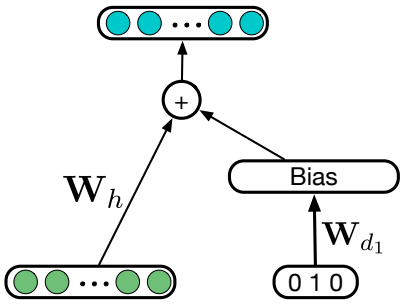


Figure 3.2: Concatenating a one-hot language vector to one layer. The concatenation multiplied by the following weight matrix is equivalent to learning a language-specific bias and adding it to the original result.

parameters to re-weight hidden units in a speaker-dependent manner. It is demonstrated that LHUC results in consistent WER reductions for speaker and environment adaptation [Swietojanski et al., 2016]. Inspired by the successful work in speaker adaptation, similar approaches could be applied for language adaptive training.

3.2.2 Language Embedding

The intuition behind language adaptive training is that a multilingual model can "specialize" on each individual language with the help of additional language information instead of being a model biased to languages with more data. Therefore, we normally assume the language information is also known during inference. The language information can be represented in several different ways (as a one-hot vector, as an embedding vector). Using a one-hot language vector and concatenating it to the input feature (or to the hidden layers) is most intuitive and effective approach to enable this "language awareness".

Essentially, concatenating the language vector to the hidden feature is equivalent to learning a language-specific bias, as shown in Figure 3.2. The matrix $\boldsymbol{W}_{d_1}$ captures the language specific characteristics and applies a language-specific shift in the network to make it specialized on each languages.

3.2.3 Learning Hidden Unit Contribution (LHUC)

Using only a language-specific bias to differentiate languages might be too simple to capture the difference between languages. In stead, we could learn a scaling factor from the language

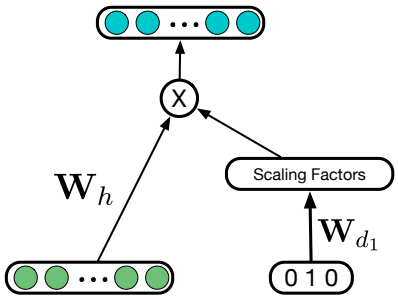


Figure 3.3: Applying Learning Hidden Unit Contribution (LHUC) in one layer. It learns language-specific scaling factors to re-weight the activation of the hidden units.

vector, which is used to element-wisely scale each activation from the previous layer, as shown in Figure 3.3.

This is conceptually the same as LHUC. LHUC is a method that linearly re-combines hidden units in a speaker- or environment-dependent manner [Swietojanski and Renals, 2014, 2016]. Given adaptation data, LHUC re-scales the contributions (amplitudes) of the hidden units in the model without actually modifying their feature receptors. A speaker-dependent amplitude function is introduced to modify $\boldsymbol{h}^{sl}$, the hidden unit outputs in layer l for speaker s:

$$\boldsymbol{h}^{sl} = \xi(\mathbf{r}^{sl}) \odot \psi(\boldsymbol{W}^l \boldsymbol{h}^{l-1} + \mathbf{b}^l) \tag{3.14}$$

$\mathbf{r}^{sl} \in \mathbb{R}$ is an adaptable speaker-dependent vector, re-parametrised by a function $\xi : \mathbb{R} \rightarrow \mathbb{R}^+$. A sigmoid function with range $(0,2)$ is usually used. $\boldsymbol{W}^l \in \mathbb{R}^{d_{\boldsymbol{h}^l} \times d_{\boldsymbol{h}^{l-1}}}$ is the weight matrix where $d_{\boldsymbol{h}^l}$ is the dimension of vector $\boldsymbol{h}^l$. $\mathbf{b}^l$ denotes the bias. ψ is the hidden unit activation function, and $\odot$ denotes a Hadamard product. The number of adaptable parameters for each speaker or language is the same as the number of hidden units in each layer.

3.2.4 Cluster Adaptive Training (CAT)

Cluster Adaptive Training (CAT) is another broadly used technique for speaker adaptive training. We could apply the same idea for language adaptive training. CAT was initially proposed for GMM/HMM acoustic models [Gales, 2000]. It was then extended to DNN by introducing multiple canonical weight matrices for a DNN layer as depicted in Figure 3.4. In CAT, multiple weight matrices or sub-networks are constructed to form the bases of a canonical parametric space. During adaptation, an interpolation vector, specific to a particular acoustic condition, is used to combine the multiple sub-networks into a single adapted DNN [Tan et al.,

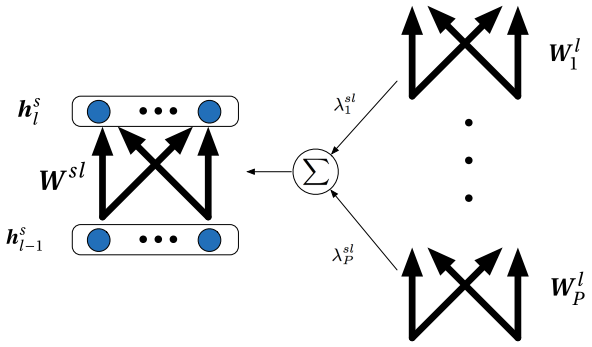


Figure 3.4: Architecture of CAT-DNN for one layer.

2015; Delcroix et al., 2015; Wu and Gales, 2015]. More formally, for a specific speaker s, the adapted weight matrix between layer $l-1$ and layer l, $\boldsymbol{W}^{sl}$, is represented as an interpolation of the canonical DNN matrices:

$$\boldsymbol{W}^{sl} = \sum_{c=1}^{P} \lambda_c^{sl} \boldsymbol{W}_c^l \tag{3.15}$$

where $[\boldsymbol{W}_1^l, ..., \boldsymbol{W}_P^l]$ is the set of weight matrix bases between layer $l-1$ and layer l, P is the number of bases, $\boldsymbol{\lambda}^{sl}$ denotes the speaker dependent interpolation vector for layer l and speaker s. Therefore a general form of CAT-layer output can be obtained as following:

$$\boldsymbol{h}_l^s = \psi(\boldsymbol{x}_l^s), \tag{3.16}$$

$$\boldsymbol{x}_l^s = \sum_{c=1}^{P} \lambda_c^{sl} \boldsymbol{W}_c^l \boldsymbol{h}_{l-1}^s + \mathbf{B}^l \boldsymbol{\alpha}^{sl} \tag{3.17}$$

where $\boldsymbol{\alpha}^{sl}$ is the interpolation vector of bias for speaker s in layer l, $\mathbf{B}^l = [\mathbf{b}_1^l, ..., \mathbf{b}_P^l]$ is the concatenated bias bases and ψ is the hidden unit activation function. During training, all the parameters in a CAT-DNN, including the interpolation vector and the canonical bases, are trained simultaneously using gradient descent algorithm. Since the canonical bases are shared across speakers, only the interpolation parameters are speaker specific. Thus, the number of adaptable parameters is equal to the number of used clusters.

3.2.5 Mixture of Expert (MoE)

The concept of a mixture of experts (MoEs) was initially introduced by Jacobs et al. [1991]. A MoE component consists of a set of n sub-networks ("experts") $E_1, ..., E_n$, and a "gating

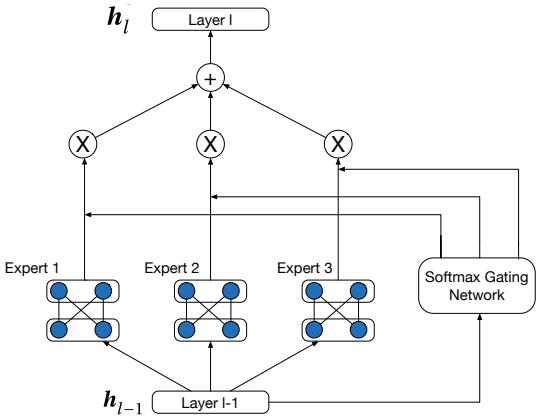


Figure 3.5: Architecture of MoE for one layer.

network" G whose output is a n-dimensional weight vector. Inference is performed based on a weighted combination of outputs from multiple experts. For layer l and a given input $\boldsymbol{h}_{l-1}$, we denote the output of the gating network and the output of the i-th expert network in this layer as $G^l(\boldsymbol{h}_{l-1})$ and $E_i^l(\boldsymbol{h}_{l-1})$. The output $\boldsymbol{h}_l$ of the MoE component can be written as follows:

$$\boldsymbol{h}_l = \sum_{i=1}^{n} G^l(\boldsymbol{h}_{l-1})_i E_i^l(\boldsymbol{h}_{l-1}) \tag{3.18}$$

Each expert E_i accepts inputs of the same size and produces the same-sized outputs. A *softmax* function is usually applied to the output of the gating network to generate soft probabilities of picking the corresponding experts. Figure 3.5 shows the overview of a MoE component. The number of adaptable parameters is subject to the complexity of the experts and the gating network and can be adaptively tuned according to the amount of available adaptation data.

MoE allows the individual experts to specialize on smaller parts of a larger problem and uses soft partitions of the data. Thus MoE is competitive for nonlinear classification problems with data that naturally contains distinctive subsets of patterns.

3.2.6 A Unified Framework

In CAT, if the interpolation vector of weight $\boldsymbol{\lambda}^{sl}$ and the interpolation vector of bias $\boldsymbol{\alpha}^{sl}$ are tied, (3.17) can be written as

$$\boldsymbol{x}_l^s = \sum_{c=1}^{P} \lambda_c^{sl}(\boldsymbol{W}_c^l \boldsymbol{h}_{l-1}^s + \mathbf{b}_c^l) \tag{3.19}$$

This interpolation can be viewed more intuitively as depicted in the left part of Figure 3.6. The graph describes a CAT layer which contains two canonical bases. If we concatenate the weight bases into a single matrix $\boldsymbol{W}_B^l$ and concatenate the bias bases into a single vector

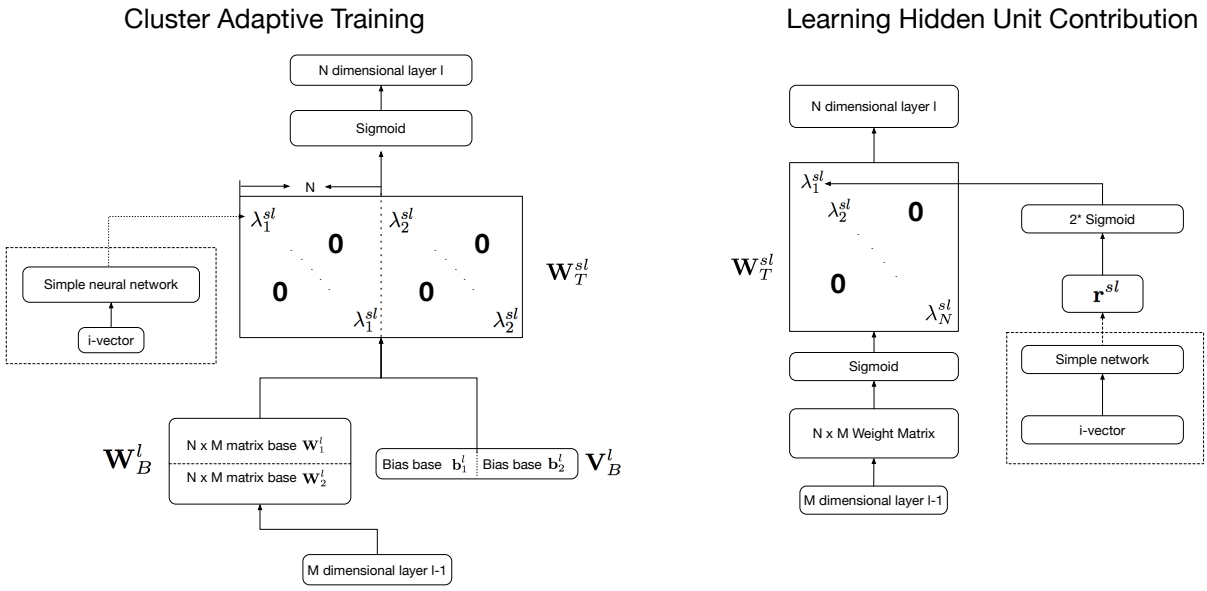


Figure 3.6: The common framework of CAT and LHUC. In CAT, an adaptation matrix $\boldsymbol{W}_T^{sl}$ is inserted upon the concatenated weight bases $\boldsymbol{W}_B^l$ and the bias. While in LHUC, a diagonal adaptation matrix is inserted above the Sigmoid activation.

$\boldsymbol{V}_B^l$, the interpolation then equals to a linear transformation $\boldsymbol{W}_T^{sl}$. The transformation is the concatenation of P scalar matrices and the scalar of each sub-matrices are $\lambda_1^{sl}, \lambda_2^{sl}, ..., \lambda_P^{sl}$ respectively. Therefore $\boldsymbol{W}_B^l$ and $\boldsymbol{V}_B^l$ are speaker-independent and are shared across all speakers. The interpolation matrix $\boldsymbol{W}_T^{sl}$ is speaker-dependent.

The operation $\boldsymbol{W}_c^l \boldsymbol{h}_{l-1}^s + \mathbf{b}_c^l$ can be considered as a sub-network consisting of only an affine transformation. It could be extended to multi-layers perceptron [Wu and Gales, 2015]. Moreover, if $\boldsymbol{\lambda}^{sl}$ is estimated from another network using $\boldsymbol{h}_{l-1}$ as input, it is exactly the same as (3.18) in MoE. It has been proposed to compute this interpolation vector through a network using i-vector, which conveys the speaker characteristic [Garcia-Romero and Espy-Wilson, 2011], as input to tackle speaker adaptation issues [Delcroix et al., 2016; Wu et al., 2016]. This idea is also very similar to the concept of mixture of experts.

One can also consider LHUC as a special case of mixture of experts which only contains one expert and $\xi(\mathbf{r}^{sl})$ is the gating function. In stead of generating weights of the experts, $\xi(\mathbf{r}^{sl})$ generates weights of each node inside the expert. The weighting process is operated over the nodes instead of the experts. In addition, LHUC inputs either a speaker-dependent vector $\mathbf{r}^{sl}$ or i-vector to the gating network [Samarakoon and Sim, 2016]. In MoE, usually the same input of the experts is used.

In summary, CAT and LHUC can be considered as particular cases of mixture of experts. They both insert another adaptable weight matrix to model the speaker variety. The differences

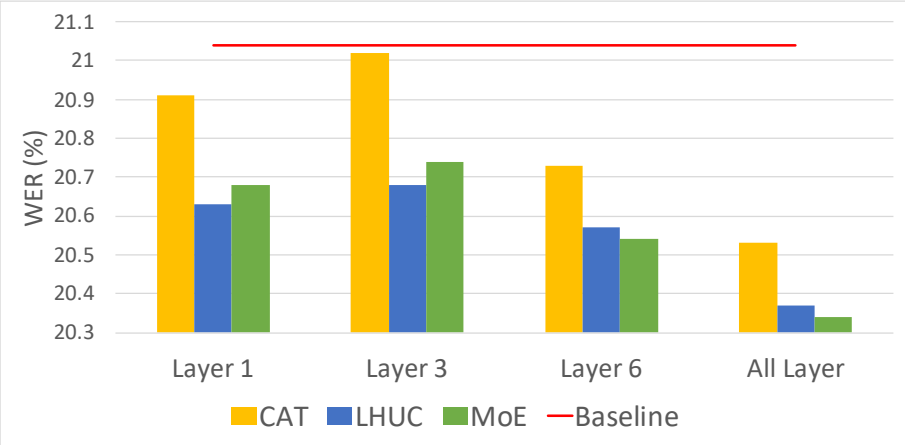


Figure 3.7: Comparison in WER(%) of LAT on different layers. The baseline is the IPA-based multilingual system.

lie on the place and the architecture of the speaker-dependent matrix. As discussed above, CAT inserts the adaptation matrix above the speaker-independent canonical bases and the adaptation matrix is the concatenation of a set of scalar matrices. While in LHUC, a diagonal matrix is inserted on top of the sigmoid layer for adaptation.

Compared with CAT and LHUC, MoE is potentially more suitable for language adaptive training. On the one hand, both CAT and LHUC apply adaptation according to the provided language information. Therefore, the same transformation will be applied for all the frames of a given utterance because the language identity is constant across one utterance. However, different languages share many basic pronunciations. Applying the same adaptation on the frames corresponding to those shared pronunciations may even hurt the performance. MoE uses the gating network to adaptively select the most appropriate transformation for each single frame by looking into the input data. On the other hand, language adaptive training is different from speaker adaptive training in the sense that much more adaptation data is available for each language. Therefore, using experts with stronger modeling capacity can better capture the language specificity.

3.2.7 Evaluation

In this section, we report experiments on the same GlobalPhone dataset [Schultz et al., 2013]. Similar to the last section, we used the French (FR), German (GE), Portuguese (PO), Russian (RU) and Spanish (SP) datasets from the GlobalPhone corpus.

For each language, we built standard maximum likelihood trained GMM/HMM systems, using 39-dimensional MFCC features (C0-C12, with delta and acceleration coefficients). A multilingual DNN using a universal IPA output layer was built. The number of the IPA-based multilingual context-dependent triphone states is 8000 with a total of 150K Gaussians. All the

Table 3.2: Detailed comparison between monolingual systems and multilingual systems with MoE in WER(%).

system	FR	GE	PO	RU	SP
monolingual model	23.2	16.6	19.9	28.8	9.0
SHL-MDNN	23.0	15.6	18.9	**28.3**	**8.6**
+MoE	22.9	**15.4**	19.0	28.4	9.0
MUL-IPA	23.3	18.5	19.3	30.4	9.8
+MoE	**22.8**	16.0	**18.3**	29.1	9.0

DNNs used in the experiments had six hidden layers, each consisting of 2,000 units and were trained from 11 consecutive frames after RBM pretraining. Kaldi speech recognition toolkit [Povey et al., 2011] was used to build all the systems.

Standard CAT, LHUC and MoE were conducted on different layers of the IPA-based universal networks. Three bases were used for CAT in all the experiments. Similarly, three experts were used for MoE and each expert is a sub-network with one hidden layer of the same size. The gating network takes both the current activation and a one-hot vector representing the language identity as input. Figure 3.7 describes the overall WER among all the five languages. It indicates that all the LAT approaches help improve the recognition performance. Adaptation on the last hidden layer looks more beneficial. However, LAT on the middle layer does not perform as well as that on the bottom or top layer. It seems that the adaptation on the first layer, which is the feature end, tries to remove the pronunciation variations across languages and the adaptation on the last layer encourages the abstract representation from the last hidden layer to be more discriminant for each language. This explains why language adaptive training on the first and the last layers works better. Applying LAT on all layers further improves the performance. MoE outperforms both LHUC and CAT while LHUC also performs better than CAT, which also demonstrates our hypothesis that more adaptation parameters would lead to more robust language specific modeling since more training data is available for language adaptation and the gating network in MoE is beneficial as it adaptively guides the network to use the appropriate expert according to the input data.

Table 3.2 lists the detailed comparison between monolingual systems and multilingual systems trained with MoE. The adaptation was conducted on all the hidden layers for both the IPA-based universal network and the SHL-MDNN. However, we did not observe similar gains from language adaptive training with the SHL-MDNN architecture. This can be attributed to the fact that, the language-specific output layers in SHL-MDNN play similar roles as language adaptive training; they model the language specificity. Thus, adding more language-specific parameters does not bring more gains. Due to the small amount of training data, overfitting was observed for some languages. By contrast, it is clear that the WERs of the IPA-based universal model are improved on all the languages by using MoE. The MUL-IPA model trained with MoE also outperforms the monolingual systems on almost all the languages. More importantly, language adaptive training helps bridge the gap between the MUL-IPA and the SHL-MDNN,

demonstrating the effectiveness of the proposed approach.

3.3 Conclusion

In this chapter, we investigated multilingual training based on state-of-the-art DNN/HMM hybrid systems. Multilingual training benefits from more training data but lacks of specialization. This drawback can be mitigated by applying language adaptive training. Various language adaptive training approaches were compared both theoretically and experimentally. Approaches such as LHUC and CAT, originating from speaker adaptive training, also work for language adaptation. They can be considered as particular cases of MoE which also gives the best performance in the experimental evaluations. Different from speaker adaptive training, much more data is available for each language to train the adaptation parameters. Applying language adaptive training on all the hidden layers is more beneficial and approaches that have stronger modeling capacity (such as MoE) perform better.

In state-of-the-art DNN/HMM hybrid systems, DNNs model context-dependent states to mitigate the error associated with the Markov assumption. This results in more challenges especially for cross-lingual ASR because of the large difference in context dependent labels arising from the phone set mismatch. In the next Chapter, we will explore new training frameworks for multilingual ASR directly modeling phonemes and investigate the cross-lingual adaptation based on the new frameworks.

4 Multilingual Training and Cross-lingual Adaptation in New Frameworks

We have shown in the last chapter that multilingual models for ASR can benefit from data in languages other than the target language. In traditional DNN/HMM hybrid framework, however, initialisation from monolingual *context-dependent* models leads to large mismatch of context-dependent states. End-to-end approaches are potential solutions to this as they performs well without requiring context-dependent states .

In this chapter, we first investigate Connectionist Temporal Classification (CTC) and end-to-end Lattice-free Maximum Mutual Information (LF-MMI) in the context of phoneme-based multilingual training. We provide comparisons with the Maximum Likelihood Estimation (MLE) training from both theoretical and practical aspect. Then, we take CTC training as a particular example of end-to-end modeling approach and investigate CTC training in the context of adaptation and regularisation techniques that have been shown to be beneficial in more conventional contexts. During cross-lingual adaptation, the idea of extending the multilingual output layer to new phonemes is introduced and investigated. In addition, we propose a novel parameter initialization approach for cross-lingual adaptation by incorporating phonological information. This chapter is a consolidation of Tong et al. [2018a][1], Tong et al. [2018b] and Tong et al. [2019a][2].

4.1 Connectionist Temporal Classification

In state-of-the-art DNN/HMM hybrid systems, DNNs model context-dependent states to mitigate the error associated with the Markov assumption. Consequently, training GMM/HMM systems and building decision trees to generate the clustered context-dependent states become a prerequisite procedure. However, this results in more challenges especially for cross-lingual ASR because of the large difference in context dependent labels arising from the phone set mismatch. According to Schultz and Waibel [2000], for example, 85% monophones in Portuguese can be covered by German, but the triphones coverage drops to 57%. Although

[1] ©2018 Elsevier
[2] ©2019 IEEE

approaches to adapt decision trees have been proposed by Schultz and Waibel [2000], the simple and effective way is to build a language-specific decision tree for the target language and replace the whole output layer of a DNN with the new targets, or to train a completely new network using bottleneck features.

In order to minimize this negative effects resulting from the large mismatch of context-dependent state, we investigate Connectionist Temporal Classification (CTC) approach [Graves et al., 2006] to remove training the prerequisite GMM/HMM model as well as building the decision tree. We can directly model phonemes or characters as it has been shown that monophone-based CTC systems can achieve equal or better performance than DNN/HMM hybrid systems when large amount of data is available [Sak et al., 2015; Miao et al., 2016].

CTC is an objective function for sequence labeling tasks without requiring any frame-level alignments between the input and the reference labels. It introduces a blank symbol representing the probability of not emitting any labels at a particular time step. The target label sequence can be extended by consecutively repeating each label and inserting the blank symbol to match the length of the input feature sequence. This extended intermediate representation is called the CTC *path*. Therefore, A CTC path is a sequence of labels at the frame level, allowing the blank symbol and the repetition of labels. Thus, one label sequence can be represented by a set of CTC paths that are mapped to it.

For an input feature sequence $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$, the conditional probability $P(\mathbf{y}|\boldsymbol{X}, \theta)$ is estimated by summing over the probabilities of all the possible paths that correspond to the target label sequence $\mathbf{y}$ after inserting the repetitions of labels and the blank symbols, i.e.,

$$p(\mathbf{y}|\boldsymbol{X}, \theta) = \sum_{\hat{\mathbf{y}} \in \Omega(\mathbf{y})} p(\hat{\mathbf{y}}|\boldsymbol{X}, \theta) = \sum_{\hat{\mathbf{y}} \in \Omega(\mathbf{y})} \prod_{t=1}^{T} p(\hat{y}_t|\boldsymbol{X}, \theta) \tag{4.1}$$

where $\Omega(\mathbf{y})$ denotes the set of all possible intermediate paths that correspond to $\mathbf{y}$ after repetitions of labels and insertions of the blank symbol and θ represents the model parameters. The conditional probability of the label at each time step, $P(\hat{y}_t|\boldsymbol{X}, \theta)$, is estimated using a neural network conditioned on the whole input sequence. The model can be trained to maximize (4.1) by using gradient descent, where the required gradients can be computed using the forward-backward algorithm [Graves et al., 2006].

As formulated by Zeyer et al. [2017], CTC can be identified as a particular case of the generalized MLE training procedure using the full-sum over the hidden state sequence. Recall that, as described in (3.3), the generalized HMM training optimizes the likelihood of observing $\boldsymbol{X}$ given a target sequence $\mathbf{W}$ with state sequences $\mathbf{S}$ as hidden variable and model parameters θ, given by:

$$p(\boldsymbol{x}|\mathbf{W}, \theta) = \sum_{s_1^T : \mathbf{W}} \prod_{t=1}^{T} p(\boldsymbol{x}_t|s_t, \theta) p(s_t|s_{t-1}) \tag{4.2}$$

In HMM/NN models, $p(\boldsymbol{x}_t|s_t,\theta)$ is modeled as

$$p(\boldsymbol{x}_t|s_t,\theta) \sim \frac{p(s_t|\boldsymbol{x}_t,\theta)}{p(s_t)} \tag{4.3}$$

In this context, comparing (4.1) and (4.2), CTC can be considered as a particular case of HMM MLE training with a special reduced HMM topology which has no transition probabilities, no state prior probability model but a special blank state, and is also optimized over all possible alignments using Baum-Welch soft alignments.

4.1.1 Multilingual CTC training

Many present-day languages evolved from common ancestors. It is therefore natural that they share some common graphemes and phonemes. Very recently, building multilingual speech recognition systems using a universal character (grapheme) set as output has been investigated [Kim and Seltzer, 2017; Toshniwal et al., 2017]. This could be a potential solution to build one system which is able to recognize multiple language without hints about the language identities. However, modeling graphemes includes implicit modelling of spelling, which requires large amount of data. Moreover, graphemes can differ a lot from language to language. Languages that have nothing in common in terms of graphemes also share some common phonemes. With this motivation, and following Imseng et al. [2011], we propose a multilingual architecture that uses a universal output label set consisting of the union of all phonemes from the multiple languages. This universal phone set can be either derived in a data-driven way, or obtained from the International Phonetic Alphabet (IPA). In this study, the monolingual phones are merged if they share the same symbol in the IPA table. The network is trained to model the universal phoneme targets using the CTC loss function on data from multiple languages.

We could also adopt the architecture which has a specific output layer for each language in multilingual training. However, the advantage of multilingual phoneme-based CTC training is that it removes the dependency of multilingual clustering of context-dependent states and thus allows us to train a universal multilingual model that is easily extensible to other languages. Therefore, we didn't provide investigation of this architecture for phoneme-based CTC training.

4.1.2 Related Work

Since the success of CTC training in ASR, there have been a few attempts to apply CTC training also in multi-accent and multilingual ASR. Yi et al. [2016] used phoneme labels for training a multi-accent CTC-based ASR system in a multitask setting. Rao and Sak [2017] trained grapheme-based acoustic models for multi-accent speech recognition using a hierarchical recurrent neural network architecture with CTC loss. Different from multi-accent ASR, phoneme set or grapheme set is not the same across languages in multilingual problems. Some pre-

Table 4.1: Comparison between CTC training and end-to-end LF-MMI for monolingual low-resourced ASR in WER(%).

system	FR	GE	PO	RU	SP
monolingual CTC	24.9	20.3	21.1	30.8	9.6
multilingual CTC	23.5	19.0	19.5	29.7	9.0

published work [e.g., Kim and Seltzer, 2017; Müller et al., 2017b] investigated the use of a universal grapheme set by merging identical graphemes shared among languages and train the model using CTC loss. However, learning the spelling directly from acoustic features still requires large amount of data and graphemes can differ a lot from language to language. Müller et al. [2017a] and their recent work [Müller et al., 2017b] investigated phoneme-based multilingual CTC training with respect to label error rate. In this section, we add to this knowledge base by also reporting word error rate (WER).

4.1.3 Evaluation

Experiments are reported on the same GlobalPhone dataset as the previous section. We used 40-dimensional log-mel filterbank coefficients as acoustic features together with their first and second-order derivatives, derived from 25 ms frames with a 10 ms frame shift. The features were normalized via mean subtraction and variance normalization on a speaker basis. All the monolingual phones were mapped to IPA symbols and we merged the phonemes from all the languages to create the universal phone set for multilingual training.

The multilingual CTC model has 4 layers of bidirectional LSTM (Bi-LSTM), with 320 cells in each layer and direction. All the weights in the models were randomly initialized and were trained using stochastic gradient descent with momentum. A learning rate of 0.00004 was used and early stopping on the validation set was applied to select the best model. For decoding, individual weighted finite-state transducer (WFST) decoding graphs were built using language-specific lexicons and language models. All the DNNs compared in this work have 6 hidden layers, each consisting of 1024 units. Thus, it contains slightly more parameters (8.8 vs 8.5 million) than the CTC models. All CTC models were trained based on the EESEN implementation [Miao et al., 2015] and DNN/HMM systems were built using the Kaldi [Povey et al., 2011] toolkit.

Results of multilingual CTC training are shown in Table 4.1; it is clear that multilingual training significantly outperforms monolingual training. CTC training on limited data tends to overfit. Thus, the monolingual performance is not as good as the DNN/HMM hybrid system reported in Section 3.1.2. Multilingual training can mitigate the overfitting to some extent and yields quite comparable results to the hybrid system. Therefore, multilingual phoneme-based CTC model can be a potential candidate, based on which the cross-lingual adaptation can be more straightforward and possibly yield better performance as we will see in the following sections.

4.2 End-to-end lattice-free MMI

CTC training is a particular case of maximum likelihood estimation. In this section, we move on to investigate Maximum mutual information (MMI). It aims to maximize the mutual information between the acoustic observation $\boldsymbol{X}$ and the word sequence $\mathbf{W}$:

$$\mathcal{L}_{MMI} = \log \frac{p(\boldsymbol{X},\mathbf{W})}{p(\boldsymbol{X})P(\mathbf{W})} \tag{4.4}$$

$$= \log \frac{p(\boldsymbol{X}|\mathbf{W})P(\mathbf{W})}{\sum_{\hat{\mathbf{W}}} p(\boldsymbol{X}|\hat{\mathbf{W}})P(\hat{\mathbf{W}})} - \log P(\mathbf{W}) \tag{4.5}$$

If the observation $\boldsymbol{X}$ and the word sequence $\mathbf{W}$ are completely independent according to the model, the equation equals 0, which implies $\boldsymbol{X}$ is unrelated to $\mathbf{W}$. Assuming $P(\mathbf{W})$ is independent of the model parameters θ, MMI can be simplified as:

$$\mathcal{L}_{MMI} = \log \frac{p(\boldsymbol{X}|\mathbf{W},\theta)P(\mathbf{W})}{\sum_{\hat{\mathbf{W}}} p(\boldsymbol{X}|\hat{\mathbf{W}},\theta)P(\hat{\mathbf{W}})}. \tag{4.6}$$

This actually maximizes the posterior probability. Therefore, this technique is also well known in literature as the Maximum a Posteriori (MAP) method [Dymarski, 2011]. Intuitively, it maximizes the probability of the ground truth transcription, while minimizing the probability of all other transcriptions. Thus, it is also considered as a sequence discriminative training criterion as it boosts the right answer and lessens the wrong ones in the sequence level.

Theoretically, the summation in the denominator should be calculated over all possible word sequences. However, this summation is usually constrained by the decoding lattice which contains the most possible hypothesis to reduce the computational cost [Valtchev et al., 1996; Woodland and Povey, 2002] and the discriminative training is normally a separate training phase after a cross-entropy model is trained as the seed model to generate the decoded lattice. Thus, the denominator can be approximated as:

$$\sum_{\hat{\mathbf{W}}} p(\boldsymbol{X}|\hat{\mathbf{W}},\theta)P(\hat{\mathbf{W}}) = \sum_{\hat{\mathbf{W}}} p(\boldsymbol{X}|\mathbb{M}_{\hat{\mathbf{W}}},\theta) \tag{4.7}$$

$$\approx p(\boldsymbol{X}|\mathbb{M}_{den},\theta) \tag{4.8}$$

where $\mathbb{M}_{den}$ is an HMM graph that includes all possible word sequences in the decoded lattices. This is called the denominator graph. More recently, Povey et al. [2016] applied MMI training with DNN/HMM models using a phone-based approximation to a full denominator graph by adopting a few different techniques such as using a phoneme-level language model (instead

of word-level language model) for the denominator graph to minimize the size of the graph so that the computation can be performed on GPUs. The phoneme-level language model for the denominator graph is a pruned n-gram model trained using the phone alignments of the training data. By doing so, decoded lattice generation for each utterance is no longer required before MMI training and the model can be trained from scratch using MMI criterion. In addition, they used a special acyclic HMM topology as the numerator graph to exploit the alignment information from a previous GMM/HMM model. More specifically, the numerator graph in the regular LF-MMI method is an expanded version of the composite HMM, where the amount of expansion of the self-loops for each utterance is determined according to its alignment [Povey et al., 2016]. This method is called regular lattice-free MMI (LF-MMI).

In the regular LF-MMI, the DNN outputs normally correspond to clustered context-dependent states, where the clustering is performed according to a decision tree. This tree is built using the alignments from an GMM/HMM system . It is still a hybrid training approach. By contrast, in the end-to-end LF-MMI proposed by Hadian et al. [2018], this prerequisite is removed by using monophones or full biphones. Moreover, the composite HMM (with self-loops) is used as the numerator graph instead of the special acyclic HMM used in regular LF-MMI.

As a result, different from the regular LF-MMI, the prior alignment information is not required and the neural network can learn the alignments freely from scratch. In order to train the phoneme-level language model for the denominator graph, the word sequences of the training transcriptions are converted to phoneme sequences based on the dictionary. The training starts from scratch without building GMM/HMM and generating the alignments. Instead, the alignments are implicitly learned during training. The required gradients can be computed using the forward-backward algorithm.

4.2.1 Multilingual Phoneme-based Model

Universal Phone Set

With the same motivation as mentioned in Section 4.1.1, and following our previous work [Tong et al., 2018a,b], we also adopt a multilingual architecture that uses a universal output label set consisting of the union of all phonemes from the multiple languages. We created a universal phone set by merging the monolingual phones which share the same symbol in the IPA table.

For multilingual end-to-end LF-MMI training, we trained a multilingual phoneme language model for denominator graph using the training transcriptions from all the multilingual data. The composite HMM graphs were created using the language-specific lexicons, and were used as the numerator graphs. In this sense, the numerator graph is language-specific while the denominator graph is multilingual.

Biphone Modelling and Pruned Biphone Tree

Although monophone-based end-to-end training fits well for multilingual ASR because of its simplicity, it is well known that context-dependent modelling further improves the performance. In this sense, using full biphones can be a good compromise. It has been shown that context-dependent modelling also helps in end-to-end LF-MMI training Hadian et al. [2018]. This was implemented as a trivial full biphone tree. This tree is not pruned at all and does not have any tying, so there is no need for alignments and the approach does not require any previously trained models. However the size of the biphone targets grows quadratically in a multilingual set-up. A lot of cross-lingual biphone combinations will be created which never occur in the training data, impacting the training efficiency. Therefore, we propose to build a pruned biphone tree where all the cross-lingual biphone combinations are pruned away. More specifically, suppose a language has a phone set of $\{a, b, c\}$ and another language has a phone set of $\{b, c, d\}$. The universal phone set would be $\{a, b, c, d\}$. When creating the biphone targets, combinations such $a - d$ and $d - a$ will also be generated. However, they will never appear in the training data and are pruned away in this work.

4.2.2 Evaluation

The same GlobalPhone dataset is used to compare multilingual training and monolingual baseline. We used 40-dimensional MFCC as acoustic features, derived from 25 ms frames with a 10 ms frame shift. The features were normalized via mean subtraction and variance normalization on a speaker basis. All the monolingual phones were mapped to IPA symbols and we merged the phonemes from FR, GE, PO, RU and SP to create the universal phone set for multilingual training. For end-to-end LF-MMI training, 8 layers of Time Delay Neural Network (TDNN) was used, with 550 nodes in each layer. The network parameters are initialized randomly to have zero mean and a small variance. All end-to-end LF-MMI systems were built using the Kaldi toolkit [Povey et al., 2011].

We have shown that multilingual training is effective in traditional DNN/HMM training and CTC training. We further investigated multilingual training in the end-to-end LF-MMI framework. For multilingual biphone modelling, the pruned biphone targets were used as described in Section 4.2.1. The number of biphone targets was reduced from 23980 to 13776. The models were trained using data from all the 5 languages.

Table 4.2: Comparison between multilingual end-to-end LF-MMI in WER(%).

system	FR	GE	PO	RU	SP
monophn LF-MMI	23.6	18.7	18.6	26.6	9.3
biphn LF-MMI	23.5	17.0	18.2	25.8	8.5
ML monophn LF-MMI	23.2	15.4	17.0	24.9	7.9
ML biphn LF-MMI	23.2	16.0	17.9	25.1	7.7

From the table we can find that multilingual LF-MMI training yields significant improvement

over monolingual LF-MMI training for both monophone and biphone-based LF-MMI. Discriminative training always has a problem of over-estimation [Watanabe and Chien, 2015]. Therefore, it benefits from more training data during multilingual training. However, different from the monolingual cases, the multilingual biphone LF-MMI performs worse than multilingual monophone model in most of the tested languages. We hypothesize that biphone targets cover more variabilities compared to the corresponding monophone, especially when they are shared by mutiliple languages. As shown in the last chapter, language-specific characteristics cannot be well modeled by an IPA-based universal network. Language adaptive training can mitigate this problem as have been shown in Chapter 3. Since monophone modeling performs well even without language adaptive training, we will continue to focus on monophone modeling.

4.3 Comparison of MLE, CTC and LF-MMI

In this section, we summarize and make comparison among these training criteria discussed so far, namely MLE, CTC and MMI, from both theoretic and practical aspects. Recall that the loss function of HMM-based maximum likelihood training can be written as:

$$\mathcal{L}_{MLE} = \log p(\boldsymbol{X}|\mathbb{M}_{\mathbf{W}},\theta) \tag{4.9}$$

$$= \log \sum_{\mathbf{s}\in\mathbb{M}_{\mathbf{W}}} \prod_{t=1}^{T} p(\boldsymbol{x}_t|s_t,\theta)\, p(s_t|s_{t-1}) \tag{4.10}$$

where the composite HMM graph $\mathbb{M}_{\mathbf{W}}$ represents all the possible state sequences $\mathbf{s}$ pertaining to the transcription $\mathbf{W}$.

Similarly, the loss function of CTC can be written as:

$$\mathcal{L}_{CTC} = \log \sum_{\mathbf{s}\in\Omega(\mathbf{W})} \prod_{t=1}^{T} p(s_t|\boldsymbol{x}_t,\theta) \tag{4.11}$$

where $\Omega(\mathbf{W})$ denotes the set of all possible paths that correspond to $\mathbf{W}$ after repetitions of labels and insertions of the blank token.

The end-to-end LF-MMI criterion can be written as:

$$\mathcal{L}_{MMI} = \log \frac{p(\boldsymbol{X}|\mathbf{W},\theta)P(\mathbf{W})}{\sum_{\hat{\mathbf{W}}} p(\boldsymbol{X}|\hat{\mathbf{W}},\theta)P(\hat{\mathbf{W}})} \tag{4.12}$$

$$= \log \frac{p(\boldsymbol{X}|\mathbb{M}_{\mathbf{w}},\theta)}{\sum_{\hat{\mathbf{W}}} p(\boldsymbol{X}|\mathbb{M}_{\hat{\mathbf{W}}},\theta)} \tag{4.13}$$

Comparing (4.9) and (4.11), it is clear that CTC training is a particular case of HMM-based

Table 4.3: Comparison among MLE, CTC and LF-MMI for low-resourced ASR in WER(%).

system	FR	GE	PO	RU	SP
monolingual hybrid	23.2	16.6	19.9	28.8	9.0
ML hybrid	23.3	18.5	19.3	30.4	9.8
monolingual CTC	24.9	20.3	21.1	30.8	9.6
ML CTC	23.5	19.0	19.5	29.7	9.0
monolingual LF-MMI	23.6	18.7	18.6	26.6	9.3
ML LF-MMI	23.2	15.4	17.0	24.9	7.9

maximum likelihood training. CTC uses a special reduced HMM topology which has no transition probabilities, no state prior probability model but a special blank state. The CTC loss is trained with Baum-Welch soft alignments from scratch opposed to traditional hybrid framework where the network is trained using hard alignment generated by GMM/HMM. This also explains why CTC training outperforms DNN/HMM hybrid systems given adequate amount of training data.

Comparing (4.9) and (4.13), LF-MMI is a sequence-discriminative training approach which also simultaneously minimizes the probability of all other transcriptions. This discriminative nature encourage the model to create cleaner decision boundaries compared with MLE training. Maximum likelihood is theoretically optimal, but only when the model is correct. When having the independence assumptions, an explicitly discriminative training criterion might be better.

Table 4.3 summarizes the comparison among these training criteria in the context of multilingual training from the practical aspect. All the multilingual models adopt IPA-based universal outputs. Both CTC and LF-MMI training use monophone as the modeling target. The experiments were conducted on the same GlobalPhone dataset, each language containing roughly 20 hours training data. When trained monolingually, hybrid training outperforms CTC training as CTC trains the model from scratch without alignment, which requires relatively more training data to learn plausible alignments [Miao et al., 2016]. Training on small amount of data tends to overfit. LF-MMI yields better results than the hybrid system and CTC training because the training is also discriminative. This implies that a more rigorously derived Bayesian approach is beneficial. Compared with the hybrid systems, multilingual training helps more for CTC and LF-MMI training since they are more sensitive to the amount of data compared with hybrid training. Different languages can benefit from each other when data from multiple languages is pooled together. Overall, the best WERs are obtained by multilingual LF-MMI training.

4.4 Cross-lingual Adaptation on CTC Model

When applied to acoustic modeling, CTC and end-to-end LF-MMI allows the model to automatically learn the alignments between acoustic features and labels. Thus, they remove the need for building the initial GMM to generate frame-level labels. In addition, we have

shown that phoneme-based modeling using CTC and end-to-end LF-MMI achieve competitive performance compared with the conventional hybrid training. It models phonemes in the output layer which can be easily transferred and extended to new languages. Therefore, in the following sections, we take CTC training as an example to investigate the cross-lingual adaptation from multilingual phoneme-based models.

4.4.1 Related Work

The cross-lingual ability of the CTC model has not been well studied. Kunze et al. [2017] shows a low-resource grapheme-based system can be initialized with a well-trained high-resourced model. In another pre-published work [Scharenborg et al., 2017], an iterative method is proposed to build a CTC-based ASR system for low-resourced languages, where the high-resourced model is iteratively adapted to the target language using the phoneme transcription generated from the adapted model. After independently investigating the CTC-based cross-lingual adaptation, we found that similar ideas had been very recently studied by Dalmia et al. [2018]. However, The author used a multi-task multilingual CTC; the output consists of separate activations for each language. By contrast, our multilingual CTC system models the IPA-based universal phoneme set, and therefore it has the unique property that the output layer can be easily extended to new languages. Furthermore, this section discusses dropout in the CTC-based cross-lingual adaptation and provides comparisons with DNN-based framework.

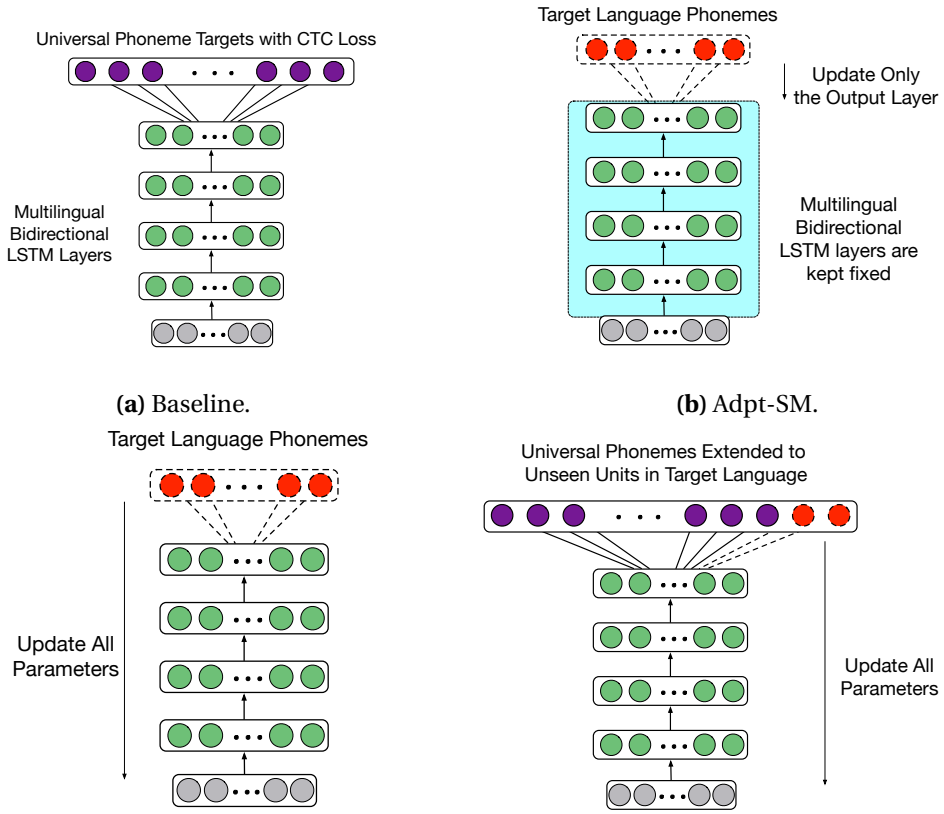
4.4.2 Universal Phone Set Multilingual CTC Model

The main goal of multilingual acoustic modelling is to share the acoustic data across multiple languages in order to learn the common properties shared among languages, which can be transferred and utilized for low-resourced languages. With the same motivation described in Section 4.1.1, we propose to train an IPA-based multilingual model that uses a universal output label set consisting of the union of all phonemes from the multiple languages and then, investigate cross-lingual adaptation from the universal model. The monolingual phones are merged if they share the same symbol in the IPA table. In this context and different from conventional context-dependent states modeling, knowledge about the shared phonemes learned from multilingual training can be directly transferred to the target language.

4.4.3 Adaptation Strategies

In the DNN framework, the shared hidden layers extracted from the multilingual DNN can be considered to be an intelligent feature extractor and are transferable across languages [Huang et al., 2013]. It is therefore interesting to investigate if the hidden layers in a CTC-based model can be carried over to distinguish phonemes in new languages.

The basic procedure of cross-lingual model adaptation on a CTC model is simple. The output



(a) Baseline. **(b)** Adpt-SM.

(c) Adpt-ALL. **(d)** Adpt-EXT-ALL.

Figure 4.1: Approaches to adapt multilingual CTC model to the target language. (a) shows the baseline multilingual CTC model. In (b), a new *softmax* (SM) output layer replaces the multilingual targets. The hidden layers are fixed and only the output layer is re-estimated. We can also update all the parameters as shown in (c). In (d), the multilingual CTC model is extended to new phonemes by adding new connections. Adaptation is performed by updating all the parameters.

layer of the seed model is removed and a new randomly initialized softmax (SM) layer, corresponding to the target language phone set, is added on top of the hidden layers. Usually the hidden layers are fixed and only the softmax layer will be re-estimated using training data from the target language. If enough data is available, further tuning of the entire network can be considered.

One major advantage of the universal phoneme-based multilingual CTC model over conventional hybrid systems that model triphones is that monophone modeling gets around the problem of mismatch of the clustered context-dependent states. It therefore becomes straightforward to extend the existing multilingual model to extra phonemes when a new target language arrives. Therefore, we propose to extend the multilingual output layer by adding connections to the unseen mono phones of the target language, rather than discarding

Table 4.4: Statistics of the dataset of each language used in this work: the amounts of speech data are in hours.

Application	Language	Dataset	Train	Dev	Test
Multilingual Training	EN	WSJ	81h	1.1h	1.1h
	FR	BREF/GP	120h	10.3h	8.8h
	GE	BCN	136h	1.1h	5.7h
	Total Amount		337h		
Cross-lingual Adaptation	PO	GP	21h	1.6h	1.8h

all the information already learned in the output layer. As is shown in Figure 4.1, those weights connecting to the unseen phones are randomly initialized and trained from scratch. The others can be quickly adapted from the multilingual model with little adaptation data.

4.4.4 Regularization Using Dropout

In our preliminary experiments with CTC, overfitting was observed on limited data. Although multilingual training mitigates overfitting to some extent, the problem still exists. Dropout has been well established for feed forward networks by Srivastava et al. [2014], and it has been also proved to significantly improve the performance of LSTM networks for sequence labelling tasks [Reimers and Gurevych, 2017]. More recently, various approaches of dropout on feedforward and recurrent connections were explored in the context of CTC [Billa, 2017]. Inspired by this work, we propose to combine dropout with both multilingual training and cross-lingual adaptation to minimize overfitting on limited data. Moreover, we hypothesize applying dropout in multilingual training has an additional advantage: It can help the model avoid being overfitted in an optimum specific to any languages, thus making the model more language-independent.

4.4.5 Experimental Setup

We trained the performance of the proposed universal phoneme-based CTC model on English (EN), French (FR), and German (GE). The English data was obtained from the Wall Street Journal (WSJ) corpus. Data preparation gave us 81 hours of transcribed training speech. WSJ dev93 and the union of eval92 and eval93 were used as the development set and the evaluation set, respectively. The French data was extracted from the BREF and GlobalPhone (GP) corpora, which consist of 120 hours of data. From the German Broadcast News (BCN) corpus, we used 136 hours of data for training. In total, 337 hours of multilingual data was used for multilingual CTC training. All the training data is quite clean read speech from similar acoustic conditions. In cross-lingual adaptation experiments, Portuguese (PO) from GlobalPhone was considered as the target low-resourced language, which has only 21 hours data. The detailed statistics for each of the languages are shown in Table 6.1. The development sets were used to tune the hyper-parameters for training.

Table 4.5: Comparison between monolingual CTC baseline systems and multilingual CTC training in WER(%). Notice that the English test set is much smaller than those in French and German. However, we only use it to indicate trends, drawing more concrete conclusions from the French and German results. Dropout is not applied.

	system	EN	FR	GE
	ML-DNN-LHUC	8.8	7.3	8.6
	monolingual CTC	9.5	8.5	8.9
sys 1	universal ML-CTC	9.6	8.1	9.0
sys 2	+LHUC	9.2	7.7	8.4

Table 4.6: Comparison between monolingual CTC baseline systems and multilingual CTC training in WER(%). Dropout is applied.

	system trained w/ dropout	EN	FR	GE
	monolingual CTC	9.2	7.7	8.7
sys 3	universal ML-CTC	9.4	7.8	8.3
sys 4	+LHUC	**8.9**	**7.4**	**7.8**

We used 40-dimensional log-mel filterbank coefficients as acoustic features together with their first and second-order derivatives, derived from 25 ms frames with a 10 ms frame shift. The features were normalized via mean subtraction and variance normalization on a speaker basis. All the monolingual phones were mapped to IPA symbols and we merged the phonemes from EN, FR and GE to create the universal phone set for multilingual training. Note that we removed the stress makers in EN phone set in order to map the phonemes to IPA symbols.

The multilingual CTC model has 4 layers of Bi-LSTM, with 320 cells in each layer and direction. All the weights in the models were randomly initialized and were trained using stochastic gradient descent with momentum. A learning rate of 0.00004 was used and early stopping on the validation set was applied to select the best model. For decoding, individual weighted finite-state transducer decoding graphs were built using language-specific lexicons and language models. All the DNNs compared in this work have 6 hidden layers, each consisting of 1024 units. Thus, it contains slightly more parameters (8.8 vs 8.5 million) than the CTC models. All CTC models were trained based on the EESEN implementation [Miao et al., 2015] and DNN/HMM systems were built using the Kaldi [Povey et al., 2011].

4.4.6 Multilingual CTC Training

In Section 4.1.3, we have shown that multilingual training is very beneficial when only small amount of data is available for each language (99 hours training data in total). We first evaluated if this conclusion still holds when using more training data (337 hours training data in total). The comparison between multilingual CTC and baseline monolingual CTC systems is listed in Table 4.5. The table shows that multilingual CTC system sometimes fails to

outperform monolingual models. It seems multilingual training is less helpful when adequate amount of data is available for each language and the multilingual model starts to suffer from the data impurity arising from mixture of multilingual data. This motivates us to apply language adaptive training in the multilingual CTC model. LHUC was applied on top of each bidirectional LSTM layer as it is easy to implement and faster to train. As shown in Table 4.5, applying LHUC improves the multilingual performance and yields better WER than the monolingual CTC models in all languages.

It has been reported that dropout can help overcome the overfitting problem in monolingual CTC training [Billa, 2017]. Dropout was further tested in multilingual conditions as described in Section 4.4.4 and the dropout rate was set to 0.2. Comparing Table 4.6 and Table 4.5, we can find that overfitting problem still exists in multilingual CTC training and dropout can help improve the generalization of the multilingual model. The systems trained with dropout consistently outperform the corresponding non-dropout systems in all languages. Combining LHUC and dropout yields the best performance.

Table 4.5 also lists the performance of the DNN-based multilingual training. Both models were trained on the same multilingual data with IPA labels. The IPA-based labels for the CTC training were obtained from the context-dependent state alignments of the multilingual GMM/HMM model. LHUC was also applied on top of each layer. Our experiment shows that dropout cannot improve DNN-based acoustic modeling. Therefore, dropout was not applied. The comparison shows multilingual CTC training can achieve competitive performance with DNN-based multilingual training.

4.4.7 Dropout in Cross-lingual Adaptation

While the first goal of this work was to create a universal phoneme-based multilingual model, we were interested in its transfer ability to other languages when the training data is limited. Previous experiments show that dropout is helpful in CTC training. We hypothesize that dropout can also improve cross-lingual adaptation where the available data is even more limited. In the present experiment, the multilingual model **sys 1** in Table 4.5 was used as the seed model, and cross-lingual adaptation was performed on limited amounts of Portuguese training data. The adaptation was done simply by replacing the multilingual output layer with a new output layer corresponding to the Portuguese phonemes and updating all the parameters. The same dropout strategy was tested on different amounts of adaptation data. As shown in Figure 4.2, although the improvement becomes smaller when more data is available, dropout consistently improves the adaptation performance. Similar improvements were also observed in the adaptation from other multilingual models and using different adaptation approaches in our experiments. Therefore, we keep applying dropout in the remaining cross-lingual adaptation experiments.

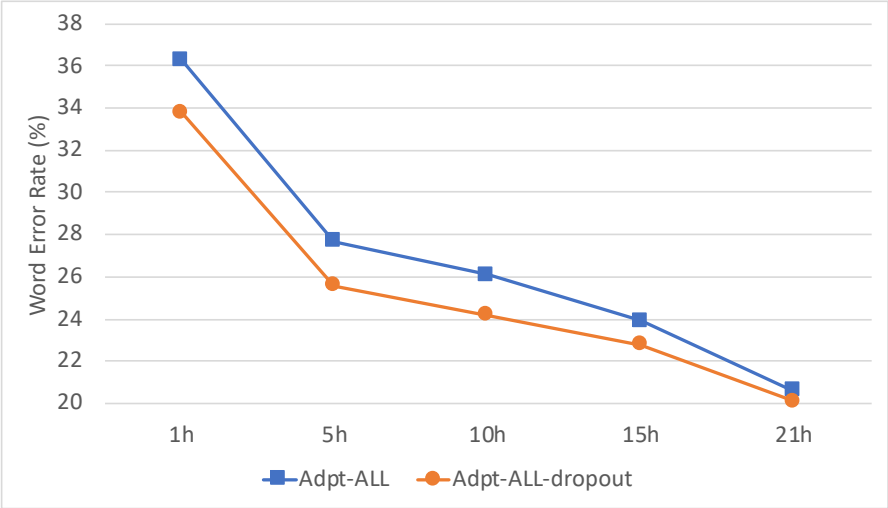


Figure 4.2: WERs (%) after cross-lingual adaptation with or without dropout.

4.4.8 Which Is the Best Seed Model for Cross-lingual Adaptation

The next problem is to choose the best multilingual model to initialize cross-lingual adaptation. In this work, the multilingual models **sys 1**, **sys 3** and **sys 4** were tested. We omitted **sys 2** as we have no a-priori belief that it will outperform **sys 4**. The adaptation was done simply by replacing the multilingual output layer with a new output layer corresponding to the Portuguese phonemes and updating all the parameters. When adapting an LHUC multilingual model, two approaches were compared: 1) updating the whole network after removing the LHUC layers and, 2) re-estimating Portuguese-specific LHUC parameters and the *softmax* (SM) output layer while keeping the rest fixed. In comparison with the latter one, adapting only the output layer from **sys 3** was also tested.

Comparing the **sys1-ALL** and **sys3-ALL**, we can clearly find that adaptation from the dropout multilingual model performs better. One conjecture is that dropout can help the multilingual model avoid being overfitted in a language-specific optimum and captures language-independent information better. Comparing **sys3-ALL** and **sys4-ALL**, we observed that the multilingual model trained with LHUC yields slightly better WER than the non-LHUC multilingual training when adapted to a new language, although the improvement is not significant. We did not report the performance of updating the LHUC parameters in addition to the whole network from **sys 4** because we found it is not helpful since the LHUC layers are merely additional adaptation parameters and may lead to overfitting.

Ideally, re-estimating only the LHUC parameters for Portuguese while keeping the rest fixed allows the adapted model to keep the performance on EN, FR and GE. However, adapting LHUC parameters as well as the output layer (**sys4-LHUC+SM**) performs already much worse than updating the whole network on the target language. Nevertheless, it yields improvement over updating only the output layer (**sys3-SM**), which still demonstrates the benefit of using

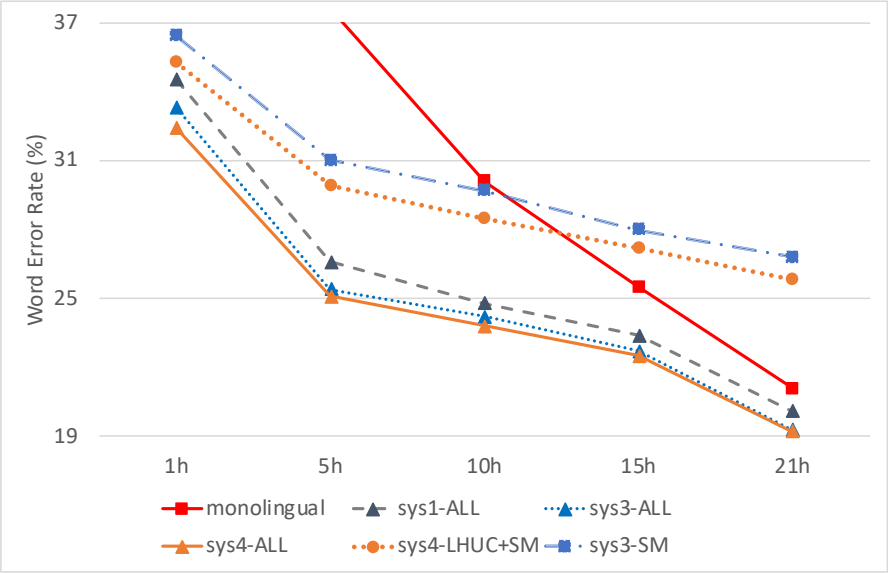


Figure 4.3: WERs (%) after cross-lingual adaptation of different multilingual models on various amounts of data. Dropout is applied in all systems. sys1-ALL denotes adapting all the parameters from **sys1**. sys4-ALL is updating the whole network after removing the LHUC layers. sys4-LHUC+SM represents adapting only the Softmax output layer and the LHUC parameters from **sys4**. sys3-SM is adapting only the output layer from **sys3**.

an LHUC-based seed model. Given the above observation, **sys 4**, trained on 3 languages using LHUC and dropout, was used as the seed model for the following cross-lingual experiments.

4.4.9 Output Layer Extension in CTC-based Cross-lingual Adaptation

Although Figure 4.3 shows that updating all the parameters performs better than updating only the output layer, it is still worth investigating their performance after output layer extension. Therefore, four approaches were investigated in this section: re-training a new output layer and the LHUC parameters while keeping the others fixed (Adpt-LHUC+SM); extending the multilingual model by concatenating parameters corresponding to the new phonemes to the output layer and then updating the extended output layer and the LHUC layers (Adpt-EXT-LHUC+SM); updating the whole network with a randomly initialized new output layer (Adpt-ALL in Figure 4.1c); updating the whole network after extending the multilingual output layer to the target language (Adpt-EXT-ALL in Figure 4.1d). Experiments on different amounts of data were conducted using these approaches. Figure 4.4 shows all the comparisons.

From the figure, it can be found that adapting the whole network outperforms monolingual CTC training in all cases. It is difficult to train a good CTC model from scratch using less than 5 hours of data. However, the adaptation from a multilingual model can still achieve good performance. When the adaptation data is more than 15 hours, monolingual training beats the adaptation on only the output layer and the LHUC layers. Moreover, updating all

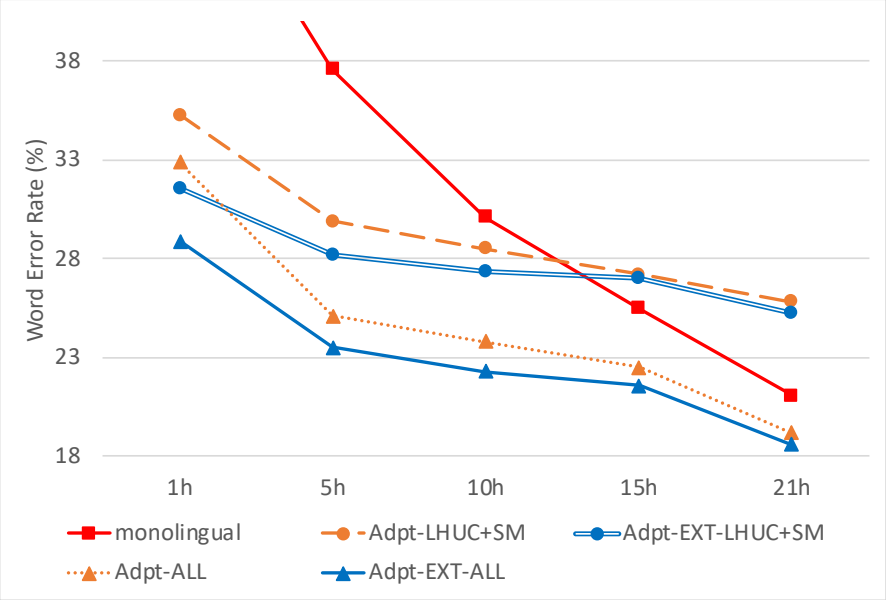


Figure 4.4: WERs (%) of different cross-lingual adaptation approaches. The WER of monolingual CTC model on 1 hour data is above 50% and exceeds the graph region. All models were trained with dropout.

the parameters still performs better than only re-training the output layer and the LHUC layers in all cases. We hence make the anecdotal inference that the Bi-LSTM layers are more interdependent than those of the DNN [Huang et al., 2013]; stronger inference would require more focused experiments. If we compare the blue lines and the orange ones, consistent improvement can be observed from extending the multilingual output layer. Although the difference becomes marginal with the increase of the adaptation data, it yields about 12% relative improvement on 1 hour adaptation data.

There are 19 extra unseen phonemes in Portuguese while 26 phonemes have been observed in multilingual training. As an example, we analyzed the phoneme error rate (PER) with respect to the overlapped phonemes and the new, unseen, phonemes separately on the development set during CTC training. The analysis was conducted on both adapting all the parameters and only the output layer plus LHUC layers, as plotted in Figure 4.5 and Figure 4.6. It shows that adaptation after extending the multilingual output layer keeps the same performance on unseen phonemes and converges much faster and better on seen phonemes. Although the adaptation data is limited, the extended model already has strong knowledge about the overlapped phonemes learned from multilingual training, and it is also able to catch up on new phonemes quickly.

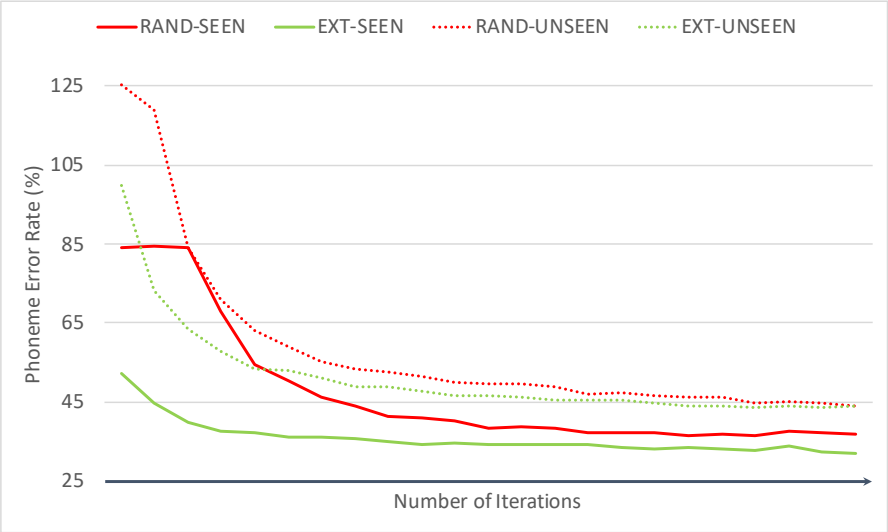


Figure 4.5: PERs (%) with respect to overlapped phonemes (SEEN) and new phonemes (UNSEEN) on PO development set. RAND denotes randomly initializing a new output layer before adaptation and EXT represents extending the multilingual output layer to the target language. The adaptation was performed by updating only the output layer and the LHUC layers on 1 hour data.

4.4.10 Comparison with DNN-based Cross-lingual Adaptation

We also compared our best CTC-based cross-lingual adaptation with DNN/HMM-based adaptation approaches, as depicted in Figure 4.7. In the DNN-based adaptation, the multilingual DNN trained on the same multilingual data was used as seed model. We then replaced the multilingual output layer with Portuguese targets. The Portuguese context-dependent states and alignments were obtained from GMM/HMM systems trained on the corresponding amount of adaptation data. The adaptation was then performed by 1) updating the whole network, 2) Estimating the new output layer plus the LHUC layers while keeping the other parameters fixed and 3) updating only the output layer. Dropout was not applied for DNN since performance degradation was observed with dropout in our experiments.

As shown in the Figure 4.7, if comparing the DNN-based cross-lingual adaptation approaches, we can find that updating the output layer together with the LHUC parameters generally outperforms only updating the output layer, except for the 1 hour data case. Updating all the parameters performs better than updating the output layer and the LHUC layers when more data is available but the difference is not significant. Meanwhile, updating the whole DNN also performs better than the CTC-based cross-lingual adaptation when adaptation data is more than 5 hours. However, CTC-based adaptation outperforms DNN/HMM based approaches when data is less than 3 hours. The CTC model retains the information about the phonemes that have been well modeled in multilingual training. Thus, the model can be easily transferred and adapted. The adapted model performs better than retraining the output layer from scratch in DNN. Given the fact that CTC training outperforms DNN/HMM

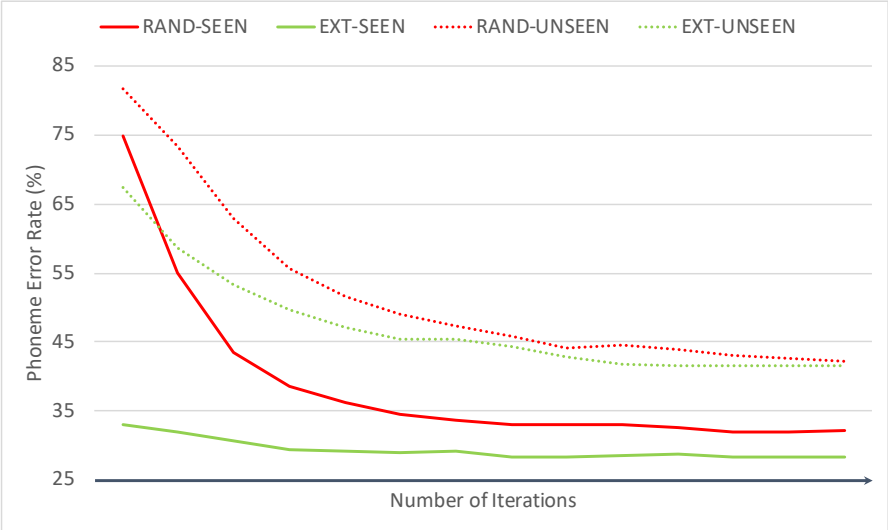


Figure 4.6: PERs (%) with respect to overlapped phonemes (SEEN) and new phonemes (UNSEEN) on PO development set. The adaptation was performed by updating the whole network on 1 hour data.

hybrid modeling when sufficient data is available, we hypothesize CTC-based cross-lingual adaptation can surpass DNN-based approaches again if more data can be used for adaptation. We leave this for the future work.

4.5 Weights Initialization using Phonological Information

We have shown that phoneme-based multilingual CTC model is easily extensible to a new language by concatenating parameters of the new phonemes to the output layer. In the following sections, we improve cross-lingual adaptation in the context of phoneme-based CTC models by using phonological information. An IPA phoneme classifier is first trained on phonological features generated from a phonological attribute detector. When adapting the multilingual CTC model to a new, never seen, language, phonological attributes of the unseen phonemes are derived based on phonology and fed into the phoneme classifier. Posteriors given by the classifier are used to initialize the parameters of the unseen phonemes when extending the multilingual CTC output layer to the target language. Adaptation experiments show that the proposed initialization approaches further improve the cross-lingual adaptation on CTC models and yield significant improvements over DNN/HMM-based adaptation using limited data.

4.5.1 Phonological Feature-based Phoneme Classifier

As shown in Figure 4.8, the proposed phoneme classifier consists of two main blocks: 1) a data-driven phonological attribute detector, and 2) a frame-based phoneme classifier using

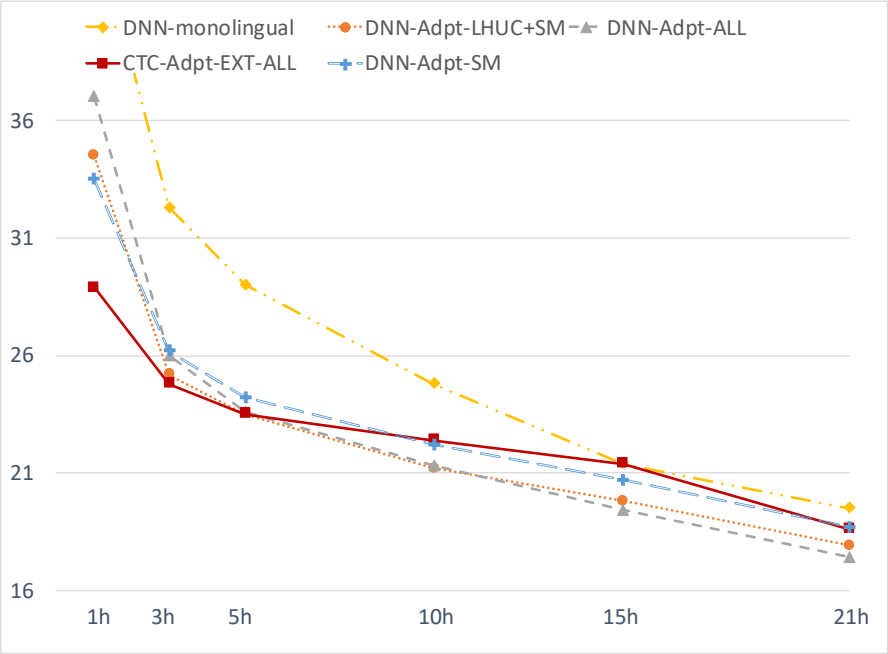


Figure 4.7: Comparison between CTC-based and DNN/HMM-based cross-lingual adaptation in WER(%). DNN-Adpt-ALL denotes updating all the parameters in DNN and DNN-Adpt-LHUC+SM represents updating the output softmax layer and the LHUC layers. DNN-Adpt-SM is only updating the output layer. The WER of monolingual DNN model on 1 hour data is above 40% and exceeds the graph region.

phonological features generated from the previous detector.

The phonological attribute detector is a multitask-learning DNN for joint estimation of phonological features. Estimating different phonological features from the same acoustic signal can be considered as a set of interrelated tasks; it has been shown effective for articulatory feature estimation in the work of Rasipuram and Magimai-Doss [2011]. To estimate the DNN parameters, multilingual training data is used. The labels for every phonological class are generated from the phoneme alignment according to the phonological mapping[3].

Once the phonological detector is trained, the phonological posteriors gathered from the detectors can be viewed as an indication that a specific phone has been articulated. In this work, the log posteriors of every phonological class are concatenated together and fed into the phoneme classifier, which is realized using a DNN. The outputs of the DNN are the monophone targets of the same IPA-based phoneme set used in multilingual CTC training. For each unseen phoneme in a target language, the phoneme classifier will be utilized to find the most probable mappings in the multilingual phoneme set.

[3]http://publications.idiap.ch/downloads/reports/2018/Tong_Idiap-Com-02-2018.pdf

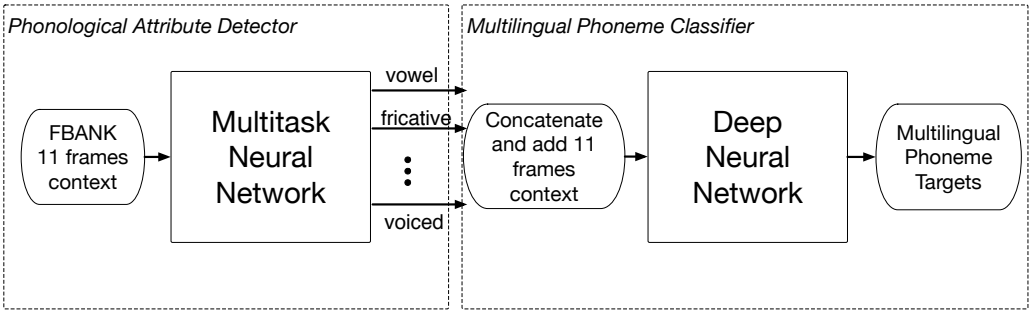


Figure 4.8: Architecture of the multilingual phoneme classifier using phonological features.

4.5.2 Parameter Initialization Using Multilingual Phoneme Posterior

When extending the multilingual CTC network to a new language, a better initialization of the parameters connecting to those unseen phonemes can be estimated using the phonological attribute-based phoneme classifier described above. For an unseen phoneme u, the corresponding phonological attributes can be obtained from prior knowledge. Inputting the phonological attributes to the phoneme classifier produces multilingual phoneme posterior $\mathbf{P}(u) = [p_1(u), p_2(u), ..., p_N(u)]$, where N denotes the size of the multilingual phoneme set. The posterior $\mathbf{P}(u)$ can be interpreted as how close the new phoneme is to those seen multilingual phonemes. In the extended output layer, the weights $\boldsymbol{\omega}_u$ and the bias b_u of the unseen phoneme u can be initialized either by taking a weighted average of the parameters of all the seen multilingual phonemes,

$$\boldsymbol{\omega}_u = \sum_{i=1}^{N} p_i(u)\boldsymbol{\omega}_i, b_u = \sum_{i=1}^{N} p_i(u) b_i \tag{4.14}$$

where $\boldsymbol{\omega}_i$ and b_i represent the weight and the bias of the i^{th} phoneme respectively, or by copying the weight and bias of the multilingual phoneme that has the maximum posterior.

$$\boldsymbol{\omega}_u = \boldsymbol{\omega}_m, b_u = b_m, \text{where } m = \underset{i}{\text{argmax}}(p_i(u)) \tag{4.15}$$

4.5.3 Experimental Setup

Similarly, the multilingual seed model was trained on the same English, French and German data. In total, 337 hours of multilingual data was used for multilingual CTC training. All the training data is quite clean read speech from similar acoustic conditions. In cross-lingual adaptation experiments, GlobalPhone Portuguese (PO) was considered as the target low-resourced language, which has only 21 hours data.

We used the same feature, network architecture, training strategy as the experiment in the

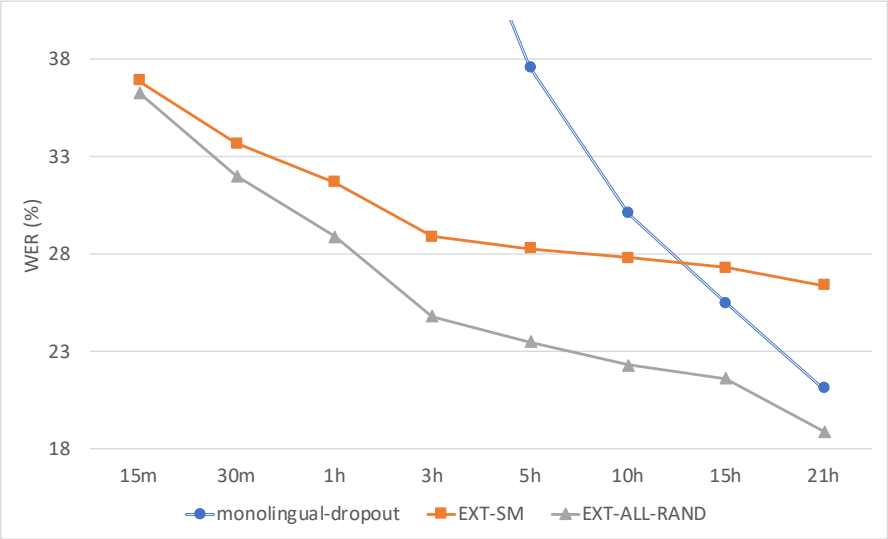


Figure 4.9: WERs (%) of different approaches in cross-lingual adaptation. The WERs of monolingual CTC models on less than 5 hours data are above 50% and exceed the graph region.

previous section. Once the multilingual model was trained, it was used as seed model for cross-lingual adaptation to Portuguese. A similar training strategy was applied. For decoding, a weighted finite-state transducer decoding graph was built using a language-specific lexicon and language model. The trigram language models that we used are publicly available. All the DNN/HMMs compared in this work also have 6 hidden layers, each consisting of 1024 units, which results in similar amount of parameters to the CTC models. All CTC models were trained using the EESEN implementation [Miao et al., 2015] and DNN/HMM systems were built using the Kaldi [Povey et al., 2011].

4.5.4 Updating Whole Network vs. Updating Output Layer

In the previous section, we showed that updating the whole network performs better than only updating the output layer and extending the output layer further improves the performance, as described in Figure 4.4. However, in the present experiment, we are interested in even smaller data sizes. We hypothesize that updating only the output layer might achieve better performance on more limited data. Therefore, we revisited the comparison between updating the whole network and updating only the output layer after extending the multilingual output layer to Portuguese and also did the comparison on less data (15 min, 30 min). The parameters connecting to unseen phonemes were randomly initialized. Since dropout has been proved to be effective in CTC-based cross-lingual adaptation, it was also applied in this work.

As shown in Figure 4.9, updating the whole network (EXT-ALL-RAND) consistently outperforms only updating the output layer (EXT-SM) even on 15-30 minutes adaptation data. It further confirms the previous observation. Therefore, all the parameters in the networks were

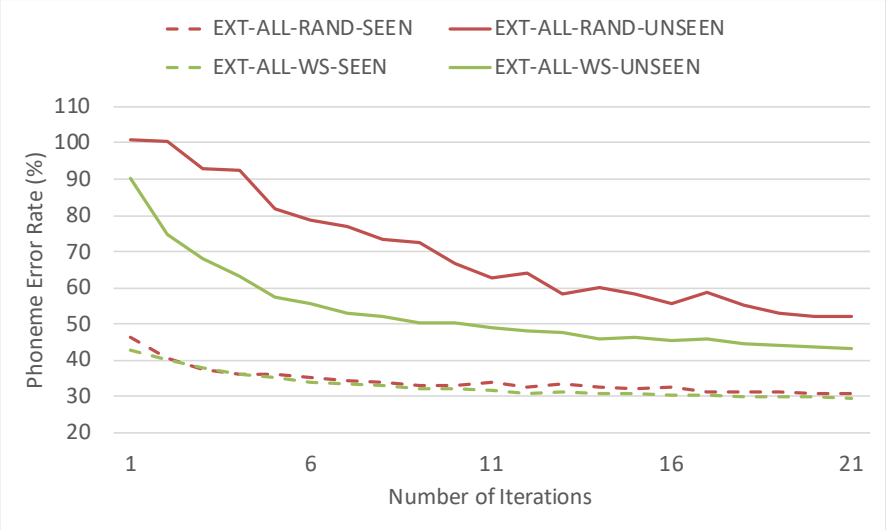


Figure 4.10: PERs (%) with respect to overlapped phonemes (SEEN) and new phonemes (UNSEEN) on PO development set. The adaptation was performed on 30 minutes data.

updated with dropout during cross-lingual adaptation in the remaining experiments.

4.5.5 Phonological Attribute Detector and Phoneme Classifier

The same multilingual data, EN, FR and GE was used to train the phonological attribute detector and the phoneme classifier. The phonological attribute detector is a 4 layer DNN, with 1024 hidden units in each layer. The same log-mel filterbank coefficients but with 5 frames context on each side were used as input features. The detector produces greater than 92.2% frame-level attribute detection accuracies for all phonological attributes used in this work and an overall 96.2% accuracy. Because of the limited space, we do not list the detection accuracy for all the attributes.

The input of the phoneme classifier is the concatenated log phonological posteriors with 5 frames context on each side. The DNN has 6 layers, each consisting of 1024 units. The output targets are multilingual IPA monophones based on EN, FR and GE, as described above. The test sets from the 3 languages were merged together to test the phoneme classification accuracy. The overall accuracy is 86.4%, which means it is a reliable phoneme predictor using phonological information.

4.5.6 Posterior-based Parameter Initialization

There are 19 phonemes in Portuguese that never appear in the experimental multilingual IPA phoneme set. Phonological attributes can be derived for each of the unseen phonemes based on prior knowledge. Phoneme posteriors were obtained by inputting the phonological attributes. Both parameter initialization approaches were tested.

Table 4.7: WERs (%) of cross-lingual adaptation with different initialization. WS denotes weighted summation of the multilingual weights in initialization and MAX represents taking the weights of the most probable mapped phonemes.

	15m	30m	1h	5h	10h	15h	21h
EXT-ALL-RAND	36.9	32.0	28.9	23.5	22.3	21.6	18.7
EXT-ALL-WS	33.7	29.6	27.7	23.5	22.0	21.2	18.5
EXT-ALL-MAX	34.3	29.7	27.9	23.5	22.2	21.4	18.9

Table 4.8: The most probable mappings of the 19 unseen Portuguese phonemes. The numbers in parentheses are the corresponding posteriors. Phonemes are represented in X-SAMPA.

a"	6 ~	6 ~ "	*d_j*	*e*"	*e* ~ "	i"
a(0.64)	6(0.96)	6(0.96)	*d*(0.98)	*e*(0.88)	*e* ~(0.95)	*i*(0.96)
i ~	*i* ~ "	*L*	*o*"	*o* ~ "	*r*	*t_j*
i(0.93)	*i*(0.93)	*j*(0.93)	*o*(0.88)	*o* ~(0.99)	*h*(0.66)	*t*(0.98)
u"	*u* ~	*u* ~ "	*l* =	*l* =~		
u(0.97)	*u*(0.96)	*u*(0.96)	*l*(0.95)	*n*(0.7)		

As shown in Table 4.7, both posterior-based initialization approaches achieve better performance with less than 3 hours adaptation data. The improvement becomes smaller and smaller with the increase of the adaptation data. As an example, we analyzed the phoneme error rate (PER) with respect to overlapped phonemes and new, unseen, phonemes separately on the development set during CTC training. As plotted in Figure 4.10. it shows that training from posterior-based initialization keeps the same performance on seen phonemes and yields much better PER on unseen phonemes. When adaptation data is limited, the model initialized using phonological information can quickly catch up on new phonemes.

The two initialization approaches perform almost the same. The phoneme posterior given by the phoneme classifier for each unseen phoneme is quite high, as listed in Table 4.8. This explains why there is little difference.

4.5.7 Compare CTC-based and DNN/HMM-based Adaptation

We also compared our proposed CTC-based adaptation with DNN/HMM-based adaptation approaches. In the DNN/HMM-based adaptation, the multilingual DNN trained on the same multilingual data was used as the seed model. We then replaced the multilingual output layer with Portuguese targets. The Portuguese context-dependent states and alignments were obtained from GMM/HMM systems trained on the corresponding amount of adaptation data. The adaptation was performed by either updating the whole network or only updating the output layer. Dropout was not applied for DNN since performance degradation was observed with dropout in our experiments.

It is clear from Figure 4.11 that the proposed CTC-based cross-lingual adaptations significantly outperform the DNN/HMM-based models on limited adaptation data (less than 3 hours).

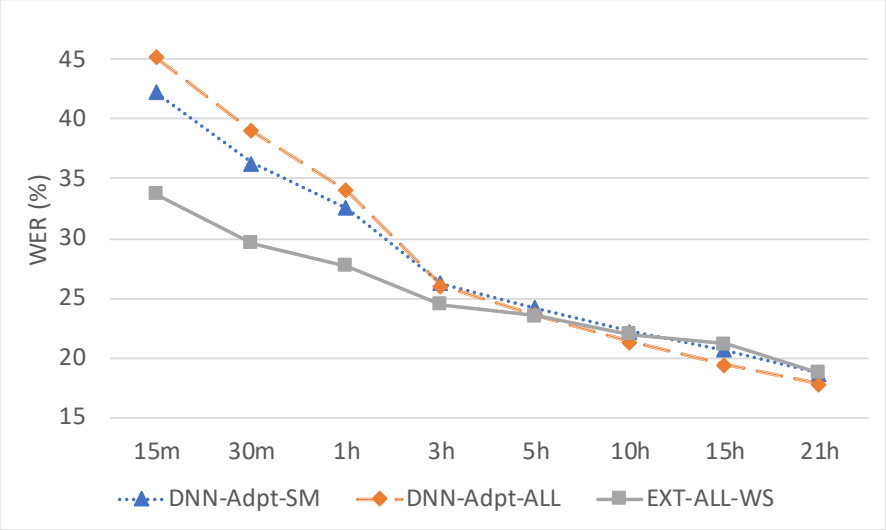


Figure 4.11: Comparison between CTC-based and DNN/HMM-based cross-lingual adaptation in WER(%). DNN-Adpt-SM denotes only updating the output layer of the DNN in the hybrid system. DNN-Adpt-ALL represents updating the whole network. EXT-ALL-WS is the CTC-based adaptation with the proposed parameter initialization.

CTC-based models retain all the information learned in multilingual training. By contrast, DNN/HMM-based adaptation only keeps the knowledge in hidden layers. This difference makes CTC-based models highly competitive when only limited data is available.

4.6 Conclusion

In this chapter, we first discussed CTC and LF-MMI for acoustic modeling and their applications in multilingual training. We systematically compared these training frameworks with MLE training from both theoretical and practical aspects. A monophone-based model trained with CTC or MMI loss was shown to achieve similar performance to the context-dependent model trained with CE. Sequence level training criteria that consider multiple hypotheses are more theoretically rigorous but also more sensitive to the amount of training data. Thus, models trained with CTC or LF-MMI benefit more from multilingual training when the amount of data for each language is limited. It was demonstrated that phoneme-based multilingual LF-MMI model outperforms both multilingual CTC models and state-of-the-art DNN/HMM systems.

Then, we take CTC training as an example to investigate phoneme-based cross-lingual adaptation. We have also shown that the universal phoneme-based multilingual CTC is extensible to new phonemes during cross-lingual adaptation. The extended model converges faster and better on overlapped phonemes and also catch up quickly on newly added phonemes. Combined with dropout during cross-lingual adaptation, the CTC-based model shows competitive performance with DNN/HMM-based adaptation on limited data. In addition, we develop a

novel parameters initialization approach by incorporating phonological information. When data is extremely limited, leveraging human knowledge and phonological information to initialize the model parameters can further improve the convergence of the model. The proposed initialization approach was shown to further improve the performance and yield much better performance than conventional DNN/HMM-based cross-lingual adaptation on limited data, potentially making the CTC model a competitive alternative in fast language adaptation of an ASR system.

Only CTC training is investigated in this chapter. Had time allowed, it would have made sense to apply the same approach on end-to-end LF-MMI training as well. We leave this as future research.

Part III

In the previous chapters, we showed that the most persuasive improvements to multilingual ASR come not from multilingual architectures per-se, but from Bayesian approaches that consider multiple hypotheses. CTC and MMI training takes into account all possible alignments in the optimization. MMI training additionally minimizes the probability of incorrect hypotheses, yielding better performance than the others. This suggests that the bottlenecks in the technology are more to do with regularization and handling of probability.

Building on this insight, in this final part we present two novel techniques that try to address these observations by better formalizing the way regularization and probability in general are handled in deep learning. In Chapter 5, we use Dropout at the test time to sample from the posterior predictive distribution of word-sequences to produce unbiased supervision for semi-supervised training to exploit unlabeled data. In Chapter 6, we revisit neural network activation functions based on Bayes's theorem. A Bayesian recurrent network is derived with probabilistic explanations, which leads to improvement of ASR performance in general so as to benefit multilingual ASR as well.

5 Semi-supervised Training Using Dropout

We have discussed cross-lingual adaptation based on phoneme-based end-to-end systems and shown that our proposed approach outperforms traditional DNN/HMM hybrid systems when adaptation data is limited. Although it is difficult and costly to obtain large amount of supervised data, abundant unsupervised audio is often easily available. Therefore, we present a novel approach for semi-supervised training in this chapter, which can address the data scarcity problem from a different aspect. The work in this chapter was published as Tong et al. [2019b].

5.1 Related Work and Motivation

As mentioned in previous chapters, the current acoustic models for ASR are based on DNNs. Sequence level training criteria such as CTC, LF-MMI and state-level Minimum Bayes Risk (sMBR) [Kaiser et al., 2000; Kanda et al., 2018] are preferred over frame-level objectives as they exploit sequential information and consider multiple hypotheses. However, these methods are known to be data hungry.

Although it is difficult and costly to obtain large amount of supervised data, abundant unsupervised audio is often easily available. A typical approach to exploit unsupervised data is to train a seed model using supervised data and use the seed model to automatically transcribe the unsupervised data [Zavaliagkos et al., 1998; Wessel and Ney, 2005]. Of course, the automatic transcripts are not perfect and the unsupervised training data is usually selected based on confidence measure on frame level [Veselỳ et al., 2013], word level [Wessel and Ney, 2005; Thomas et al., 2013; Veselỳ et al., 2017] or utterance level [Novotney et al., 2009; Grezl and Karafiát, 2013; Zhang et al., 2014].

More recently, lattice-based supervision has been combined with lattice-free MMI objective for semi-supervised training [Manohar et al., 2018]. Instead of using only the best path as supervision, training with the whole decoding lattice for the unsupervised data allows the model to learn from alternative hypotheses when the best path is not accurate. Although it has

shown significant improvement, directly learning from the whole lattice can deteriorate the performance in cases where the best path hypothesis has much lower WER than the alternate hypotheses. This is because as opposed to sampling the alternative training hypothesis from the posterior-predictive distribution over the word-sequences; the decoding lattice simply contains the most competitive hypothesis for the Maximum Likelihood estimate of the weights. This can bias the training towards incorrect hypotheses in the supervision lattice even if the best path is perfectly correct. It is thus very important to sample from the posterior predictive distribution over word sequences and to estimate the confidence or uncertainty of the ASR system for each hypothesis in the decoding lattice.

To this end, we propose to use a novel dropout-based approach to sample alternate hypothesis from the approximate posterior-predictive distribution instead of using decoding lattice which contains the most competitive hypothesis for the Maximum Likelihood estimate of the weights. Although this study was conducted based on LF-MMI training criterion, the same idea can be applied on other end-to-end training criteria as well. For example, our colleague, Dey et al. [2019] applied exactly the same idea for attention-based end-to-end training.

5.2 Model Uncertainty Using Dropout

5.2.1 Theoretical Background

Dropout-based training [Srivastava et al., 2014] of DNN acoustic models is a standard regularization technique often used to improve generalization properties (hence robustness) of state-of-the-art ASR systems. While dropout is typically used during training to prevent overfitting of DNNs, it was recently shown in the work of Gal and Ghahramani [2016] that dropout during inference can also provide a way to compute the model's uncertainty on its predictions. Gal and Ghahramani [2016] interpret dropout as a sampling approach which is equivalent to a variational approximation of a deep Gaussian process. A deep Gaussian process is a Bayesian machine learning model that could generate a probability distribution as the output. Applying standard dropout during inference allows us to estimate characteristics of this underlying distribution. The estimated variance of the distribution is taken to indicate the uncertainty of the model for a particular input. This method of estimating uncertainty is called Monte Carlo dropout [Labach et al., 2019].

In order to implement Monte Carlo dropout, a neural network is first trained with standard dropout. When performing inference on an test sample, the network is run N times while keeping dropout on. A different randomly generated dropout mask is used for the same input sample each time. Estimators for the mean and variance of the implicit Bayesian model output

are given by Gal [2016]:

$$\mathbb{E}[\mathbf{y}] \approx \frac{1}{N}\sum_{i=1}^{N}\hat{\mathbf{y}}_i(\boldsymbol{x}) \tag{5.1}$$

$$\mathbb{V}[\mathbf{y}] \approx \tau^{-1}\mathbf{I}_D + \frac{1}{N}\sum_{i=1}^{N}\hat{\mathbf{y}}_i(\boldsymbol{x})^{\intercal}\hat{\mathbf{y}}_i(\boldsymbol{x}) - \mathbb{E}[\mathbf{y}]^{\intercal}\mathbb{E}[\mathbf{y}] \tag{5.2}$$

where $\hat{\mathbf{y}}(\boldsymbol{x})$ is the output of the network given the inputs $\boldsymbol{x}$ and the i^{th} set of dropout masks and τ is a constant determined by the model structure. These are respectively taken to be the model output and an indication of the model uncertainty. We refer the readers to the work of Gal [2016] for more details.

Computing the *prediction uncertainty* of a DNN model using Monte Carlo sampling with dropout has been successfully used not only to characterize model errors but also to improve the system performance in various applications [Gal and Ghahramani, 2016; Gal, 2016; Kendall et al., 2015; Kendall and Cipolla, 2016].

5.2.2 Model Uncertainty in Acoustic Model

Inspired by the previous work, Vyas et al. [2019] have investigated the dropout uncertainty based approach in the context of automatic speech recognition (ASR) systems and explore its implications including (1) estimation of WERs without using oracle transcriptions and (2) error localization in the decoded ASR hypotheses. Unlike previous approaches, the proposed method does not require a lattice N-best list or a dedicated DNN to predict word-level confidences. We only exploit the uncertainty in the output of the acoustic models through the Monte Carlo sampling of the neural networks by using dropout at the test time.

For each utterance, we forward-pass it N times through a dropout enabled neural network acoustic model. Each of the N acoustic model outputs is then processed though the decoding pipeline to generate N dropout-hypotheses. Separately, we also obtain a hypothesis by keeping the dropout off during test time, as is done traditionally. The resulting $N+1$ hypotheses are then used to get an estimate of both the WER and the word-level confidences for the given utterance.

It has been shown that the variations in different decoded hypotheses with dropout are often highly localized at certain word positions and depict locations where the ASR decoding might be inaccurate. When the acoustic model is uncertain at a certain word, we observe variations in the predicted Monte Carlo hypotheses; we see the same hypothesis sampled when the model is confident.

5.3 Semi-supervised Training Using Dropout

In this section, we explain our approach to maximize the expected LF-MMI objective for unlabelled data by sampling target word-sequences from the posterior-predictive distribution for a given utterance. Our proposed loss for semi-supervised training is given as follows :

$$\mathcal{L}_{\text{MMI}} = \max_{\theta} \sum_{u=1}^{U} \log \left(\mathop{\mathbb{E}}_{\mathbf{W} \sim P(\mathbf{W}|\boldsymbol{X}^u, \mathbf{D}_s)} P(\mathbf{W}|\boldsymbol{X}^{(u)}, \theta) \right) \tag{5.3}$$

where $\boldsymbol{X}^{(u)}$ is the sequence of acoustic observations for utterance u, $\mathbf{D}_s$ is the supervised training data. $\mathbf{W}$ is the sampled target word sequence for the utterance. In this work, we use dropout to decode the same utterance multiple times to perform approximate Bayesian inference over the model parameters. This allows to sample from the approximate posterior-predictive distribution $P(\mathbf{W}|\boldsymbol{X}^{(u)}, \mathbf{D}_s)$. In contrast to this, the regular semi-supervised LF-MMI objective proposed in the work of Manohar et al. [2018] is given by:

$$\mathcal{L}_{\text{MMI}} = \max_{\theta} \sum_{u=1}^{U} \log \left(\sum_{\mathbf{W} \in \mathcal{G}_{\text{num}}^{(u)}} P(\mathbf{W}|\boldsymbol{X}^{(u)}, \theta) \right) \tag{5.4}$$

where $\mathcal{G}_{\text{num}}^{(u)}$ is the decoding lattice for the utterance u.

This can be seen an approximation to the proposed loss (5.3) where the expectation is taken over the word-sequences in the decoding lattice and each output word sequence in the lattice is assumed to be equally likely. Using the whole lattice as supervision provides additional information especially when the seed network is not confident on the unsupervised data. However, these alternative paths can also spoil the supervision in some cases. For instance, when the best decoding is quite accurate or when the utterance is short (containing only 1 or 2 words), the supervision might be biased towards the incorrect paths. One decoding lattice example from the unlabeled data is shown in Figure 5.2(a). The example utterance is quite clean. Although the model is quite confident on the sentence, the decoding lattice still contains many incorrect paths which will deteriorate the supervision quality.

Therefore, we propose to employ dropout at test time and decode the unlabeled data multiple times. Sampling from the posterior-predictive distribution will lead to an unbiased estimate to (5.3). We investigate Dropout-based sampling for both the acoustic and the language model.

5.3.1 Dropout-based Sampling

As mentioned before, Gal and Ghahramani [2016] recently showed that dropout during inference can lead to Bayesian inference over the model parameters and thus provide a way to sample from posterior-predictive distribution as well as to compute the model's uncertainty

on its predictions. This work is a novel attempt to study the usage and utility of dropout uncertainty in the context of semi-supervised training for ASR systems.

Dropout Sampling from Acoustic Model

Given an already trained DNN-based acoustic model, for each utterance, we forward-pass it N times through a dropout enabled neural network acoustic model. Each of the N acoustic model outputs is then processed though the decoding pipeline to generate N dropout-lattices. As shown in our previous work [Vyas et al., 2019], the acoustic model uncertainty about a test utterance is reflected in the variations observed in the predicted hypotheses for each Monte Carlo sample. Moreover, the variations in different decoded hypotheses for any utterance are often highly localized at certain word positions and depict locations where the ASR decoding might be inaccurate.

Therefore, we can generate an unbiased supervision lattice for each unlabeled utterance by composing the predicted hypotheses from the Monte Carlo samples. More specifically, as shown in Figure 5.1, for each unlabeled utterance, we prune the dropout-lattices with a very small beam and combine them together to create the supervision lattice for semi-supervised training. Optimizing $P(\mathbf{W}|\boldsymbol{X}^{(u)},\theta)$ over this lattice leads to an unbiased estimate of (5.3). We keep the rest of the training steps the same as proposed in the work of Manohar et al. [2018].

Figure 5.2(b) shows the lattice for the same example utterance, generated using the proposed approach. As we can find, most of the paths in this lattice correspond to the correct transcription since the model is confident on this clearly spoken utterance (high $P(\mathbf{W}|\boldsymbol{X}^{(u)},\theta_s)$ for the decoded sequence). If the model is uncertain about an utterance, more variations will appear in each decoding sample [Vyas et al., 2019] so that the combined lattice can still retain alternative paths to provide additional information. We hypothesize that the unbiased lattice combined from different dropout-based decoding samples better reflects the uncertainty of the acoustic model and is able to foster the more likely word sequences, while keeping variations for uncertain utterances, thus improving the semi-supervised training performance.

Dropout Sampling from Language Model

It is not straight forward to apply the dropout-based sampling in N-gram language model (LM) that is used in decoding. Instead, we investigated the same framework for neural network-based language models during re-scoring. For each unlabeled utterance, we first obtain the decoding lattice using the acoustic model with dropout off. The lattice is then re-scored N times by using a dropout enabled neural network language model. These re-scored lattices are then pruned and combined together to generate the supervision lattice which reflects uncertainties in the language model. Similarly, we keep everything else the same in the semi-supervised training setup and evaluate the performance. Additionally, we hypothesize that the combination of the dropout sampling from acoustic model and the sampling from language

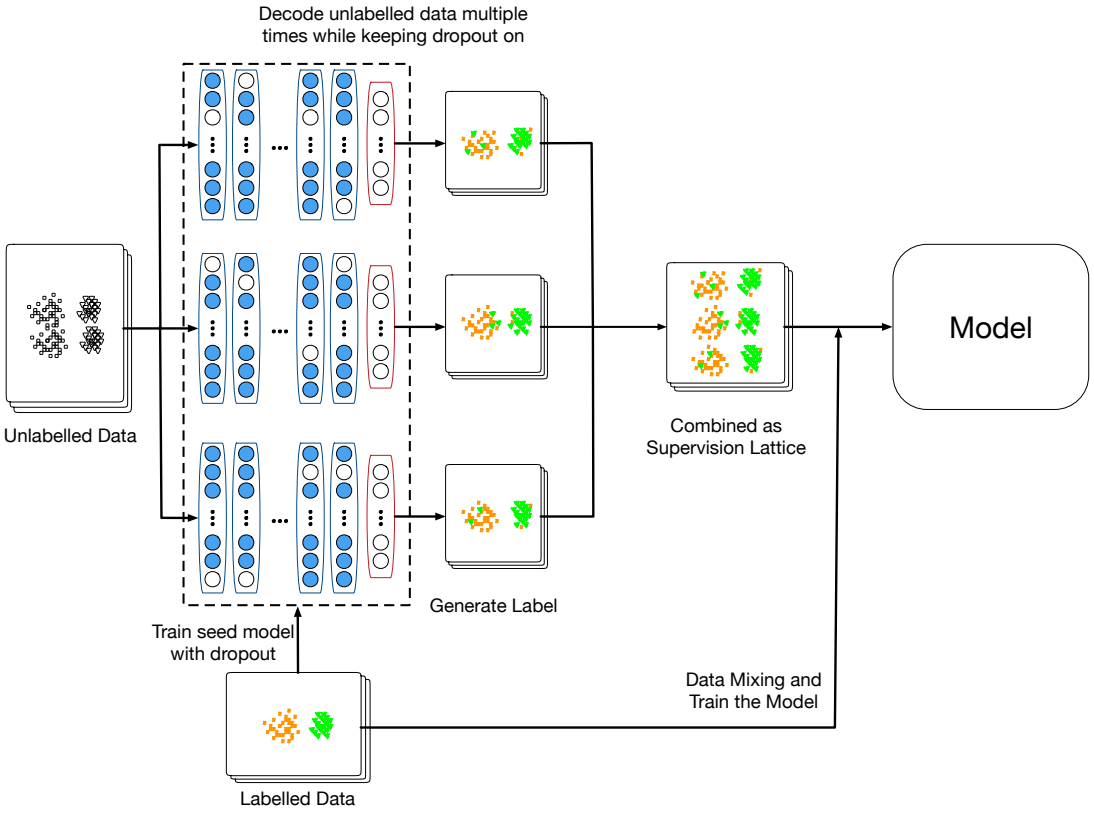
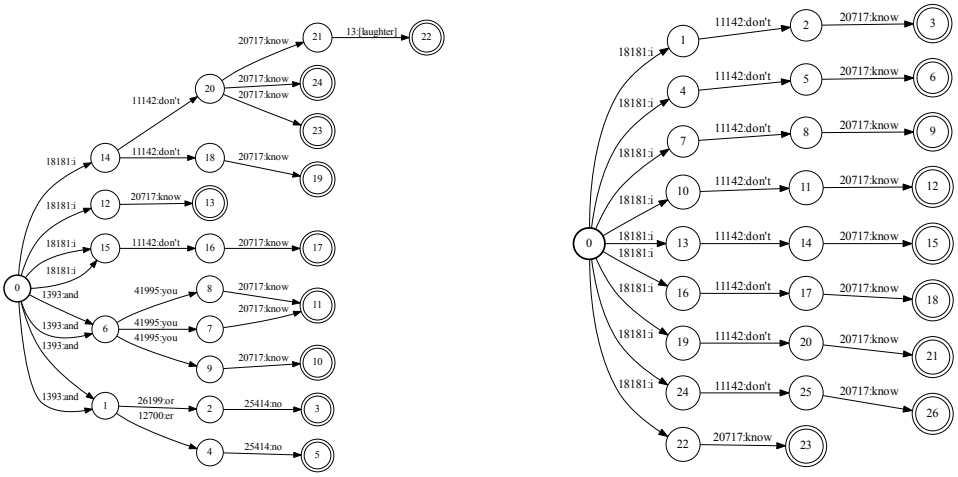


Figure 5.1: Flow-chart of the proposed method. Each network in the figure represents one network sample because of a different random selection of the active nodes. The white nodes denote that they are dropped out.

model could help further because it covers the uncertainties from each of the two major components of an ASR system. This combination will also be investigated in Section 5.5.4.

5.3.2 Discussion

The proposed approach has similarities to Negative Conditional Entropy (NCE) [Manohar et al., 2015] for semi-supervised training where the authors minimize the expected risk over the uncertain decoding of the unsupervised data. However, in contrast to the work of Manohar et al. [2015], where the decoding lattice with forward-backward likelihood computation estimates the likelihood of word-sequence, in this work, we directly sample from the approximate posterior-predictive distribution using dropout to generate the supervision lattice. The approach proposed in the work of Li et al. [2017] also shares some similarities in the sense that the labels of unlabeled data are the decoding output from multiple seed models to incorporate the diversity. An ensemble of models is trained in parallel using these diverse labels, and then averaged as the final model. In the context of our framework, these multiple seed models can be considered as the dropout-based neural network samples and all the diverse labels are



(a) Standard decoding lattice.

(b) The unbiased lattice.

Figure 5.2: Lattices of a clearly spoken utterance. (a) represents the pruned decoding lattice from a dropout-off acoustic model. (b) denotes the unbiased lattice combined from multiple dropout decoding samples.

combined into one supervision lattice used for LF-MMI training. Thus, the proposed method is simpler and more rigorous.

5.4 Experimental Setup

Similar to the work of Manohar et al. [2018], we report our results on the Fisher English corpus [Cieri et al., 2004]. A randomly chosen subset of speakers (250 hours) from the corpus is used as unsupervised data. The transcripts from the remaining 1250 hours are used to train the language models for decoding and re-scoring the unsupervised data. We use a 50 hours subset from the corpus as the supervised data to train the seed model. The supervised data is then combined with the unsupervised data to train the final models. The results are reported on separately held-out development and test sets (about 5 hours each), which are part of the standard Kaldi [Povey et al., 2011] recipe for Fisher English. WER Recovery Rate (WRR) [Ma and Schwartz, 2008] is used as an additional metric to evaluate the WER improvements from semi-supervised training:

$$\text{WRR} = \frac{\text{BaselineWER} - \text{SemisupWER}}{\text{BaselineWER} - \text{OracleWER}}.$$

Following the standard Kaldi recipe, we first train a GMM system using only the supervised data and use this to get supervision to train a seed LF-MMI time-delay neural network (TDNN). The TDNN consists of 8 hidden layers, with 450 hidden units in each layer. Dropout is applied on top of each layer. We use i-vector [Dehak et al., 2011] for speaker adaptation of the neural

network. The i-vector extractor is trained using the combined supervised and unsupervised datasets. Also, for comparison purposes, we use statistics from only the supervised data to train the context-dependency decision tree. Following Manohar et al. [2018], the phone LM used for creating the denominator FST is estimated using phone sequences from both supervised and unsupervised data with a higher weight to the phone sequences from supervised data (1.5 for the 50 hours supervised dataset and 1 for the unsupervised data).

In addition to N-gram language models, A neural network-based language model is trained on the same data. The network consists of 3 temporal convolutional layers [Bai et al., 2018], with 600 units in each layer. The size of the word embeddings is fixed to 600 and the kernel size is taken to be 3. Similarly, dropout is applied on top of each layer. The language model was trained using Pytorch.

5.5 Results

5.5.1 Effect of Dropout Sample Numbers from Acoustic Model

As a hyper-parameter, N denotes the number of dropout samples needed to represent the posterior-predictive distribution. Although more posterior samples can better represent the distribution, it is more time consuming. Therefore, it is of importance to investigate appropriate value of N for a good trade-off. Here, we have only applied the dropout-based sampling on the acoustic model. To generate the supervision lattice of the unsupervised data, we decoded the data N time while keeping dropout on and varied N from 5 to 40. As a baseline, we use the decoding lattice generated from the same acoustic model in a standard way (with dropout off), following Manohar et al. [2018]. The decoding lattices were not re-scored and the performance was evaluated on development set. As shown in Figure 5.3, the performance of the proposed method first gets improved as we combined more decoding lattices. It seems to saturate after reaching 20 times of decoding. Therefore, we keep using $N = 20$ for the following experiments except when explicitly stated.

5.5.2 Quality Analysis of the Supervision Lattices

From Figure5.3, we can also see that the unbiased lattices yield better WER than the regular semi-supervised training approach. We analyzed the averaged WER and the sentence error rate (SER) of the unbiased lattices with $N = 20$ and compared it with the regular decoding lattice on the whole unsupervised data. We evaluated the WER of each lattice by averaging the WER of the N-best hypotheses for each utterance. The regular decoding lattice was generated from the dropout-off model and was pruned before this evaluation.

Table 5.1 shows that the unbiased lattice has a better WER and a much better SER than the regular lattice. The better WER and SER confirms our hypothesis that the lattice combination from different dropout samples can help reduce the effect of incorrect hypotheses in the

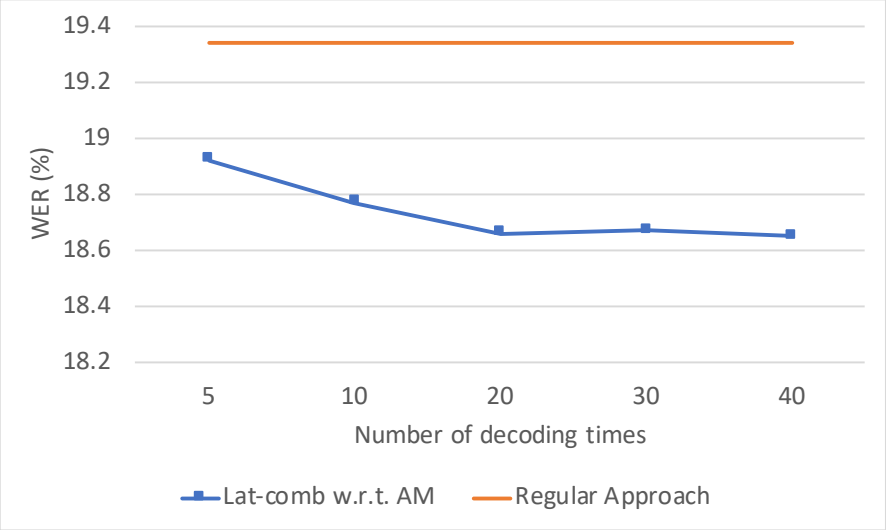


Figure 5.3: WER (%) of different semi-supervised training setup by varying the value of N. The dropout-based sampling is only applied on the acoustic model. The red line denotes the regular semi-supervised training approach [Manohar et al., 2018].

Table 5.1: Comparison the averaged WER(%) and SER (%) between combined lattice and regular decoding lattice.

	avg. WER	SER
Regular Lat	23.6	87.8
Lat-comb	23.1	75.7

supervision lattice when the acoustic model is confident on the unlabeled sentence, while keeping alternative paths to be exploited when the acoustic model is uncertain. It also explains the improvement on development set after semi-supervised training because the unbiased lattice provides supervision with better quality.

5.5.3 Effect of Number of Dropout Samples from Language Model

Similar to Section 5.5.1, in this section, we analyze the effect of N with respect to language model only. To generate the unbiased supervision with respect to language model, we first obtained the lattice by decoding the data in regular way (keeping dropout off). The lattice was then re-scored N times by the network-based language model while keeping dropout on. Similarly, we varied N from 5 to 40.

As shown in Figure 5.4, the performance on the development set does not change much with different values of N and the proposed approach yields very slight improvement. One of our previous hypotheses was that the Dropout-based Monte Carlo sampling can help reduce the confusion in the supervision lattice especially for shorter sentences. However, language model re-scoring for sentences with one or two words wouldn't make much difference by its nature.

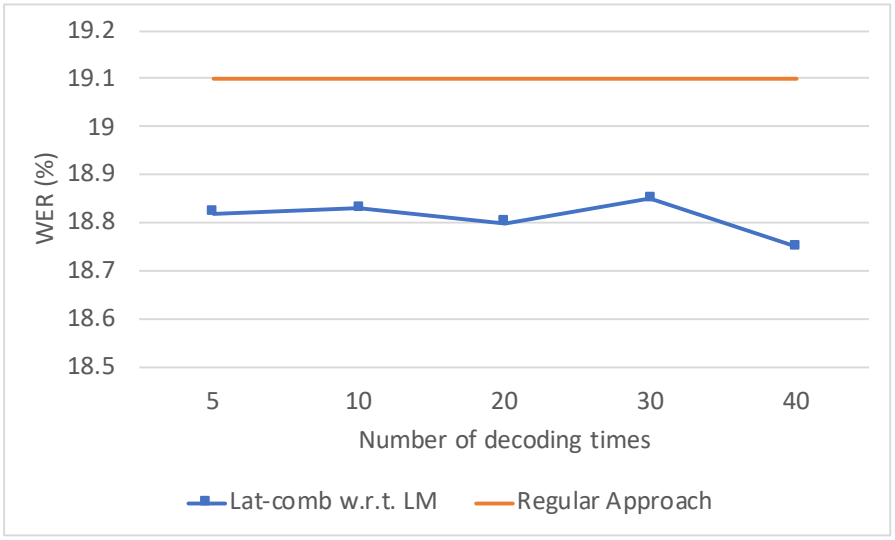


Figure 5.4: WER (%) of different semi-supervised training setup by varying the value of N. The dropout-based sampling is only applied on the language model. The red line denotes the regular semi-supervised training approach Manohar et al. [2018] where the supervision lattices of unsupervised data were also re-scored using NN LM.

We found there are around one third of the unsupervised utterances containing only 3 words or less. Therefore, applying dropout sampling on language model only slightly improves the performance.

5.5.4 Complete Comparison

Table 5.2 shows a complete comparison of the alternatives we are exploring. The first row shows the performance of supervised training using only 50 hours supervised data. The last row shows supervised training results using oracle transcripts for the unsupervised data. All the supervision lattices for unlabeled data were re-scored using the network language model. For re-scoring the unbiased acoustic lattice in the proposed framework, we first generated the decoding lattice samples by keeping dropout on in the acoustic model. Then, each decoding lattice was re-scored before pruning and combination. In order to testify whether the dropout sampling from both acoustic model and language model can further improve the performance, we simply combined the lattice evaluated in Section 5.5.1 and Section 5.5.3 and tested the WER after semi-supervised training.

As we can see in the Table 5.2, semi-supervised training approach as proposed in the work of Manohar et al. [2018] yields around 8.6% relative WER reduction. Incorporated with uncertainty information from only the acoustic model, the unbiased supervision lattice improves over the supervised system by around 12.2%. Dropout sampling from network language model also brings improvement, although the improvement is not as much as the one from acoustic model. The combination cannot further improve the performance significantly. Most of the

Table 5.2: Comparison between combined lattice and regular decoding lattice in WER(%). The 50h supervised system is used as baseline to calculate WRR.

System	Dev	Test	WRR
50h supervised	21.0	20.9	-
Regular Approach	19.1	19.2	53.7 %
Lat-comb w.r.t. AM	18.5	18.3	76.1%
Lat-comb w.r.t. LM	18.8	18.7	65.7%
Lat-comb w.r.t. AM+LM	**18.5**	**18.2**	77.6%
Oracle	17.7	17.5	

gains come from the acoustic part. In total, the proposed semi-supervised training approach yields approximately 12.4% relative improvement over the supervised setup. Compared with the regular LF-MMI semi-supervised training, the proposed approach gives 4.2% relative WER reduction and 51.6% WER recovery rate.

5.6 Conclusion

We have proposed a novel way to exploit dropout uncertainty in context of semi-supervised LF-MMI training. It was demonstrated that the unbiased lattice combined from different dropout-based decoding samples is able to help reduce the confusion of the lattice paths, while keeping variations for uncertain unlabeled utterances. Experiments on the Fisher English shows that the proposed approach can further improve the WER over the regular semi-supervised training framework. While this chapter primarily focused on LF-MMI training, it is clear that the idea can be further extended to other frameworks such as end-to-end based semi-supervised training [Dey et al., 2019]. Had the time allowed, it would have made sense to investigate the proposed semi-supervised training approach in cross-lingual adaptation scenarios. As a general semi-supervised training framework, it can help mitigate the data scarcity problem for low-resourced languages. We leave this as future research.

6 Bayesian Recurrent Unit

So far we have shown various multilingual training and cross-lingual adaptation techniques. Inevitably, neural networks, especially RNNs, serve as the key components. Furthermore, our previous research implies that Bayesian approach is beneficial. This motivates us to dive more deeply into the theoretical basis of neural networks, with a particular focus on RNNs, to provide a thorough analysis of the current field.

We begin by reiterating that common neural network activation functions have simple Bayesian origins. In this spirit, we go on to show that Bayes's theorem also implies a simple recurrence relation; this leads to a Bayesian recurrent unit (BRU) with a prescribed feedback formulation. We show that introduction of a context indicator leads to a variable feedback that is similar to the forget mechanism in conventional recurrent units. A similar approach leads to a probabilistic input gate. The Bayesian formulation leads naturally to the two pass algorithm of the Kalman smoother or forward-backward algorithm, meaning that inference naturally depends upon future inputs as well as past ones. Experiments on speech recognition confirm that the resulting architecture can perform as well as a bidirectional recurrent network with the same number of parameters as a unidirectional one. Further, when configured explicitly bidirectionally, the architecture can exceed the performance of a conventional bidirectional recurrence. The text[1] of this chapter was a collaborative work with my supervisor and was published as Garner and Tong [2020].

6.1 Related work

In signal processing and statistical pattern recognition, recurrent models have been ubiquitous for some time. They are perhaps exemplified by two cases: the state space filter of Kalman [Kalman, 1960; Scharf, 1991] is appropriate for continuous states; the HMM [Baum and Petrie, 1966; Bahl et al., 1983] for discrete states. Both of these approaches can be characterised as being statistically rigorous; each has a forward-backward training procedure that arises from a statistical estimation formulation.

[1] ©2020 IEEE

Recurrence is also important in modern deep learning. The foundations were laid shortly after the introduction of the MLP [Rumelhart and McClelland, 1986; Rumelhart et al., 1986] with the back-propagation through time algorithm [Rumelhart and McClelland, 1986; Williams and Zipser, 1989]. Such architectures can be difficult to train; some of the difficulties were addressed by the LSTM of Hochreiter and Schmidhuber [1997]. The LSTM was subsequently modified by Gers et al. [2000] to include a forget gate, and Gers et al. [2002] to include peephole connections. The full LSTM is illustrated in Figure 6.1.

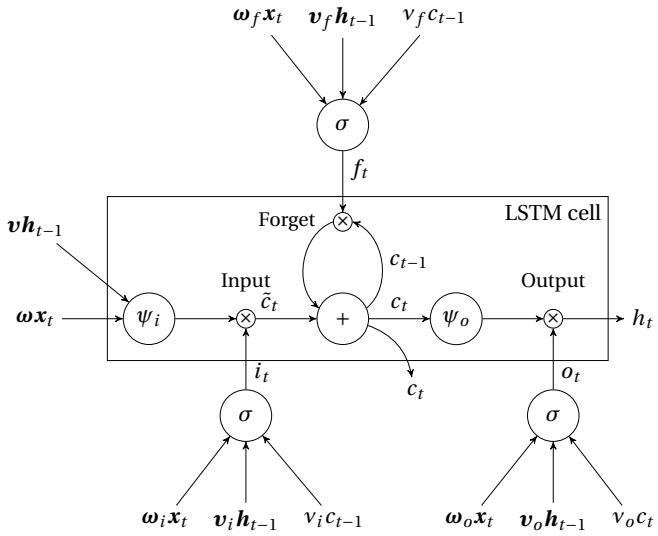


Figure 6.1: The long short term memory of Hochreiter and Schmidhuber [1997]. Non-linearities ψ are taken to be tanh.

The LSTM's concept of gates has since been used in the GRU of Cho et al. [2014], and remains important. In GRU, the input and forget gates are combined into a single operation, and the output gate is applied to the recurrent part of the input instead. It is illustrated in Figure 6.2. The GRU has also been modified: In a minimally gated unit (MGU), Zhou et al. [2016]

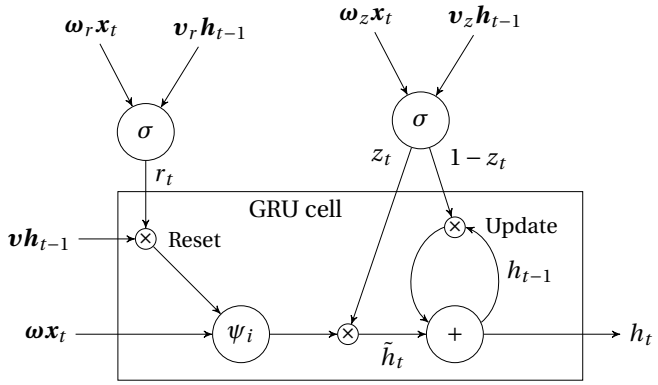


Figure 6.2: The gated recurrent unit of Cho et al. [2014]. As in the LSTM, the non-linearity ψ is usually tanh.

replace the reset gate with a signal from the update gate; in the notation here, r_t is replaced by $1 - z_t$. Ravanelli et al. [2017, 2018] remove the reset gate altogether in their light GRU (Li-GRU), equivalent to setting $r_t = 1$.

Notice that the LSTM and GRU implicitly define three types of recurrence:

1. A *unit-wise* recurrence, exemplified by the constant error carousel (CEC, forget loop) of the LSTM or the GRU update loop.

2. A *layer-wise* recurrence, being the vector loop $\boldsymbol{h}_{t-1}$ from output to input.

3. A *gate* recurrence, being the vector loop from output to gate.

Several authors have noted the similarities between HMMs and (recurrent) networks. Bourlard and Wellekens [1989] show that the two architectures can be made to compute similar probabilistic values. Bridle [1990b] shows that a suitably designed network can mimic the *alpha* part of the forward-backward algorithm. Bridle also points out similarities between the back-propagation (of derivatives) in the training of MLPs and the backward pass in HMMs. With the bidirectional recurrent neural network (BiRNN), in contrast to *seeking* relationships, Schuster and Paliwal [1997] *imposed* the backward relationship between HMMs and MLPs by means of a second recurrence relationship running in the opposite direction. This was in fact to explicitly allow the network to take account of "future" observations. The natural substitution of LSTMs for the same purpose was described by Graves and Schmidhuber [2005] resulting in the bidirectional LSTM (BLSTM or BiLSTM); this type of network remains the state of the art in several fields.

Putting aside the concept of recurrence, probabilistic interpretations of feed-forward MLPs are well known. Although the sigmoid is usually described as being a smooth (hence differentiable) approximation of a step function, its probabilistic origin was pointed out by Bridle [1990a], and is well known to physicists via the Boltzmann distribution. It has also been shown that the training process yields parameters that make sense in a statistical sense; this is evident from the work of Richard and Lippman [1991], summarising work such as that of Bourlard and Wellekens [1989], and most thoroughly by MacKay [MacKay, 1992a,b,c] in papers that constituted his PhD thesis, later popularised by Bishop [Bishop, 1995].

In this chapter, we build on this latter body of work, recalling that several MLP concepts have sound Bayesian origins. We show that this implies a natural probabilistic recurrence, leading to an architecture similar to the GRU [Cho et al., 2014]. We go on to show that, because the derivation is probabilistic, a backward recursion is also evident; this without the explicit extra backward recurrence of the BiRNN architectures described above.

6.2 Background

6.2.1 Bayesian Interpretation of MLP Units

We begin by making explicit a relationship, pointed out by Bridle [1990a], between Bayes's theorem and the sigmoid activation; we show that the same relationship also applies to ReLU (rectifying linear unit) activations.

Say we have an observation vector, $\boldsymbol{x}$, and we want the probability that it belongs to class i, where $i \in \{1, 2, \ldots, C\}$. The Bayesian solution is

$$P(c_i \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid c_i) P(c_i)}{\sum_{j=1}^{C} p\left(\boldsymbol{x} \mid c_j\right) P\left(c_j\right)}, \tag{6.1}$$

where c_i refers to the event that the class takes value i, and $\boldsymbol{x}$ refers to the event that the observation random variable takes value $\boldsymbol{x}$.

If we take the observations to be from multivariate Gaussian distributions then, in the two class case, $C = 2$,

$$P(c_1 \mid \boldsymbol{x}) = \frac{1}{1 + \exp\left(-(\boldsymbol{\omega}^\mathsf{T} \boldsymbol{x} + v)\right)}, \tag{6.2}$$

where

$$\begin{aligned} \boldsymbol{\omega}^\mathsf{T} &= (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\mathsf{T} \boldsymbol{\Sigma}^{-1} \\ b &= \log P(c_1) - \log P(c_2) - \frac{1}{2}\left(\boldsymbol{\mu}_1^\mathsf{T} \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_1 - \boldsymbol{\mu}_2^\mathsf{T} \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_2\right), \end{aligned} \tag{6.3}$$

and $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}$ are respectively the mean and covariance of the constituent Gaussians. The class priors in this case, $P(c_i)$, are taken to be constant and subsumed in the bias term. This is the commonly used sigmoid activation.

In the multi-class case, $C \geq 2$,

$$P(c_i \mid \boldsymbol{x}) = \frac{\exp\left(\boldsymbol{\omega}_i^\mathsf{T} \boldsymbol{x} + v_i\right)}{\sum_{j=1}^{C} \exp\left(\boldsymbol{\omega}_j^\mathsf{T} \boldsymbol{x} + v_j\right)}, \tag{6.4}$$

where

$$\boldsymbol{\omega}_i^\mathsf{T} = \boldsymbol{\mu}_i^\mathsf{T} \boldsymbol{\Sigma}^{-1} \tag{6.5}$$

$$b_i = \log P(c_i) - \frac{1}{2} \boldsymbol{\mu}_i^\mathsf{T} \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_i. \tag{6.6}$$

This is the softmax activation function introduced in the work of Bridle [1990a].

A Gaussian assumption is appropriate for MLP inputs. However, hidden layers take inputs

from previous layers with sigmoid outputs; their values are closer to beta distributions. If, instead of a Gaussian, the observations are assumed to follow independent beta distributions,

$$p(x) = \frac{1}{B(\alpha,\beta)} x^{\alpha-1}(1-x)^{\beta-1} \tag{6.7}$$

$$= \frac{1}{B(\alpha,\beta)} e^{(\alpha-1)\log(x)} e^{(\beta-1)\log(1-x)}, \tag{6.8}$$

where the second line emphasises that the beta is exponential family. With $\beta = 1$, we then have:

$$P(c_1 \mid \boldsymbol{x}) = \frac{1}{1+\exp\left(-(\boldsymbol{\omega}^\mathsf{T}\log(\boldsymbol{x})+v)\right)}, \tag{6.9}$$

with

$$\boldsymbol{\alpha}_j = (\alpha_{j,1},\ldots,\alpha_{j,P})^\mathsf{T}, \tag{6.10}$$

$$\boldsymbol{\omega} = \boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_2 \tag{6.11}$$

$$b = \log P(c_1) - \log P(c_2) - \sum_{i=1}^{P}\left[\log B(\alpha_{1,i},1) - \log B(\alpha_{2,i},1)\right] \tag{6.12}$$

and P is the input dimension.

So, when a sigmoid output is used as the input to a subsequent layer, the value that makes sense under a beta assumption is its logarithm. Taking a logarithm of a sigmoid results in the softplus described by Dugas et al. [2001] albeit for a different reason. Glorot et al. [2011] show that the ReLU is a linear approximation to the softplus.

6.3 General Probabilistic Recurrence

In the previous section, we showed that the main activations used in MLPs have probabilistic explanations. In this spirit, we derive a recursive activation from a probabilistic point of view. At the outset, we expect the formulation to dictate the form of the recursion, removing otherwise ad-hoc aspects of standard techniques.

6.3.1 Conditional Independence of Observations

Let us assume that we have a (temporal) sequence of observations $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T$. (6.1) becomes (abbreviated for the moment)

$$P(c_i \mid \boldsymbol{x}_T, \boldsymbol{x}_{T-1}, \ldots, \boldsymbol{x}_1) \propto p(\boldsymbol{x}_T \mid c_i, \boldsymbol{x}_{T-1}, \ldots, \boldsymbol{x}_1)\, P(c_i \mid \boldsymbol{x}_{T-1}, \ldots, \boldsymbol{x}_1). \tag{6.13}$$

If we then assume that all the $\boldsymbol{x}_t$ are conditionally independent given c_i, we have

$$P(c_i \mid \boldsymbol{x}_T, \boldsymbol{x}_{T-1}, \ldots, \boldsymbol{x}_1) \propto p(\boldsymbol{x}_T \mid c_i) P(c_i \mid \boldsymbol{x}_{T-1}, \ldots, \boldsymbol{x}_1). \tag{6.14}$$

This is a standard recursion where the posterior at time $t-1$ forms the prior for time t.

6.3.2 Application to MLP

More generally, say we have a matrix, $\boldsymbol{X}_T$, the rows of which are observation vectors $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T$. There is a corresponding matrix, $\boldsymbol{H}_T$, the rows of which are vectors $\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_T$. We assume each element $h_{t,i}$ of $\boldsymbol{H}$ represents a probability $P(\phi_i \mid \boldsymbol{X}_t)$ of the event that feature i exists in the observation sequence up to time t. Conversely, $1 - h_{t,i} = P(\bar{\phi}_i \mid \boldsymbol{X}_t)$. Notice that, at this stage, ϕ_i is not time dependent; the feature exists (or not) for the whole sequence, with each observation in the sequence updating $P(\phi_i \mid \boldsymbol{X}_t)$. Now say that the probabilities $P(\phi_i \mid \boldsymbol{X})$ are independent given some parameters, $\boldsymbol{\theta}$. So the joint probability is the product

$$P(\phi_1, \phi_2, \ldots, \phi_F \mid \boldsymbol{\theta}, \boldsymbol{X}_t) = P(\phi_1 \mid \boldsymbol{\theta}, \boldsymbol{X}_t) P(\phi_2 \mid \boldsymbol{\theta}, \boldsymbol{X}_t) \ldots P(\phi_F \mid \boldsymbol{\theta}, \boldsymbol{X}_t). \tag{6.15}$$

For a given feature, ϕ_i,

$$\begin{aligned} h_{t,i} &= P(\phi_i \mid \boldsymbol{\theta}, \boldsymbol{X}_t) && (6.16)\\ &= \frac{p(\boldsymbol{x}_t \mid \phi_i, \boldsymbol{\theta}, \boldsymbol{X}_{t-1}) P(\phi_i \mid \boldsymbol{\theta}, \boldsymbol{X}_{t-1})}{\sum_{\phi_i} p(\boldsymbol{x}_t \mid \phi_i, \boldsymbol{\theta}, \boldsymbol{X}_{t-1}) P(\phi_i \mid \boldsymbol{\theta}, \boldsymbol{X}_{t-1})} && (6.17)\\ &= \frac{1}{1 + \dfrac{p(\boldsymbol{x}_t \mid \bar{\phi}_i)}{p(\boldsymbol{x}_t \mid \phi_i)} \cdot \dfrac{P(\bar{\phi}_i \mid \boldsymbol{X}_{t-1})}{P(\phi_i \mid \boldsymbol{X}_{t-1})}}, && (6.18) \end{aligned}$$

where, in the final line and hereafter, we drop the conditioning on $\boldsymbol{\theta}$ for clarity. The final expression contains two fractional terms. The first of these follows from the conditional independence assumption above, and leads to the sigmoid of (6.2) and (6.9), but without the priors in the bias terms. Instead of being static, the priors form the second fractional term which is a multiplicative feedback

$$\frac{P(\bar{\phi}_i \mid \boldsymbol{X}_{t-1})}{P(\phi_i \mid \boldsymbol{X}_{t-1})} = \frac{1 - h_{t-1,i}}{h_{t-1,i}} = \frac{1}{\text{odds}(h_{t-1,i})} \tag{6.19}$$

If this were indeed included as an additive component of the bias in (6.2) or (6.9) then the fed back term would be

$$\begin{aligned} \log\left(\frac{h_{t-1,i}}{1 - h_{t-1,i}}\right) &= \text{logit}(h_{t-1,i}) && (6.20)\\ &= \log(h_{t-1,i}) - \log(1 - h_{t-1,i}). && (6.21) \end{aligned}$$

6.4 Probabilistic Forget

The Bayesian Recurrent Unit (BRU) described above carries the assumption that a feature is present (or not) in the entire input sequence. By contrast, we know from the LSTM that it is necessary to allow an activation to respond differently to different inputs depending on the context. In an LSTM this is achieved using gates. We show here that gates can be derived probabilistically.

Say that $P\left(\phi_i\right)$ is somehow dependent upon another variable indicative of context. For instance, if ϕ_i is indicative of a characteristic of a sentence, it is dependent upon the previous words in the sentence, but resets after a (grammatical) period, when the sentence changes. Say there is a binary state variable, ζ, where $\zeta = 1$ indicates the context remaining relevant, and $\zeta = 0$ indicates that it is not relevant. We can assign a probability, $z_t = P\left(\zeta_t = 1 \mid \boldsymbol{X}_t\right)$ and the inverse $(1 - z_t) = P\left(\zeta_t = 0 \mid \boldsymbol{X}_t\right)$, where z_t is predicted by the network. It is then the prior (in (6.19)) that depends on the context. $\boldsymbol{\phi}$ is now dependent upon the time index, t.

Note that the state variable can be defined for one or multiple features. In the following derivation, we assume only one feature, removing the need for an index. However, it is common for recurrence to use one variable per feature.

6.4.1 Unit-wise Recursion

We first consider the case where the ϕ_i are taken to be independent; it is derived below,

$$P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right) = \sum_{\phi_{t-1,i}} \sum_{\zeta_{t-1}} P\left(\phi_{t,i} \mid \phi_{t-1,i}, \zeta_{t-1}, \boldsymbol{X}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) P\left(\zeta_{t-1} \mid \boldsymbol{X}_{t-1}\right) \tag{6.22}$$

$$\begin{aligned} &= P\left(\phi_{t,i} \mid \phi_{t-1,i}, \zeta_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) P\left(\zeta_{t-1} \mid \boldsymbol{X}_{t-1}\right) \\ &\quad + P\left(\phi_{t,i} \mid \bar{\phi}_{t-1,i}, \zeta_{t-1}\right) P\left(\bar{\phi}_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) P\left(\zeta_{t-1} \mid \boldsymbol{X}_{t-1}\right) \\ &\quad + P\left(\phi_{t,i} \mid \phi_{t-1,i}, \bar{\zeta}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) P\left(\bar{\zeta}_{t-1} \mid \boldsymbol{X}_{t-1}\right) \\ &\quad + P\left(\phi_{t,i} \mid \bar{\phi}_{t-1,i}, \bar{\zeta}_{t-1}\right) P\left(\bar{\phi}_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) P\left(\bar{\zeta}_{t-1} \mid \boldsymbol{X}_{t-1}\right) \end{aligned} \tag{6.23}$$

$$\begin{aligned} &= 1 \times h_{t-1,i} z_{t-1} + 0 \times (1 - h_{t-1,i}) z_{t-1} + p_i h_{t-1,i} (1 - z_{t-1}) \\ &\quad + p_i (1 - h_{t-1,i})(1 - z_{t-1}) \end{aligned} \tag{6.24}$$

$$= (1 - z_{t-1}) p_i + z_{t-1} h_{t-1,i}, \tag{6.25}$$

where p_i is the unconditional prior probability of feature i being present. Notice that the simplifications arise from the interaction of $\phi_{t,i}$, $\phi_{t-1,i}$ and ζ_{t-1}: context remaining relevant implies the feature should remain. So, for instance, the feature changing from not present to present when context is relevant has zero probability.

In a Kalman filter sense, $P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right)$ is the predictor. The result is an intuitive linear combination of the previous output with a prior. Here, although we deal with a discrete state variable, we use the Kalman filter analogy because it is easier to follow. Nevertheless, a correspondence

with *alpha*, *beta* and *gamma* probabilities will be evident to readers familiar with Markov models.

There is a question of initialisation. The first output corresponding to $t = 1$ should use the value $h_{0,i} = p_i$; thereafter, the value from the feedback loop can be taken.

$$\begin{array}{lll} \text{At time } t = 1, & z_{t-1} = 0, & h_{t-1,i} = p_i \\ \text{At time } t = 2, & z_{t-1} = f_z(\boldsymbol{X}_{t-1}), & h_{t-1,i} = f_h(\boldsymbol{X}_{t-1}) \end{array}$$

where $f_{\cdot}(\cdot)$ is taken to mean "some function of". If $h_{t-2,i}$ is required, the same value as $h_{t-1,i}$ can be used. In turn, the fed back value (6.19) is actually

$$\frac{1 - P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right)}{P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right)} = \frac{1}{\text{odds}\left([1 - z_{t-1}]\, p_i + z_{t-1} h_{t-1,i}\right)}, \tag{6.26}$$

with the logarithm of the reciprocal being the additive term inside the exponential. This is illustrated in Figure 6.3 where,

$$f(\cdot) = \text{logit}\left([1 - z_{t-1}]\, p_i + z_{t-1} h_{t-1,i}\right). \tag{6.27}$$

In Figure 6.3, note that the unit-wise recurrence is probabilistic, but an *ad-hoc* layer-wise and gate recurrence are also retained for comparison with a GRU. The $\boldsymbol{h}_{t-2}$ term in this and later cases arises to maintain a consistent definition of z_t across the LSTM, GRU and (6.69); we note that, in practice, the extra delay makes no difference in performance.

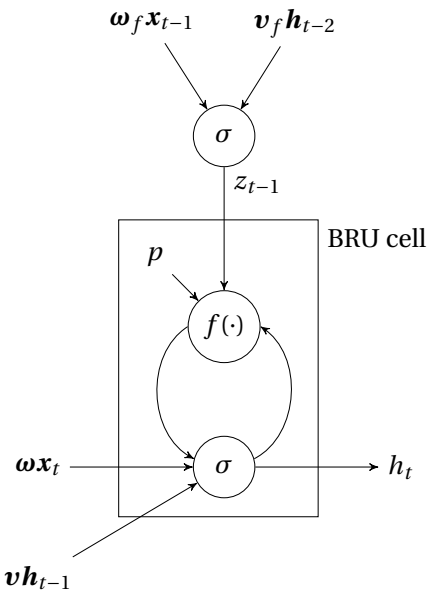


Figure 6.3: A Bayesian recurrent unit incorporating a probabilistic forget gate. An ad-hoc layer-wise and gate recurrence are retained.

6.4.2 Discussion

The unit-wise recursion above was an attempt to formalise the "constant error carousel" (CEC) — the central recurrence — of the LSTM. Whilst the result is self consistent, in practice we find two difficulties:

1. The logit function of (6.27) causes instability in the training process. This is because it can tend to $\pm\infty$.
2. The formulation does not explain the layer-wise recursion around the whole layer of units.

In the following, we address both of these difficulties using approximations. We find that the resulting layer-wise recursion is both stable and more complete.

6.4.3 Layer-wise Recursion

In contrast to the unit-wise recursion above, here we take the elements of $\boldsymbol{\phi}$ to be dependent, meaning the summation is over the whole vector. The main derivation is (6.28)–(6.30) below,

$$P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right) = \sum_{\boldsymbol{\phi}_{t-1}} \sum_{\zeta_{t-1}} P\left(\phi_{t,i} \mid \boldsymbol{\phi}_{t-1}, \zeta_{t-1}, \boldsymbol{X}_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1} \mid \boldsymbol{X}_{t-1}\right) P\left(\zeta_{t-1} \mid \boldsymbol{X}_{t-1}\right) \tag{6.28}$$

$$\begin{aligned} &= P\left(\zeta_{t-1} \mid \boldsymbol{X}_{t-1}\right) \sum_{\boldsymbol{\phi}_{t-1}} P\left(\phi_{t,i} \mid \boldsymbol{\phi}_{t-1}, \zeta_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1} \mid \boldsymbol{X}_{t-1}\right) \\ &\quad + P\left(\bar{\zeta}_{t-1} \mid \boldsymbol{X}_{t-1}\right) \sum_{\boldsymbol{\phi}_{t-1}} P\left(\phi_{t,i} \mid \boldsymbol{\phi}_{t-1}, \bar{\zeta}_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1} \mid \boldsymbol{X}_{t-1}\right). \end{aligned} \tag{6.29}$$

$$\begin{aligned} &= z_{t-1} \sum_{\boldsymbol{\phi}_{t-1}} P\left(\phi_{t,i} \mid \boldsymbol{\phi}_{t-1}, \zeta_{t-1}\right) \prod_i P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) \\ &\quad + (1 - z_{t-1}) \sum_{\boldsymbol{\phi}_{t-1}} P\left(\phi_{t,i} \mid \boldsymbol{\phi}_{t-1}, \bar{\zeta}_{t-1}\right) \prod_i P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right). \end{aligned} \tag{6.30}$$

The calculation can be rendered tractable if we model $P\left(\phi_{t,i} \mid \boldsymbol{\phi}_{t-1}, \zeta_{t-1}\right)$ as $\boldsymbol{\omega}_i^\top \boldsymbol{\phi}_{t-1}$, where $\boldsymbol{\omega}_i$ is a trainable vector and each element $\omega_{j,i}$ models the weight that $\phi_{t-1,j}$ has on $\phi_{t,i}$. The occurrence of $\phi_{t,i}$ is considered to be the weighted average of the occurrences of $\boldsymbol{\phi}_{t-1}$. This is an extension of unit-wise recursion where the occurrence of $\phi_{t,i}$ only depends on $\phi_{t-1,i}$ and $\boldsymbol{\omega}_i$ is a one-hot vector with $\omega_{i,i} = 1$. Therefore, we have

$$\sum_{\boldsymbol{\phi}_{t-1}} P\left(\phi_{t,i} \mid \boldsymbol{\phi}_{t-1}, \zeta_{t-1}\right) \prod_i P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) = \boldsymbol{\omega}_i^\top \boldsymbol{h}_t + c \tag{6.31}$$

where, $\boldsymbol{\omega}_i$ denotes the i^{th} column of $\boldsymbol{\omega}$ and $c = \sum_{j \in \{j \mid \omega_{j,i} < 0\}} \omega_{j,i}$. To understand the above equation, consider N independent lotteries, where N is the total number of nodes in a layer. The winning rate of the i^{th} lottery is h_i, the corresponding prize is ω_i. Now we buy each of the lottery once. The left side of the above equation actually calculate the expectation of the total prizes we can win from the lotteries by listing all the possibilities. On the other hand, each

lottery is independent. Therefore, the expectation prize for i^{th} lottery is $\omega_i h_i$. The expectation of the total prizes we can get is then $\boldsymbol{\omega}_i^\top \boldsymbol{h}$.

In this sense, the recursion is parameterised by matrix $\boldsymbol{\omega}$. Given the fact that $\boldsymbol{\omega}_i^\top \boldsymbol{\phi}_{t-1}$ represents probabilities and the expectation of probabilities should be positive, it is sensible to constrain the L_1 norm of each column in $\boldsymbol{\omega}$ to 1 and add the bias term c. Thus, (6.30) can be written as

$$P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right) = z_{t-1}(\boldsymbol{\omega}_i^\top \boldsymbol{h}_{t-1} + c - p_i) + p_i \tag{6.32}$$

This is illustrated in Figure 6.4, where,

$$f(\cdot) = \text{logit}\Big(z_{t-1}(\boldsymbol{\omega}_i^\top \boldsymbol{h}_{t-1} + c - p_i) + p_i\Big). \tag{6.33}$$

Note that in Figure 6.4, the unit-wise and layer-wise recurrence are combined into a single probabilisitic recurrence. However, the ad-hoc gate recurrence is retained.

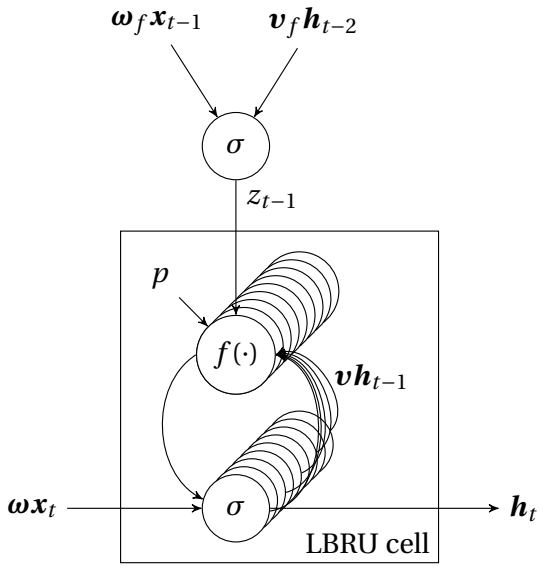


Figure 6.4: The layer-wise recursion with a forget gate.

With reference to Figure 6.5, the function $\log\left(\frac{h}{1-h}\right)$ appears linear except for narrow regions close to 0 and 1. Since we are not aware of the distribution of h, we further approximate $\log\left(\frac{h}{1-h}\right) \approx \alpha h + \beta$, yielding

$$\log\left(\frac{P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right)}{1 - P\left(\phi_{t,i} \mid \boldsymbol{X}_{t-1}\right)}\right) \approx z_{t-1}(\boldsymbol{\omega}_i^\top \boldsymbol{h}_{t-1} + c - p_i) + p_i + \beta, \tag{6.34}$$

where α is absorbed by $\boldsymbol{\omega}_i^\top$ and p_i. The range of α is $[4, +\infty)$. Therefore, we do not normalise $\boldsymbol{\omega}_i$ in the forward pass.

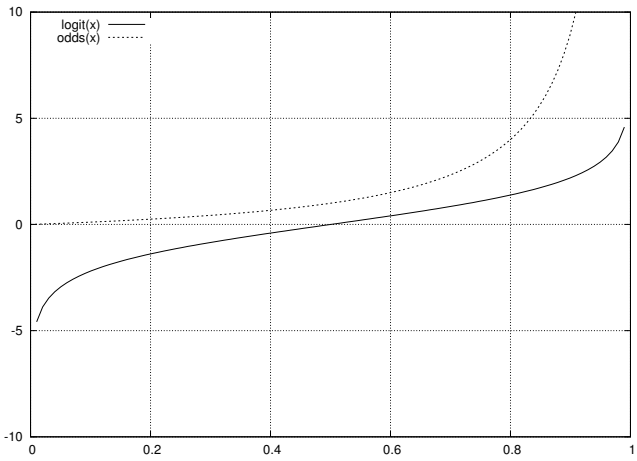


Figure 6.5: Logit and odds curves.

Substituting back into (6.18), that equation can be rewritten as:

$$\boldsymbol{h}_t = \sigma(\boldsymbol{\omega}_{ih}\boldsymbol{x}_t + \mathbf{b}_{ih} + \mathbf{z}_{t-1} \odot (\boldsymbol{\omega}_{hh}\boldsymbol{h}_{t-1} + \mathbf{b}_{hh})) \tag{6.35}$$

which is quite similar to the function of the reset gate in a GRU:

$$\mathbf{n}_t = \tanh(\boldsymbol{\omega}_{in}\boldsymbol{x}_t + \mathbf{b}_{in} + \mathbf{r}_t \odot (\boldsymbol{\omega}_{hn}\boldsymbol{h}_{t-1} + \mathbf{b}_{hn})) \tag{6.36}$$

Besides the activation function, another main difference is that the forget gate z_{t-1} is computed in the previous time step. If z_{t-1} degrades to a constant 1, we get the formulation of a basic recurrent layer that is used in practice.

6.5 Backward Recursion

The recursions described thus far only yield accurate probabilities at time $t = T$. The earlier ones ($1 < t < T$) depend upon future observations. This is normally corrected via the backward passes of either the Kalman smoother or forward-backward algorithm. In this section, we derive backward recursions for the recurrent units derived above. In fact, the ability to do this is one of the most compelling reasons to derive probabilistic recurrence.

6.5.1 Unit-wise Recursion

Although the unit-wise recurrence (without approximations) is unstable, it turns out to be beneficial (see section 6.7) to derive the backward pass. It can be done without adding extra parameters, making it directly comparable to the GRU.

Following the method for the Kalman smoother, we first integrate over the state at time t and

the context variable,

$$P\left(\phi_{t-1,i} \mid \boldsymbol{X}_t\right) = \sum_{\phi_{t,i},\zeta_{t-1}} P\left(\phi_{t-1,i} \mid \phi_{t,i}, \zeta_{t-1}, \boldsymbol{X}_t\right) P\left(\phi_{t,i}, \zeta_{t-1} \mid \boldsymbol{X}_t\right) \tag{6.37}$$

$$\begin{aligned} &= P\left(\phi_{t-1,i} \mid \phi_{t,i}, \zeta_{t-1}, \boldsymbol{X}_t\right) h_{t,i} z_{t-1} \\ &\quad + P\left(\phi_{t-1,i} \mid \bar{\phi}_{t,i}, \zeta_{t-1} \boldsymbol{X}_t\right) (1 - h_{t,i}) z_{t-1} \\ &\quad + P\left(\phi_{t-1,i} \mid \phi_{t,i}, \bar{\zeta}_{t-1}, \boldsymbol{X}_t\right) h_{t,i} (1 - z_{t-1}) \\ &\quad + P\left(\phi_{t-1,i} \mid \bar{\phi}_{t,i}, \bar{\zeta}_{t-1} \boldsymbol{X}_t\right) (1 - h_{t,i})(1 - z_{t-1}). \end{aligned} \tag{6.38}$$

Note that, given $\phi_{t,i}$, $P\left(\phi_{t-1,i}\right)$ is conditionally independent of any data after time $t-1$. (6.39)–(6.46) show how to use Bayes's theorem to expand the remaining terms.

$$P\left(\phi_{t-1,i} \mid \phi_{t,i}, \zeta_{t-1}, \boldsymbol{X}_t\right) = \frac{P\left(\phi_{t,i} \mid \phi_{t-1,i}, \zeta_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)}{\sum_{\phi_{t-1,i}} P\left(\phi_{t,i} \mid \phi_{t-1,i}, \zeta_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)} \tag{6.39}$$

$$= \frac{1 \times h_{t-1,i}}{1 \times h_{t-1,i} + 0 \times (1 - h_{t-1,i})} = 1. \tag{6.40}$$

$$P\left(\phi_{t-1,i} \mid \bar{\phi}_{t,i}, \zeta_{t-1}, \boldsymbol{X}_t\right) = \frac{P\left(\bar{\phi}_{t,i} \mid \phi_{t-1,i}, \zeta_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)}{\sum_{\phi_{t-1,i}} P\left(\bar{\phi}_{t,i} \mid \phi_{t-1,i}, \zeta_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)} \tag{6.41}$$

$$= \frac{0 \times h_{t-1,i}}{0 \times h_{t-1,i} + 1 \times (1 - h_{t-1,i})} = 0. \tag{6.42}$$

$$P\left(\phi_{t-1,i} \mid \phi_{t,i}, \bar{\zeta}_{t-1}, \boldsymbol{X}_t\right) = \frac{P\left(\phi_{t,i} \mid \phi_{t-1,i}, \bar{\zeta}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)}{\sum_{\phi_{t-1,i}} P\left(\phi_{t,i} \mid \phi_{t-1,i}, \bar{\zeta}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)} \tag{6.43}$$

$$= \frac{p_i h_{t-1,i}}{p_i h_{t-1,i} + p_i (1 - h_{t-1,i})} = h_{t-1,i}. \tag{6.44}$$

$$P\left(\phi_{t-1,i} \mid \bar{\phi}_{t,i}, \bar{\zeta}_{t-1}, \boldsymbol{X}_t\right) = \frac{P\left(\bar{\phi}_{t,i} \mid \phi_{t-1,i}, \bar{\zeta}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)}{\sum_{\phi_{t-1,i}} P\left(\bar{\phi}_{t,i} \mid \phi_{t-1,i}, \bar{\zeta}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)} \tag{6.45}$$

$$= \frac{(1 - p_i) h_{t-1,i}}{(1 - p_i) h_{t-1,i} + (1 - p_i)(1 - h_{t-1,i})} = h_{t-1,i}. \tag{6.46}$$

Putting the above together, we initialise

$$h'_{T,i} = h_{T,i} \tag{6.47}$$

then recurse

$$h'_{t-1} = P\left(\phi_{t-1,i} \mid \boldsymbol{X}_t\right) \tag{6.48}$$

$$\begin{aligned} &= h'_{t,i} z_{t-1} + h_{t-1,i} h'_{t,i} (1 - z_{t-1}) + h_{t-1,i} (1 - h'_{t,i})(1 - z_{t-1}) \\ &= (1 - z_{t-1}) h_{t-1,i} + z_{t-1} h'_{t,i}. \end{aligned} \tag{6.49}$$

6.5.2 Layer-wise Recursion

Now we consider the case that $\phi_{t-1,i}$ is dependent on the whole vector $\boldsymbol{\phi}_t$.

$$P\left(\phi_{t-1,i} \mid \boldsymbol{X}_t\right) = \sum_{\boldsymbol{\phi}_t, \zeta_{t-1}} P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \zeta_{t-1}, \boldsymbol{X}_t\right) P\left(\boldsymbol{\phi}_t, \zeta_{t-1} \mid \boldsymbol{X}_t\right) \tag{6.50}$$

$$\begin{aligned} &= z_{t-1} \sum_{\boldsymbol{\phi}_t} P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \zeta_{t-1}, \boldsymbol{X}_t\right) P\left(\boldsymbol{\phi}_t \mid \boldsymbol{X}_t\right) \\ &\quad + (1 - z_{t-1}) \sum_{\boldsymbol{\phi}_t} P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \bar{\zeta}_{t-1}, \boldsymbol{X}_t\right) P\left(\boldsymbol{\phi}_t \mid \boldsymbol{X}_t\right) \end{aligned} \tag{6.51}$$

(6.52)–(6.56) show how to use use Bayes's theorem to expand the remaining terms,

$$P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \zeta_{t-1}, \boldsymbol{X}_t\right) = \frac{P\left(\boldsymbol{\phi}_t \mid \phi_{t-1,i}, \zeta_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)}{\sum_{\boldsymbol{\phi}_{t-1}} P\left(\boldsymbol{\phi}_t \mid \boldsymbol{\phi}_{t-1}, \zeta_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1} \mid \boldsymbol{X}_{t-1}\right)} \tag{6.52}$$

$$= \frac{\sum_{\boldsymbol{\phi}_{t-1,\bar{i}}} P\left(\boldsymbol{\phi}_t \mid \phi_{t-1,i}, \boldsymbol{\phi}_{t-1,\bar{i}}, \zeta_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1,\bar{i}} \mid \boldsymbol{X}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)}{\sum_{\boldsymbol{\phi}_{t-1}} P\left(\boldsymbol{\phi}_t \mid \boldsymbol{\phi}_{t-1}, \zeta_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1} \mid \boldsymbol{X}_{t-1}\right)} \tag{6.53}$$

$$P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \bar{\zeta}_{t-1}, \boldsymbol{X}_t\right) = \frac{\sum_{\boldsymbol{\phi}_{t-1,\bar{i}}} P\left(\boldsymbol{\phi}_t \mid \phi_{t-1,i}, \boldsymbol{\phi}_{t-1,\bar{i}}, \bar{\zeta}_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1,\bar{i}} \mid \boldsymbol{X}_{t-1}\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right)}{\sum_{\boldsymbol{\phi}_{t-1}} P\left(\boldsymbol{\phi}_t \mid \boldsymbol{\phi}_{t-1}, \bar{\zeta}_{t-1}\right) P\left(\boldsymbol{\phi}_{t-1} \mid \boldsymbol{X}_{t-1}\right)} \tag{6.54}$$

$$= \frac{\prod_k p_k}{\prod_k p_k (1 + (1 - h_{t-1,i})/h_{t-1,i})} \tag{6.55}$$

$$= h_{t-1,i} \tag{6.56}$$

where $\boldsymbol{\phi}_{t-1,\bar{i}}$ denotes the features of all the units in the layer except the i^{th} unit and p_k is the prior probability of unit k. The first term $P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \zeta_{t-1}, \boldsymbol{X}_t\right)$ seems intractable, although it allows us to re-use the weights learnt from the forward pass to smooth the output via backward recursion. Now suppose there is another binary state variable, ξ_t, where $\xi_t = 1$ indicates the future context remaining relevant, meaning that ϕ_t is dependent on ϕ_{t+1} and $\xi = 0$ indicates that the future context is irrelevant. We can assign a new probability, $s_t = P(\xi_t = 1 \mid \boldsymbol{X}_t)$ and the inverse $(1 - s_t) = P(\xi_t = 0 \mid \boldsymbol{X}_t)$. We assume ξ_t is independent of future observations $\boldsymbol{X}_{t+1}^T$. Thus, we can write:

$$P\left(\phi_{t-1,i} \mid \boldsymbol{X}_T\right) = \sum_{\boldsymbol{\phi}_t, \xi_t} P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \xi_{t-1}, \boldsymbol{X}_T\right) P\left(\boldsymbol{\phi}_t, \xi_{t-1} \mid \boldsymbol{X}_T\right) \tag{6.57}$$

$$\begin{aligned} &= s_{t-1} \sum_{\boldsymbol{\phi}_t} P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \xi_{t-1}\right) \prod_k P\left(\phi_{t,k} \mid \boldsymbol{X}_T\right) \\ &\quad + (1 - s_{t-1}) \sum_{\boldsymbol{\phi}_t} P\left(\phi_{t-1,i} \mid \bar{\xi}_{t-1}, \boldsymbol{X}_{t-1}\right) P\left(\boldsymbol{\phi}_t \mid \boldsymbol{X}_T\right) \end{aligned} \tag{6.58}$$

$$= s_{t-1} \sum_{\boldsymbol{\phi}_t} P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \xi_{t-1}\right) \prod_k P\left(\phi_{t,k} \mid \boldsymbol{X}_T\right) + h_{t-1,i}(1 - s_{t-1}). \tag{6.59}$$

Similarly, we model $P\left(\phi_{t-1,i} \mid \boldsymbol{\phi}_t, \xi_{t-1}\right)$ as $\boldsymbol{\omega}_i^\top \boldsymbol{\phi}_t$, the product of a trainable vector $\boldsymbol{\omega}_i$ and $\boldsymbol{\phi}_t$, and denote $h'_{t,i} = P\left(\phi_{t,i} \mid \boldsymbol{X}_T\right)$ and put the above together, we initialise

$$h'_{T,i} = h_{T,i} \tag{6.60}$$

then recurse

$$P\left(\phi_{t-1,i} \mid \boldsymbol{X}_T\right) = h'_{t-1,i} = s_t(\boldsymbol{\omega}_i^\top \boldsymbol{h}'_t + c) + h_{t-1,i}(1 - s_t), \tag{6.61}$$

where $c = \sum_{j \in \{j \mid \omega_{j,i} < 0\}} \omega_{j,i}$. It is sensible to apply the same constraints discussed in Section 6.4.3 to the backward recurrent matrix and add the bias term.

The layer-wise backward pass hence requires extra parameters. In this sense it is not directly comparable to a similar GRU. Nevertheless, the parameter count is smaller than for a bidirectional GRU. The repercussions of this are examined in section 6.7.

6.6 Probabilistic Input

In examining the probabilistic forget derivations above, whilst we set out to formalise the CEC of the LSTM, the result is closer to the reset gate of a GRU. In this section, we show that the update gate of a GRU can also be derived rather simply.

6.6.1 Recursion

In the same spirit as the previous section, say there is a binary state variable, ρ, where $\rho = 1$ indicates the current input is relevant, and $\rho = 0$ indicates that it is not relevant. We can assign a probability, $r_t = P\left(\rho_t = 1 \mid \boldsymbol{X}_t\right)$ and the inverse $(1 - r_t) = P\left(\rho_t = 0 \mid \boldsymbol{X}_t\right)$. We assume if the current input is irrelevant, then ϕ_t is completely dependent on ϕ_{t-1}. For a given feature, ϕ_i, the derivation is shown in (6.62)–(6.66) below.

$$\begin{aligned}
h_{t,i} &= P\left(\phi_{t,i} \mid \boldsymbol{X}_t\right) && (6.62)\\
&= \sum_{\rho_{t,i}} P\left(\phi_{t,i} \mid \boldsymbol{X}_t, \rho_{t,i}\right) P\left(\rho_{t,i} \mid \boldsymbol{X}_t\right) && (6.63)\\
&= P\left(\phi_{t,i} \mid \boldsymbol{X}_t, \rho_{t,i}\right) P\left(\rho_{t,i} \mid \boldsymbol{X}_t\right) + \sum_{\phi_{t-1,i}} P\left(\phi_{t,i} \mid \boldsymbol{X}_t, \phi_{t-1,i}, \bar{\rho}_{t,i}\right) P\left(\bar{\rho}_{t,i} \mid \boldsymbol{X}_t\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_t\right) && (6.64)\\
&\approx P\left(\phi_{t,i} \mid \boldsymbol{X}_t\right) P\left(\rho_{t,i} \mid \boldsymbol{X}_t\right) + \sum_{\phi_{t-1,i}} P\left(\phi_{t,i} \mid \phi_{t-1,i}, \bar{\rho}_{t,i}\right) P\left(\bar{\rho}_{t,i} \mid \boldsymbol{X}_t\right) P\left(\phi_{t-1,i} \mid \boldsymbol{X}_{t-1}\right) && (6.65)\\
&= r_{t,i} P\left(\phi_{t,i} \mid \boldsymbol{X}_t\right) + (1 - r_{t,i}) h_{t-1,i} && (6.66)
\end{aligned}$$

The first term follows the same derivations in previous sections. This is illustrated in Figure 6.6, where, as before, the unit-wise and layer-wise recursions are merged, and the gate recursion

remains ad-hoc; this provides for a fair comparison with GRU in section 6.7.

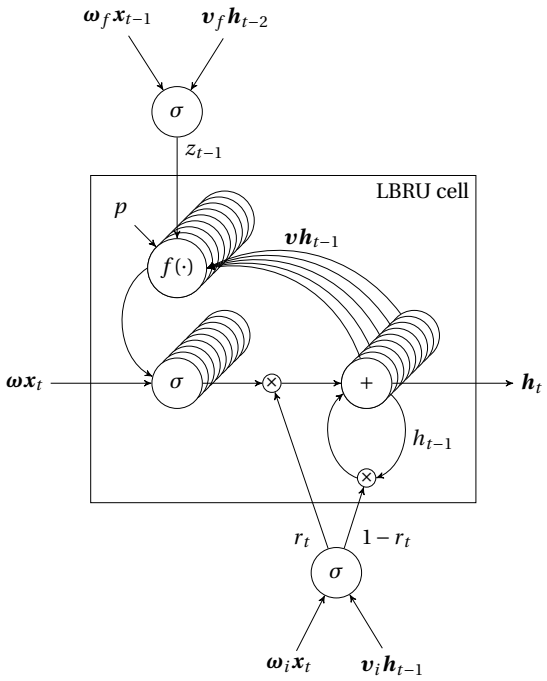


Figure 6.6: The layer-wise recursion with a forget gate and an input gate.

This correlates to the update function in a GRU:

$$\boldsymbol{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{n}_t + \mathbf{z}_t \boldsymbol{h}_{(t-1)}, \tag{6.67}$$

where n_t is defined as (6.36) and z_t is the update gate computed as

$$\mathbf{z}_t = \sigma(\boldsymbol{\omega}_{iz} \boldsymbol{x}_t + \mathbf{b}_{iz} + \boldsymbol{\omega}_{hz} \boldsymbol{h}_{(t-1)} + \mathbf{b}_{hz}) \tag{6.68}$$

It may be argued that the input gate and the forget gate have simliar functionality. Indeed, if we only keep the forget gate and let $z_t = P\left(\rho_t = 0 \mid \boldsymbol{X}_t\right)$, this leads to the MGU [Zhou et al., 2016]; If we keep the forget gate always equal to 1, it leads to the Li-GRU [Ravanelli et al., 2018].

We do not derive a backward recursion for the input gate. Rather, the resulting resemblance to the GRU provides us with a candidate architecture to compare experimentally; this is reported in section 6.7.

6.6.2 Summary

The forward pass of this final BRU can be summarised as:

$$\mathbf{z}_t = \sigma(\boldsymbol{\omega}_{iz}\boldsymbol{x}_t + \boldsymbol{\omega}_{hz}\boldsymbol{h}_{t-1} + \mathbf{b}_z) \tag{6.69}$$

$$\mathbf{r}_t = \sigma(\boldsymbol{\omega}_{ir}\boldsymbol{x}_t + \boldsymbol{\omega}_{hr}\boldsymbol{h}_{t-1} + \mathbf{b}_r) \tag{6.70}$$

$$\mathbf{n}_t = \sigma(\boldsymbol{\omega}_{ih}\boldsymbol{x}_t + \mathbf{b}_{ih} + \mathbf{z}_{t-1} \odot (\boldsymbol{\omega}_{hh}\boldsymbol{h}_{t-1} + \mathbf{b}_{hh})) \tag{6.71}$$

$$\boldsymbol{h}_t = (1 - \mathbf{r}_t) \odot \mathbf{n}_t + \mathbf{r}_t \odot \boldsymbol{h}_{t-1}, \tag{6.72}$$

In the backward pass, two cases can be considered, namely unit-wise BRU (UBRU):

$$\boldsymbol{h}'_{t-1} = \boldsymbol{h}'_t \odot \mathbf{z}_t + \boldsymbol{h}_{t-1} \odot (1 - \mathbf{z}_t) \tag{6.73}$$

and layer-wise BRU (LBRU):

$$\mathbf{s}_t = \sigma(\boldsymbol{\omega}_{is}\boldsymbol{x}_t + \mathbf{b}_{is} + \boldsymbol{\omega}_{hs}\boldsymbol{h}_{t-1} + \mathbf{b}_{hs}) \tag{6.74}$$

$$\boldsymbol{h}'_{t-1} = (\boldsymbol{\omega}_{hhb}\boldsymbol{h}'_t + \mathbf{b}_{hhb}) \odot \mathbf{s}_t + \boldsymbol{h}_{t-1} \odot (1 - \mathbf{s}_t). \tag{6.75}$$

Note that in the above, we retain the ad-hoc gate recurrence as we find that it performs marginally better than not doing so. However, there is currently no probabilistic reason to do so. We set this matter aside for the future. With reference to section 6.4, in defining the gates as vectors, we are assuming one gate per feature; this is usual in LSTM and GRU, but not a constraint.

6.7 Experiments

We present evaluations of the techniques described thus far on ASR tasks. Recurrent networks are particularly suited to ASR as there is an explicit time dimension and well known context dependency. Reciprocally, ASR is a difficult task that has driven recent advances in deep learning [Graves et al., 2006; Seide et al., 2011; Xiong et al., 2017; Hadian et al., 2018].

6.7.1 Hypotheses

In running experiments, we are testing the Bayesian recurrent unit derived in the previous three sections. This raises two explicit hypotheses:

1. We would expect the incorporation of a backward pass to improve upon the performance of a (forward-only) GRU.
2. We would expect the LBRU to approach the performance of a conventional GRU-based BiRNN architecture. It has the same contextual knowledge, but does not have higher representational capability. If it falls short of a BiRNN architecture then either the approximations in the derivation are not valid, or the BiRNN is taking advantage of

temporal asymmetry in the data.

This is all dependent upon the number of parameters: A BiRNN has roughly twice as many parameters as a Bayesian RNN with a backward pass.

6.7.2 Corpora and Method

Detailed statistics of the corpora considered in this work are summarised in Table 6.1.

Table 6.1: Statistics of datasets used in this work: speakers and sentences are counts, the amounts of speech data for training and evaluation sets are in hours.

Dataset	Speakers	Sentences	Train	Eval
TIMIT	462	3696	5	0.16
WSJ	283	37416	81.3	0.7
AMI-IHM	10487	98397	70.3	8.6

A first set experiments with the TIMIT corpus [Garofolo et al., 1993] was performed to test the proposed model for a phoneme recognition task. We used the standard 462-speaker training set and removed all SA records, since they may bias the results. A separate development set of 50 speakers was used for tuning all meta-parameters including the learning schedule and multiple learning rates. Results are reported using the 24-speaker core test set, which has no overlap with the development set. Following the implementation of [Ravanelli et al., 2018, 2019], all the recurrent networks tested on this dataset have 5 layers, each consisting 550 units in each direction and use 40 fMLLR features (extracted based on the Kaldi recipe) as the input.

The second set of experiments was carried out on the Wall Street Journal (WSJ) speech corpus to gauge the suitability of the proposed model for large vocabulary speech recognition. We used the standard configuration si284 dataset for training, dev93 for tuning hyper-parameters, and eval92 for evaluation. All the tested recurrent networks have 3 layers, each consisting of 320 units in each direction. We used 40 fMLLR features as input for speaker adaptation.

The TIMIT and WSJ datasets yield results with modest statistical significance. In order to yield more persuasive significance, a set of experiments was also conducted on the AMI corpus [Carletta et al., 2005] with the data recorded through individual headset microphones (IHM). The AMI corpus contains recordings of spontaneous conversations in meeting scenarios, with 70 hours of training data, 9 hours of development, and 8 hours of test data. All the tested recurrent networks have 3 layers, each consisting of 512 units in each direction and use 40 fMLLR features as the input.

Lastly, the RNN architectures are evaluated on a multilingual ASR task using the same Global-Phone dataset with the IPA-based universal phone set. All the tested recurrent networks have 4 layers, each consisting of 320 units in each direction and use 40 MFCC features as the input.

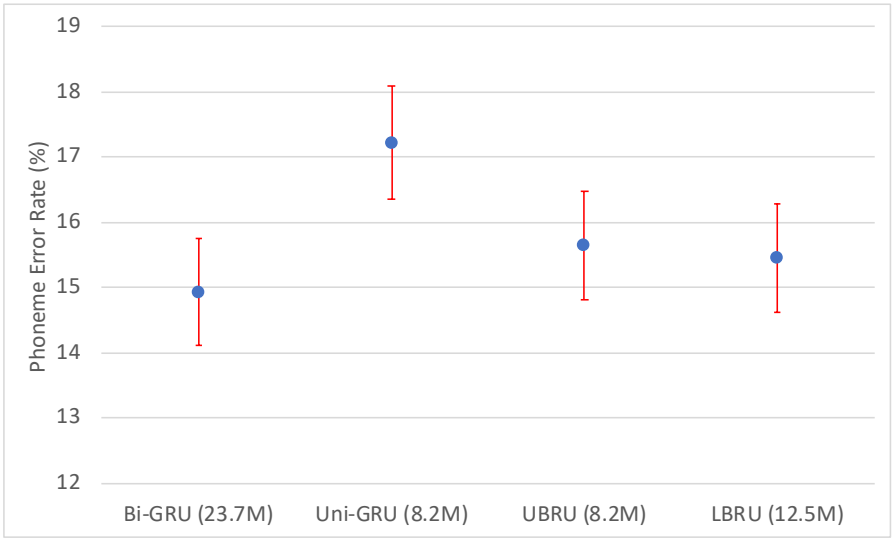


Figure 6.7: Phoneme Error Rate (%) on TIMIT for various RNN architectures. The numbers in the parentheses indicate the number of parameters each model contains. The error bars indicate equal-tailed 95% credible interval for a beta assumption for the error rate.

We evaluated the ASR performance using NN/HMM hybrid framework as the training is faster. The neural networks were trained to predict context-dependent phone targets. The labels were derived by performing a forced alignment procedure on the training set using GMM/HMM, as in the standard recipe of Kaldi[2] [Povey et al., 2011]. During testing, the posterior probabilities generated for each frame by the neural networks are normalised by their priors, then processed by an HMM-based decoder, which estimates the sequence of words by integrating the acoustic, lexicon and language model information. The neural networks of the ASR system were implemented in PyTorch[3], including, crucially, the gradient calculation; they were coupled with the Kaldi decoder [Povey et al., 2011] to form a context-dependent RNN/HMM speech recogniser.

6.7.3 Training Details

The network architecture adopted for the experiments contains multiple recurrent layers, which are stacked together prior to the final softmax context-dependent (senon) classifier. If the networks are bidirectional, the forward hidden states and the backward hidden states at each layer are concatenated before feeding to the next layer. A dropout rate of 0.2 was used for regularisation. Moreover, batch normalization [Ioffe and Szegedy, 2015] was adopted on each layer to accelerate the training. The optimization was performed using the Adaptive Moment Estimation (Adam) algorithm [Kingma and Ba, 2014] running for 24 epochs with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$. The performance on the cross validation set was monitored after each epoch, while the learning rate was halved when the performance improvement

[2] http://kaldi-asr.org/
[3] https://pytorch.org/

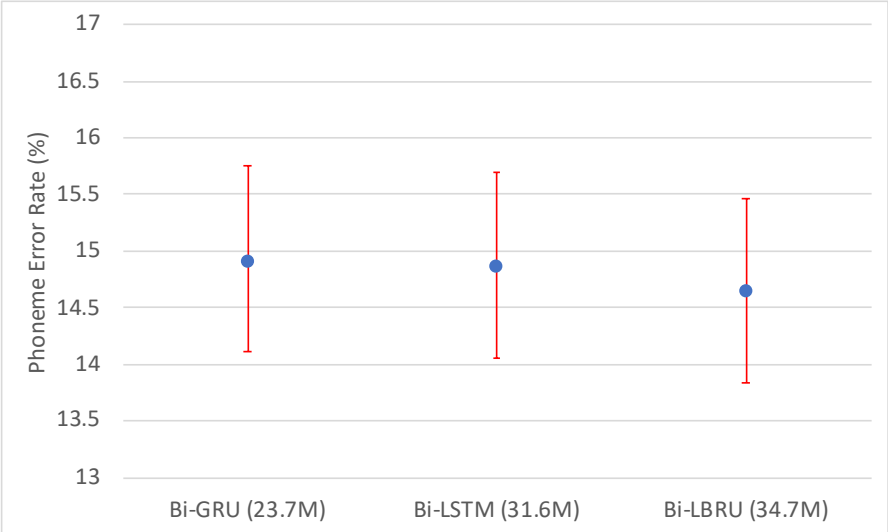


Figure 6.8: Phoneme Error Rate (%) on TIMIT for various RNN architectures. The numbers in the parentheses indicate the number of parameters each model contains. The error bars indicate equal-tailed 95% credible interval for a beta assumption for the error rate.

dropped below a certain threshold (0.001).

6.7.4 Phoneme Recognition Performance on TIMIT

In order to confirm the suitability of the proposed model for acoustic modeling, TIMIT was first considered to reduce the linguistic effects (such as lexicon and language model) on the performance evaluation. The state of the art for this task is probably that of Ravanelli et al. [2018], with a phone error rate (PER) of 14.9%. We duplicate the architecture of those authors and aim for a similar figure. We performed the comparison with GRU as shown in Figure 6.7. The error bars indicate equal-tailed 95% credible interval for a beta assumption for the error rate. The numbers in the parentheses indicate the number of parameters each model contains. It is clear that the unidirectional GRU (Uni-GRU) is significantly worse than bidirectional GRU (Bi-GRU) as the credible intervals do not overlap. By contrast, the unit-wise BRU (UBRU) yields much better performance compared to Uni-GRU with exactly the same model size, and the layer-wise BRU (LBRU) is slightly better than UGRU, yielding similar performance to Bi-GRU.

Since the test set in TIMIT is quite small, we also performed a matched-pair t-test between Uni-GRU and UBRU, the test statistic being the utterance-wise difference in word-level errors normalised by the reference length. This yields $p < 0.001$, showing that the UBRU is significantly better. This confirms our first hypothesis that the incorporation of a backward pass can improve upon the performance of a unidirectional GRU. The t-test between Bi-GRU and LBRU yields $p = 0.230$, which implies there is no significant difference between the two systems. The two comparisons together show that our proposed model can achieve performance

indistinguishable from the Bi-GRU, without the explicit extra backward recurrence.

Although the difference between Bi-GRU and LBRU is not significant, the latter one is slightly worse. This can be explained by our second hypothesis. Physiological filters are known to have asymmetric impulse responses [Honnet et al., 2018]. This is one explanation for the large improvement arising from doubling up the Uni-GRU to explicitly modelling the backward recursion. However, the proposed BRU does not have the explicit extra backward recurrence of the BiRNN architectures. Therefore, we further doubled up the LBRU to be explicitly bidirectional and compared it with Bi-GRU and Bi-LSTM, as shown in Figure 6.8. Similarly, we plot the error bars and the sizes of the models; this shows that GRU and LSTM perform almost the same while the Bi-LBRU seems to be slightly better with a few more parameters, although the difference is insignificant from the t-test ($p = 0.43$). Our hypothesis is that BRU has a stronger modelling ability in each of the directions because the prediction is always conditioned on the whole sequence due to the implicit backward recursion. We note that the average PER of 14.6% obtained with Bi-LBRU outperforms the state of the art 14.9% of Ravanelli et al. [2018] on the TIMIT test-set, although it is well within the 95% confidence bounds.

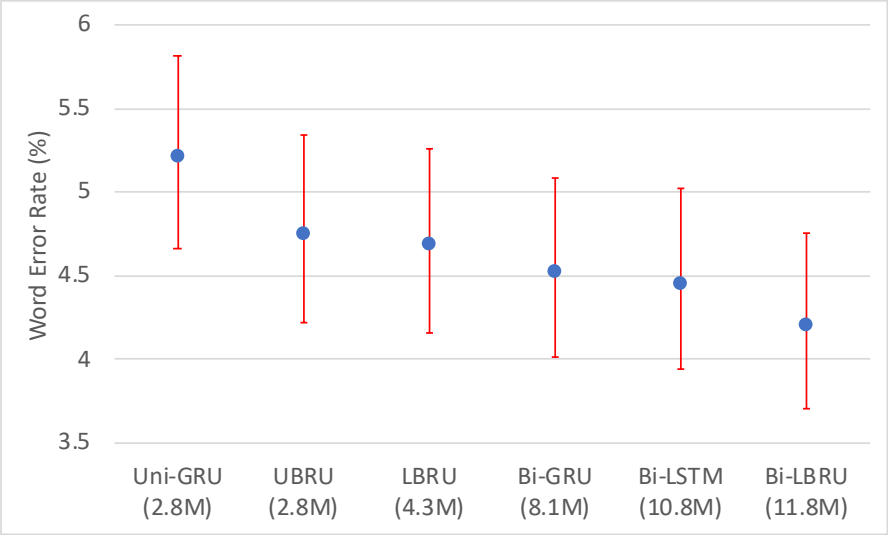


Figure 6.9: Word Error Rate (%) on WSJ for various RNN architectures. The numbers in the parentheses indicate the number of parameters each model contains. The error bars indicate equal-tailed 95% credible interval for a beta assumption for the error rate.

6.7.5 Speech Recognition Performance on WSJ

Since TIMIT is too small to yield significant comparisons, in this sub-section, we evaluate the RNNs on WSJ, a large vocabulary continuous speech recognition task. Following the TIMIT case, we plot the word error rate (WER) in Figure 6.9, together with the corresponding error bars and model sizes. These results exhibit a similar trend to that observed on TIMIT. Both UBRU and LBRU outperform the Uni-GRU ($p = 0.19$ from the t-test). LBRU is slightly

better than UBRU and it yields very similar performance to the that of Bi-GRU ($p = 0.21$ from the t-test). The Bi-LBRU still performs slightly better than Bi-GRU and Bi-LSTM. Again, the differences are not significant owing to the fact that the test set of WSJ is still quite small. Overall, the results are comparable with the baselines reported in the Kaldi software; for instance, 4.27% using a Bi-LSTM and i-vectors.

6.7.6 Speech Recognition Performance on AMI

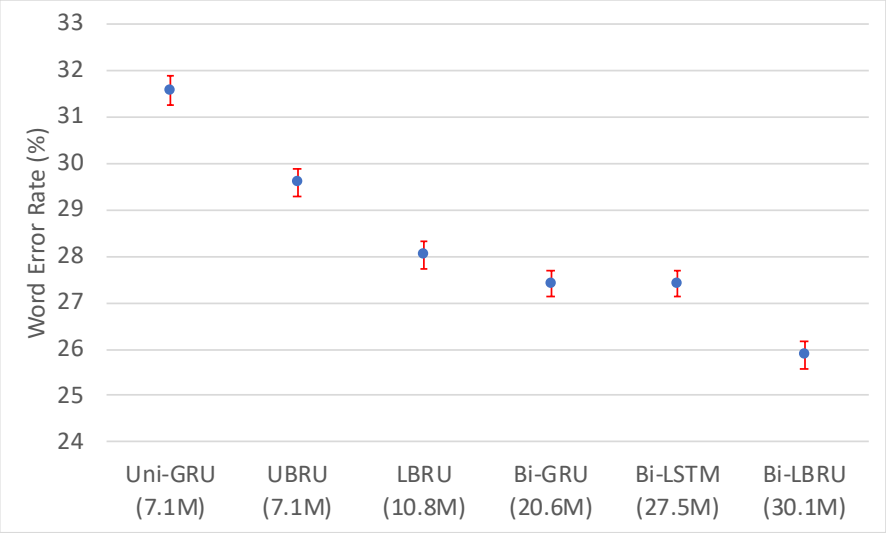


Figure 6.10: Word Error Rate (%) on AMI for various RNN architectures. The numbers in the parentheses indicate the number of parameters each model contains. The error bars indicate equal-tailed 95% credible interval for a beta assumption for the error rate.

Owing to the small test set of WSJ, in this sub-section we conduct the evaluation on AMI, which is a more challenging task with a much larger test set. AMI is more challenging as the data is recorded in meetings, capturing natural spontaneous conversations between participants who play different roles in the meeting. Overlapping speech segments appear in both training and testing. State of the art results on AMI tend to be for complicated systems with elements of speaker and environment adaptation, e.g., Kanda et al. [2018] report a WER of 17.84%. Rather than aim to duplicate such results, we simply aim for a self-consistent comparison of techniques; our results are in the same range as the 26.8% of Dighe et al. [2018].

Figure 6.10 summarises the results obtained on AMI. These results show the same trend as previous experiments, but also exhibit more significant differences. Both UBRU and LBRU significantly outperform Uni-GRU while LBRU is also significantly better than UBRU ($p < 0.001$ from the t-test), showing that the layer-wise backward recursion is able to capture richer characteristics in the backward transition. Comparison between LBRU and Bi-GRU shows that LBRU can achieve similar performance without an extra explicit backward network. Bi-LSTM does not have any advantages over Bi-GRU, although it contains one more gate and, therefore, more parameters. However, if we double up the LBRU to be explicitly bidirectional, the

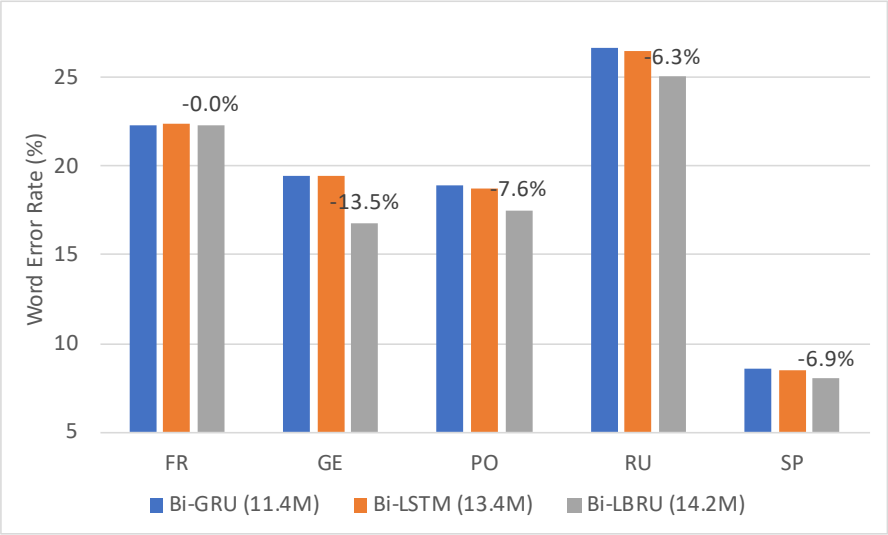


Figure 6.11: Word Error Rate (%) on the 5 selected languages from GlobalPhone for various RNN architectures. The numbers in the parentheses indicate the number of parameters each model contains. The numbers on top of the columns are the relative WER reduction of BRU compared with GRU architecture.

model yields significantly better performance than both Bi-GRU and Bi-LSTM ($p < 0.001$ from the t-test). This confirms the hypothesis that BRU has a stronger unidirectional modelling ability and explicit bidirectional modelling can help capture the asymmetric characteristics in physiological filters.

6.7.7 Multilingual ASR on GlobalPhone

We further evaluated and compared these RNN architectures on the multilingual ASR task using GlobalPhone dataset. Figure 6.11 summarizes the WERs on the 5 selected languages from GlobalPhone and similar trend can be observed. Bi-GRU and Bi-LSTM yield similar performance while Bi-LBRU gives consistent improvement on almost all the languages except for FR. These results further confirm that the proposed Bayesian recurrent unit is beneficial for general ASR tasks as well as multilingual training.

6.8 Conclusion

Given a probabilistic interpretation of common neural network components, it is possible to derive recurrent components in the same spirit. Such components have two advantages:

1. The architecture of the recursion is dictated by the probabilistic formulation, removing otherwise ad-hoc choices.

2. They naturally support a backward recursion of the type used in Kalman smoothers and

the forward-backward algorithm of the HMM.

Unit-wise recursions follow analytically, but are found to lead to instabilities. Approximations lead to stable layer-wise recursions. Nevertheless, useful backward recursions can be derived for both cases. The resulting Bayesian recurrent unit (BRU) can be configured with a probabilistic input gate, being directly comparable to a common GRU.

Evaluation on simple and on state of the art speech recognition tasks shows that:

1. Even the unit-wise backward recursion can out-perform a standard GRU.
2. A more involved layer-wise backward recursion can approach the performance of a bidirectional GRU. This shows that the approximations in the derivations are reasonable.

Further, an explicit bidirectional BRU can out-perform a state of the art bidirectional GRU.

There are some ad-hoc methods in our approach: the gate recurrences are retained for performance; some approximations may be better formulated. These remain matters for future research. Nevertheless, we have shown that recurrence in neural networks can be formulated much more rigorously than conventional wisdom would hold. This in turn can lead to significant performance advantages.

7 Conclusion and future directions

In this chapter, Section 7.1 summarizes the conclusions of this thesis and Section 7.2 discusses the directions of future research.

7.1 Conclusions

In this thesis, we addressed the acoustic modeling issues in general for ASR, with a particular focus on multilingual ASR and cross-lingual adaptation. We explored phoneme-based acoustic modeling for multilingual training and adaptation. CTC and end-to-end LF-MMI training were systematically investigated and compared with conventional DNN/HMM hybrid systems. Through experimental evaluation, we found that sequence-level training criteria are more theoretically rigorous but are also more sensitive to the amount of training data. Thus, they benefit more from multilingual training when language-specific data is limited to build robust acoustic models. It was demonstrated in our experiment that phoneme-based multilingual LF-MMI model outperforms both multilingual CTC models and state-of-the-art DNN/HMM systems.

In order to address this data impurity problem arising from mixture of multilingual data and improve the multilingual ASR in general, we studied language adaptive training approaches. It was demonstrated that approaches such as LHUC and CAT, originating from speaker adaptive training, also work for language adaptation and they can be considered as particular cases of MoE. Applying language adaptive training on all the hidden layers is more beneficial and approaches such as MoE that have stronger modeling capacity perform better.

We further demonstrated that phoneme-based multilingual model is a competitive alternative in fast language adaptation of an ASR system. We took phoneme-based CTC training as an example and showed that the universal phoneme-based multilingual CTC is extensible to new phonemes during cross-lingual adaptation. The extended model converges faster and better on shared phonemes and also catch up quickly on newly added phonemes. Combined with dropout and the proposed parameter initialization during cross-lingual adaptation, the

CTC-based model shows much better performance than DNN/HMM-based adaptation on limited data, potentially making the CTC model a competitive alternative for fast cross-lingual adaptation.

Then, we addressed the data scarcity issue by developing a novel dropout-based semi-supervised training approach to exploit unlabeled data. It was demonstrated that the pseudo-transcriptions sampled from different dropout-based decoding results lead to an unbiased supervision lattices for semi-supervised training and it is able to help reduce the confusion of the lattice paths, while keeping variations for uncertain unlabeled utterances. Experiments shows that the proposed approach can further improve the WER over the regular semi-supervised training framework.

Lastly, we derived a novel recurrent architecture with probabilistic explanation, which naturally supports a backward recursion. Experimental evaluation confirms that the proposed architecture can perform as well as a bidirectional RNN as a unidirectional one with the same number of parameters. Further, it can exceed the performance of a conventional bidirectional RNN when configured explicitly bidirectionally.

Had time allowed, it would have made sense to apply the techniques of the semi-supervised training presented in Chapter 5 on the multilingual problem. However, the resulting novel techniques could best be presented monolingually. Multilingual evaluation is clearly a matter for future research.

In conclusion, comprehensive experiments showed that the phoneme-based multilingual model can be a competitive alternative for multilingual ASR and fast cross-lingual adaptation. Theoretical analysis further consolidated the experimental validation and also provided critical understanding of the recurrent neural networks in the context of deep learning.

7.2 Potential Future Research Directions

In this thesis, we mainly focused on multilingual acoustic modeling problems and showed how to exploit multilingual acoustic training data to improve the performance of ASR systems for language with only limited amount of data. However, how to efficiently handle code-switching speech remains a very challenging research problem. End-to-end approaches are potential solutions. For instance, character-based end-to-end model is able to generate good recognition results without external language model given enough training data. Multilingual training on such models might also implicitly learn a multilingual language model. It is of great interest to investigate the performance of such framework on code-switching tasks. Possible research directions include but are not limited to: creating code-switching training set with labels of where the code-switching happens to guide model to detect the place of the code-switch; applying multi-task training with language identification as the secondary task to explicitly improve the ability to distinguish different languages; integrating an additional language detector into the end-to-end ASR framework and adapt the model accordingly.

As mentioned in Chapter 6, there are still some ad-hoc methods in our Bayesian recurrent unit: the gate recurrences are retained for performance; some approximations that have been used may be better formulated. These remain matters for future research. It is also of great interest to investigate streaming ASR using Bayesian recurrent unit. Because Bayesian recurrent unit naturally supports a backward recursion, it is trivial to control the length of future context to be exposed to the model. In other words, it provides an intuitive way to control the trade-off between recognition accuracy and latency. Given the successful experiments for ASR tasks, it also makes sense to extend the Bayesian recurrent unit to other language processing tasks such as machine translation.

Bibliography

Lalit Bahl, Peter Brown, Peter De Souza, and Robert Mercer. Maximum mutual information estimation of hidden Markov model parameters for speech recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 1986.

Lalit R. Bahl, Frederick Jelinek, and Robert L. Mercer. A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(2):179–190, March 1983.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

Leonard E. Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *The Annals of Mathematical Statistics*, 37(6):1554–1563, December 1966.

Jayadev Billa. Improving LSTM-CTC based ASR performance in domains with limited training data. *arXiv preprint arXiv:1707.00722*, 2017.

Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995. ISBN 978-0-198-53864-6.

Hervé Bourlard and Nelson Morgan. *Connectionist speech recognition: a hybrid approach*. 2012.

Hervé Bourlard and C. J. Wellekens. Links between Markov models and multilayer perceptrons. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 1*, pages 502–510. Morgan Kaufmann, 1989.

Hervé A. Bourlard and Nelson Morgan. *Connectionist Speech Recognition: A Hybrid Approach*. The Springer International Series in Engineering and Computer Science. Kluwer Academic Publishers, 1994. doi:10.1007/978-1-4615-3210-1.

John S. Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In Françoise Fogelman Soulié and Jeanny Hérault, editors, *Neurocomputing*, volume 68 of *NATO ASI Series F: Computer and Systems Sciences*, pages 227–236. Springer-Verlag, Berlin Heidelberg, 1990a. doi:10.1007/978-3-642-76153-9_28.

John S. Bridle. Alpha-nets: A recurrent 'neural' network architecture with a hidden Markov model interpretation. *Speech Communication*, 9(1), February 1990b. doi:10.1016/0167-6393(90)90049-F.

J. Carletta, S. Ashby, S. Bourban, M. Guillemot, M. Kronenthal, G. Lathoud, M. Lincoln, I. McCowan, T. Hain, W. Kraaij, W. Post, J. Kadlec, P. Wellner, M. Flynn, and D. Reidsma. The AMI meeting corpus. In *Proceedings of MLMI'05*, Edinburgh, 2005.

William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 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*, 2015.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics.

Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. End-to-end continuous speech recognition using attention-based recurrent NN: first results. *arXiv preprint arXiv:1412.1602*, 2014.

Christopher Cieri, David Miller, and Kevin Walker. The Fisher corpus: a resource for the next generations of speech-to-text. In *LREC*, 2004.

Siddharth Dalmia, Ramon Sanabria, Florian Metze, and Alan W Black. Sequence-based multi-lingual low resource speech recognition. *arXiv preprint arXiv:1802.07420*, 2018.

Najim Dehak, Patrick J Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech and Language Processing*, 2011.

Marc Delcroix, Keisuke Kinoshita, Takaaki Hori, and Tomohiro Nakatani. Context adaptive deep neural networks for fast acoustic model adaptation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2015.

Marc Delcroix, Keisuke Kinoshita, Chengzhu Yu, Atsunori Ogawa, Takuya Yoshioka, and Tomohiro Nakatani. Context adaptive deep neural networks for fast acoustic model adaptation in noisy conditions. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016.

Subhadeep Dey, Petr Motlicek, Trung Bui, and Franck Dernoncourt. Exploiting semi-supervised training through a dropout regularization in end-to-end speech recognition. *arXiv preprint arXiv:1908.05227*, 2019.

Pranay Dighe, Hervé Bourlard, and Afsaneh Asaei. Far-field ASR using low-rank and sparse soft targets from parallel data. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 581–587, Athens, Greece, December 2018.

Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating second-order functional knowledge for better option pricing. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 472–478. MIT Press, 2001.

Stéphane Dupont, Christophe Ris, Olivier Deroo, and Sébastien Poitoux. Feature extraction and acoustic modeling: an approach for improved generalization across languages and accents. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 2005.

Przemyslaw Dymarski. *Hidden Markov Models: Theory and Applications*. BoD–Books on Demand, 2011.

Yarin Gal. *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, 2016.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, 2016.

Mark J. F. Gales. Cluster adaptive training of hidden Markov models. *IEEE transactions on speech and audio processing*, 2000.

Daniel Garcia-Romero and Carol Y. Espy-Wilson. Analysis of i-vector length normalization in speaker recognition systems. In *Proceedings of Interspeech*, 2011.

Philip N Garner and Sibo Tong. A Bayesian approach to recurrence in neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

John S. Garofolo, Lori F. Lamel, William M. Fisher, Jonathan G. Fiscus, and David S. Pallett. DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1. *NASA STI/Recon technical report n*, 93, 1993.

Felix A. Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with LSTM. *Neural Computation*, 12:2451–2471, 2000. doi:10.1162/089976600300015015.

Felix A. Gers, Nicol N. Schraudolph, and Jürgen Schmidhuber. Learning precise timing with LSTM recurrent networks. *Journal of Machine Learning Research*, 3:115–143, August 2002.

Arnab Ghoshal, Pawel Swietojanski, and Steve Renals. Multilingual training of deep neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 315–323, Fort Lauderdale, FL, USA, 2011.

Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional LSTM networks. In *Proceedings of the 2005 IEEE International Joint Conference on Neural Networks*, Montreal, Quebec, Canada, July 2005. doi:10.1109/IJCNN.2005.1556215.

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*, 2006.

Frantisek Grézl, Martin Karafiát, and Karel Veselỳ. Adaptation of multilingual stacked bottleneck neural network structure for new language. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2014.

Frantiseli Grezl and Martin Karafiát. Semi-supervised bootstrapping approach for neural network feature extractor training. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 2013.

Hossein Hadian, Hossein Sameti, Daniel Povey, and Sanjeev Khudanpur. End-to-end speech recognition using lattice-free MMI. In *Proceedings of Interspeech*, 2018.

Georg Heigold, Vincent Vanhoucke, Alan Senior, Patrick Nguyen, M. Ranzato, Matthieu Devin, and Jeffrey Dean. Multilingual acoustic models using distributed deep neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 2012.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9 (8):1735–1780, November 1997. doi:10.1162/neco.1997.9.8.1735.

Pierre-Edouard Honnet, Branislav Gerazov, Aleksandar Gjoreski, and Philip N. Garner. Intonation modelling using a muscle model and perceptually weighted matching pursuit. *Speech Communication*, 97:81–93, March 2018. doi:10.1016/j.specom.2017.10.004.

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 *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013.

David Imseng, Hervé Bourlard, John Dines, Philip N. Garner, and Mathew Magimai.-Doss. Improving non-native ASR through stochastic multilingual phoneme space transformations. In *Proceedings of Interspeech*, Florence, Italy, August 2011.

David Imseng, Hervé Bourlard, and Philip N Garner. Using KL-divergence and multilingual information to improve ASR for under-resourced languages. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2012.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural computation*, 1991.

Frederick Jelinek. *Statistical methods for speech recognition*. MIT press, 1997.

Janez Kaiser, Bogomir Horvat, and Zdravko Kacic. A novel loss function for the overall risk criterion based discriminative training of HMM models. In *Proceedings of the International Conference on Spoken Language Processing*, 2000.

R. E. Kalman. A new approach to linear filtering and prediction problems. *ASME Journal of Basic Engineering*, 82(1):35–45, March 1960. doi:10.1115/1.3662552.

Naoyuki Kanda, Yusuke Fujita, and Kenji Nagamatsu. Lattice-free state-level minimum Bayes risk training of acoustic models. In *Proceedings of Interspeech*, Hyderabad, India, September 2018. doi:10.21437/Interspeech.2018-79.

Alex Kendall and Roberto Cipolla. Modelling uncertainty in deep learning for camera relocalization. *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4762–4769, 2016.

Alex Kendall, Vijay Badrinarayanan, , and Roberto Cipolla. Bayesian SegNet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *arXiv preprint arXiv:1511.02680*, 2015.

Suyoun Kim and Michael L Seltzer. Towards language-universal end-to-end speech recognition. *arXiv preprint arXiv:1711.02207*, 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv:1412.6980, December 2014. URL https://arxiv.org/abs/1412.6980. Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

Brian Kingsbury. Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2009.

K.M. Knill, Mark J.F. Gales, Shakti P. Rath, Philip C. Woodland, Chao Zhang, and S.-X. Zhang. Investigation of multilingual deep neural networks for spoken term detection. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 2013.

Julius Kunze, Louis Kirsch, Ilia Kurenkov, Andreas Krug, Jens Johannsmeier, and Sebastian Stober. Transfer learning for speech recognition on a budget. *arXiv preprint arXiv:1706.00290*, 2017.

Alex Labach, Hojjat Salehinejad, and Shahrokh Valaee. Survey of dropout methods for deep neural networks. *arXiv preprint arXiv:1904.13310*, 2019.

Lori F Lamel, Jean-Luc Gauvain, Mazcine Eskénazi, et al. BREF, a large vocabulary spoken corpus for french1. 1991.

Sheng Li, Xugang Lu, Shinsuke Sakai, Masato Mimura, and Tatsuya Kawahara. Semi-supervised ensemble DNN acoustic model training. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017.

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 *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2009.

Jeff Ma and Richard Schwartz. Unsupervised versus supervised training of acoustic models. In *Ninth Annual Conference of the International Speech Communication Association*, 2008.

David J. C. MacKay. Bayesian interpolation. *Neural Computation*, 4(3):415–447, May 1992a. doi:10.1162/neco.1992.4.3.415.

David J. C. MacKay. A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, May 1992b. doi:10.1162/neco.1992.4.3.448.

David J. C. MacKay. The evidence framework applied to classification networks. *Neural Computation*, 4(5):720–736, May 1992c. doi:10.1162/neco.1992.4.5.720.

Vimal Manohar, Daniel Povey, and Sanjeev Khudanpur. Semi-supervised maximum mutual information training of deep neural network acoustic models. In *Proceedings of Interspeech*, 2015.

Vimal Manohar, Hossein Hadian, Daniel Povey, and Sanjeev Khudanpur. Semi-supervised training of acoustic models using lattice-free MMI. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018.

Yajie Miao, Mohammad Gowayyed, and Florian Metze. EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 2015.

Yajie Miao, Mohammad Gowayyed, Xingyu Na, Tom Ko, Florian Metze, and Alexander Waibel. An empirical exploration of CTC acoustic models. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016.

Markus Miiller, Sebastian Stiiker, and Alex Waibel. Multilingual adaptation of RNN based ASR systems. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018.

Nelson Morgan and Hervé Bourlard. Continuous speech recognition using multilayer perceptrons with hidden Markov models. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 1990.

Markus Müller and Alex Waibel. Using language adaptive deep neural networks for improved multilingual speech recognition. In *Proceedings of the 12th International Workshop on Spoken Language Translation (IWSLT)*, 2015.

Markus Müller, Sebastian Stüker, and Alex Waibel. Language adaptive multilingual CTC speech recognition. In *International Conference on Speech and Computer*, 2017a.

Markus Müller, Sebastian Stüker, and Alex Waibel. Multilingual adaptation of RNN based ASR systems. *arXiv preprint arXiv:1711.04569*, 2017b.

Scott Novotney, Richard Schwartz, and Jeff Ma. Unsupervised acoustic and language model training with small amounts of labelled data. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2009.

Douglas B Paul and Janet M Baker. The design for the Wall Street Journal-based CSR corpus. In *Proceedings of the workshop on Speech and Natural Language*, 1992.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The Kaldi speech recognition toolkit. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 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 *Proceedings of Interspeech*, 2016.

Lawrence R Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Kanishka Rao and Haşim Sak. Multi-accent speech recognition with hierarchical grapheme based models. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017.

Ramya Rasipuram and Mathew Magimai-Doss. Improving articulatory feature and phoneme recognition using multitask learning. In *International Conference on Artificial Neural Networks*, 2011.

Mirco Ravanelli, Philemon Brakel, Maurizio Omologo, and Yoshua Bengio. Improving speech recognition by revising gated recurrent units. In *Proceedings of Interspeech*, pages 1308–1312, Stockholm, Sweden, August 2017. doi:10.21437/Interspeech.2017-775.

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. doi:10.1109/TETCI.2017.2762739.

Mirco Ravanelli, Titouan Parcollet, and Yoshua Bengio. The pytorch-kaldi speech recognition toolkit. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2019.

Nils Reimers and Iryna Gurevych. Optimal hyperparameters for deep LSTM-networks for sequence labeling tasks. *arXiv preprint arXiv:1707.06799*, 2017.

Michael D. Richard and Richard P. Lippman. Neural network classifiers estimate Bayesian a posteriori probabilities. *Neural Computation*, 3(4):461–483, Winter 1991. doi:10.1162/neco.1991.3.4.461.

David E. Rumelhart and James L. McClelland. *Parallel Distributed Processing. Explorations in the Microstructure of Cognition*, volume 1: Foundations. MIT Press, July 1986.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, October 1986. doi:10.1038/323533a0.

Tara N Sainath, Abdel-rahman Mohamed, Brian Kingsbury, and Bhuvana Ramabhadran. Deep convolutional neural networks for LVCSR. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013.

Haşim Sak, Andrew Senior, Kanishka Rao, Ozan Irsoy, Alex Graves, Françoise Beaufays, and Johan Schalkwyk. Learning acoustic frame labeling for speech recognition with recurrent neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2015.

Lahiru Samarakoon and Khe Chai Sim. Subspace LHUC for fast adaptation of deep neural network acoustic models. In *Proceedings of Interspeech*, 2016.

George Saon, Hagen Soltau, David Nahamoo, and Michael Picheny. Speaker adaptation of neural network acoustic models using i-vectors. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 2013.

Stefano Scanzio, Pietro Laface, Luciano Fissore, Roberto Gemello, and Franco Mana. On the use of a multilingual neural network front-end. In *Ninth Annual Conference of the International Speech Communication Association*, 2008.

Odette Scharenborg, Francesco Ciannella, Shruti Palaskar, Alan Black, Florian Metze, Lucas Ondel, and Mark Hasegawa-Johnson. Building an ASR system for a low-research language through the adaptation of a high-resource language ASR system: Preliminary results, 2017.

Louis L. Scharf. *Statistical Signal Processing. Detection, Estimation and Time Series Analysis*. Addison Wesley, 1991.

Tanja Schultz and Alex Waibel. Polyphone decision tree specialization for language adaptation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2000.

Tanja Schultz, Ngoc Thang Vu, and Tim Schlippe. GlobalPhone: A multilingual text & speech database in 20 languages. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013.

Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, November 1997. doi:10.1109/78.650093.

Frank Seide, Gang Li, and Dong Yu. Conversational speech transcription using context-dependent deep neural networks. In *Proceedings of Interspeech*, pages 437–440, Florence, Italy, August 2011.

Tom Sercu, George Saon, Jia Cui, Xiaodong Cui, Bhuvana Ramabhadran, Brian Kingsbury, and Abhinav Sethy. Network architectures for multilingual speech representation learning. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017.

Khe Chai Sim and Haizhou Li. Stream-based context-sensitive phone mapping for cross-lingual speech recognition. In *Proceedings of Interspeech*, 2009.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 2014.

Pawel Swietojanski and Steve Renals. Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models. In *Proceedings of the IEEE Workshop on Spoken Language Technology*, 2014.

Pawel Swietojanski and Steve Renals. SAT-LHUC: Speaker adaptive training for learning hidden unit contributions. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016.

Pawel Swietojanski, Arnab Ghoshal, and Steve Renals. Unsupervised cross-lingual knowledge transfer in DNN-based LVCSR. In *Proceedings of the IEEE Workshop on Spoken Language Technology*, 2012.

Pawel Swietojanski, Jinyu Li, and Steve Renals. Learning hidden unit contributions for unsupervised acoustic model adaptation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2016.

Tian Tan, Yanmin Qian, Maofan Yin, Yimeng Zhuang, and Kai Yu. Cluster adaptive training for deep neural network. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2015.

Samuel Thomas, Sriram Ganapathy, and Hynek Hermansky. Multilingual MLP features for low-resource LVCSR systems. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2012.

Samuel Thomas, Michael L Seltzer, Kenneth Church, and Hynek Hermansky. Deep neural network features and semi-supervised training for low resource speech recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013.

Sibo Tong, Philip N Garner, and Hervé Bourlard. An investigation of deep neural networks for multilingual speech recognition training and adaptation. In *Proceedings of Interspeech*, 2017a.

Sibo Tong, Philip N Garner, and Hervé Bourlard. Multilingual training and cross-lingual adaptation on CTC-based acoustic model. *arXiv preprint arXiv:1711.10025*, 2017b.

Sibo Tong, Philip N Garner, and Hervé Bourlard. Cross-lingual adaptation of a CTC-based multilingual acoustic model. *Speech Communication*, 104:39–46, 2018a.

Sibo Tong, Philip N Garner, and Hervé Bourlard. Fast language adaptation using phonological information. In *Proceedings of Interspeech*, 2018b.

Sibo Tong, Philip N Garner, and Hervé Bourlard. An investigation of multilingual ASR using end-to-end LF-MMI. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2019a.

Sibo Tong, Apoorv Vyas, Philip N Garner, and Hervé Bourlard. Unbiased semi-supervised LF-MMI training using dropout. In *Proceedings of Interspeech*, 2019b.

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. *arXiv preprint arXiv:1711.01694*, 2017.

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

Karel Veselỳ, Martin Karafiát, František Grézl, Miloš Janda, and Ekaterina Egorova. The language-independent bottleneck features. In *Proceedings of the IEEE Workshop on Spoken Language Technology*, 2012.

Karel Veselỳ, Mirko Hannemann, and Lukáš Burget. Semi-supervised training of deep neural networks. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 2013.

Karel Veselỳ, Lukás Burget, and Jan Cernockỳ. Semi-supervised DNN training with word selection for ASR. In *Proceedings of Interspeech*, 2017.

Ngoc Thang Vu, David Imseng, Daniel Povey, Petr Motlicek, Tanja Schultz, and Hervé Bourlard. Multilingual deep neural network based acoustic modeling for rapid language adaptation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2014.

Apoorv Vyas, Pranay Dighe, Sibo Tong, and Hervé Bourlard. Analyzing uncertainties in speech recognition using dropout. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2019.

Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. Phoneme recognition using time-delay neural networks. *Backpropagation: Theory, Architectures and Applications*, 1995.

Shinji Watanabe and Jen-Tzung Chien. *Bayesian speech and language processing*. Cambridge University Press, 2015.

Felix Weninger, Björn Schuller, Florian Eyben, Martin Wöllmer, and Gerhard Rigoll. A broadcast news corpus for evaluation and tuning of German LVCSR systems. *arXiv preprint arXiv:1412.4616*, 2014.

Frank Wessel and Hermann Ney. Unsupervised training of acoustic models for large vocabulary continuous speech recognition. *IEEE Transactions on Speech and Audio Processing*, 2005.

Ronald J. Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, 1989. doi:10.1162/neco.1989.1.2.270.

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

Chunyang Wu and Mark J. F. Gales. Multi-basis adaptive neural network for rapid adaptation in speech recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2015.

Chunyang Wu, Penny Karanasou, and Mark J. F. Gales. Combining i-vector representation and structured neural networks for rapid adaptation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016.

W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, and G. Zweig. The Microsoft 2016 conversational speech recognition system. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5255–5259, New Orleans, LA, USA, March 2017. doi:10.1109/ICASSP.2017.7953159.

Jiangyan Yi, Hao Ni, Zhengqi Wen, Bin Liu, and Jianhua Tao. CTC regularized model adaptation for improving LSTM RNN based multi-accent mandarin speech recognition. In *Chinese Spoken Language Processing (ISCSLP), 2016 10th International Symposium on*, 2016.

George Zavaliagkos, Man-Hung Siu, Thomas Colthurst, and Jayadev Billa. Using untranscribed training data to improve performance. In *Proceedings of the International Conference on Spoken Language Processing*, 1998.

Albert Zeyer, Eugen Beck, Ralf Schlüter, and Hermann Ney. CTC in the context of generalized full-sum HMM training. In *Proceedings of Interspeech*, 2017.

Pengyuan Zhang, Yulan Liu, and Thomas Hain. Semi-supervised DNN training in meeting recognition. In *Proceedings of the IEEE Workshop on Spoken Language Technology*, 2014.

Guo-Bing Zhou, Jianxin Wu, Chen-Lin Zhang, and Zhi-Hua Zhou. Minimal gated unit for recurrent neural networks. *International Journal of Automation and Computing*, 13(3): 226–234, June 2016. doi:10.1007/s11633-016-1006-2.

Sibo Tong

PhD Student, EPFL/Idiap

+41 (766) 90 22 25
sibo.tong@idiap.ch, sibo.tong@epfl.ch
http://www.idiap.ch/~stong

Summary

A computer science engineer working on application of signal processing and machine learning to speech research.

Education

2020(expected) **Doctoral Student in Electrical Engineering**, *École Polytechnique Fédérale de Lausanne (EPFL)*, Switzerland.

2016 **Master of Science (by Research) - Computer Science**, *Shanghai Jiao Tong University (SJTU)*, China.

2013 **Bachelor of Science - Computer Science**, *Tongji University*, China.

Work Experience

May'16 – present **Research in Multilingual Speech Recognition**, *Idiap/ École Polytechnique Fédérale de Lausanne (EPFL)*, Switzerland.
Supervisors: Prof. Hervé Bourlard, Dr. Philip N. Garner

- Working on Bayesian recurrent unit for neural networks with probabilistic explanations
- Working on uncertainty estimation in automatic speech recognition (ASR) and semi-supervised training using dropout
- Exploited phoneme-based end-to-end speech recognition frameworks for fast cross-lingual adaptation
- Explored multilingual training architectures and language adaptative training techniques to improve multilingual ASR

Oct'19 – Jan'20 **Research Intern**, *Amazon*, Germany.
Supervisor: Dr. Simon Wiesler

- Designed and implemented multilingual ASR systems based on Recurrent Neural Network Transducer (RNN-T) using one hundred thousand hours data
- Investigated several language adaptive training approaches which improves the model performance.

Sep'13 – Mar'16 **Research Assistant**, *SJTU Speech Lab*, China.
Thesis Title: Voice activity detection with deep learning
Supervisor: Prof. Kai Yu

- Explored various Deep Neural Network (DNN) architectures for voice activity detection (VAD)
- Investigated the effect of voice activity detection on speech recognition systems and built an integrated VAD evaluation framework taking into account various boundary effect

Sep'15 – Nov'15 **Research Intern**, *Toshiba R&D center*, Japan.
Supervisors: Dr. Masanobu Nakamura, Dr. Masami Akamine

- Implemented VAD frameworks based on various recurrent architectures
- Explored smoothing techniques to post-process neural network outputs

Dec'14 – Mar'15 **Research Intern**, *AiSpeech*, China.
Supervisor: Prof. Kai Yu

- Developed speech enhancement framework and noise-aware training for DNN-based VAD system to improve the robustness

Awards and Academic Achievements

- Best Student Paper Award at the 10^{th} International Symposium on Chinese Spoken Language Processing
- Showa Denko Scholarship Award at SJTU
- Academic Scholarship Award at SJTU

Journal Publications

- P.N. Garner, **S. Tong**, *"A Bayesian Approach to Recurrence in Neural Networks"*, in **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 2020. (accepted)
- **S. Tong**, P.N. Garner, and H. Bourlard, *"Cross-lingual adaptation of a CTC-based multilingual acoustic model"*, in **Speech Communication**, 2018.

Conference and Other Publications

- **S. Tong**, A. Vyas, P. Motlicek, and H. Bourlard, *"Unbiased semi-supervised LF-MMI training using dropout"*, **Interspeech 2019**.
- **S. Tong**, P. N. Garner, and H. Bourlard, *"An investigation of multilingual ASR using end-to-end LF-MMI"*, in **ICASSP 2019**.
- A. Vyas, P. Dighe, **S. Tong** and H. Bourlard, *"Analyzing uncertainties in speech recognition using dropout"*, in **ICASSP 2019**.
- **S. Tong**, P. N. Garner, and H. Bourlard, *"Fast language adaptation using phonological information"*, in **Interspeech 2018**.
- M. Cernak and **S. Tong**, *"Nasal speech sounds detection using connectionist temporal classification"*, in **ICASSP 2018**.
- **S. Tong**, P. N. Garner, and H. Bourlard, *"An investigation of deep neural networks for multilingual speech recognition training and adaptation"*, in **Interspeech 2017**.
- **S. Tong**, H. Gu and K. Yu, *"A comparative study of robustness of deep learning approaches for VAD"*, in **ICASSP 2016**.
- J. Lai, B. Chen, T. Tan, **S. Tong**, and K. Yu, *"Phone-aware LSTM-RNN for voice conversion"*, in **ICSP 2016**.
- Y. Zhuang, **S. Tong**, M. Yin, Y. Qian, and K. Yu, *"Multi-task joint-learning for robust voice activity detection"*, in **ISCSLP 2016**.
- **S. Tong**, N. Chen, Y. Qian, and K. Yu, *"Evaluating VAD for automatic speech recognition"*, in **ICSP 2014**.