MULTI-LANGUAGE DATASETS FOR SPEECH RECOGNITION BASED ON THE END-TO-END FRAMEWORK

A THESIS SUBMITTED TO AUCKLAND UNIVERSITY OF TECHNOLOGY
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF COMPUTER AND INFORMATION SCIENCES

Supervisor

Wei Qi Yan

3 May 2021

By

Sendong Liang

School of Engineering, Computer & Mathematical Sciences

Abstract

Speech recognition is an important research field in natural language processing. With the vigorous development of Artificial Intelligence (AI), especially deep neural networks in recent years, Automatic Speech Recognition (ASR) has begun to use deep neural networks to process speech recognition. The deep neural network has been directly fused to form HMM-DNN based on the HMM-GMM model. An end-to-end framework was created directly in machine translation. The end-to-end framework for speech recognition does not require complicated alignment and construction of the pronunciation dictionary, which shows a promising prospect.

In this thesis, the end-to-end framework for speech recognition is probed with multilanguage datasets. The focus of this thesis is on the end-to-end framework. Our objective is to improve the performance of the CTC/Attention model. To compare speech recognition performance in different languages, we designed and built three small datasets, including Chinese, English and Code-Switch. We compare the performance of the hybrid CTC/Attention model in multiple languages environment. Throughout our experiments, we explore that the end-to-end framework of the CTC/Attention model achieves similar or better performance with the HMM-DNN model in a single language and Code-Switch speaking environment. Moreover, speech recognition in different languages is compared in this thesis.

Keywords: Speech recognition, End-to-end framework, Attention model, CTC model, Code-Switch, Low-resource datasets

Contents

List of Tables

List of Figures

Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the qualification of any other degree or diploma of a university or other institution of higher learning.

Signature of student

Acknowledgements

First of all, I am extremely thankful to my supervisor Wei Qi Yan, for his incisive advice, valuable support, and continuous patience during my MCIS study. His knowledgeable capability and great experience have encouraged and inspired me in all the time of this MCIS project. Without his invaluable support, it would be impossible for me to complete my Master study.

I would like to thank Mr Chong Wang for sharing his experience about the utilization of Latex to draft the thesis. I say thanks to Mr Jason Zhang for his hardware support of my research experiments. I do appreciate Mr Ke Wang for his technical support on Kaldi experiment. I deeply thank Mr Guangpeng Jiang for his support of utilization of the iFlytek InterPhonic5.0 toolkit. I am very pleased to say thanks to Mr Chen Qiu and Mr Longru Zhu for their support of data collection. I oblige all the staffing members of the student hub at the Auckland University of Technology.

Moreover, I would like to express my gratitude to my parents for their financial support and kind encouragement to my MCIS study. I am very happy to say thanks to my girlfriend for her understandings, kind encouragement and supports for all through my MCIS study. I would like to thank Dr Zhenyu Sun for his professional medical and psychological counselling to help me relieve the anxiety due to the COVID-19 epidemic.

I hope that the COVID-19 epidemic will be ended soon. I hope the world will be peaceful, and everyone will be safe, healthy, and happy.

Chapter 1

Introduction

This chapter contains five parts. In the first part, we introduce the background and motivation of this research. In the second part, we will ask the research questions related to the thesis. In the third part, our contributions of this thesis will be addressed. The goals of this research project are described in the fourth part. The last part is related to detail the structure of this thesis.

1.1 Background and Motivation

Spoken language is essential to modern human cultures. Speech is a way of communications between people through languages. In the past decades, with the popularity of intelligent devices and the development of applications, such as intelligent voice assistants, intelligent speakers, intelligent home appliances, began to enter people's daily life. Therefore, the requirements of human-computer interaction are more and more in human life. When people communicate with the intelligent devices through languages, automatic speech recognition(ASR) will take its main role. ASR aims to convert the audio of conversation content into corresponding text, the text will be further processed to achieve human-computer interaction, such as voice input and voice translation. Speech recognition plays a primary role in human-computer interaction, so speech recognition research has essential academic value and application value.

Speech recognition refers to the conversion from audio to text. In the early stages of the research work, since it was impossible to directly model the audio-to-text conversion process, Bayes' theorem was applied to convert speech recognition problems into given text so as to calculate the probability of corresponding audio features multiplied by the prior probability of that text. Since the prior probability of audio does not affect the recognition result, this kind of probabilities will be ignored in the calculation. The assumption of independence further decomposes the probability of finding corresponding audio feature sequences for a given text into the probability of generating corresponding audio feature sequences for the HMM state sequence and text, multiplied by the probability of state sequences for the text condition.

With the continuous development of HMM, speech recognition has transited from isolated word research to a small system of speech recognition to a large vocabulary continuous system (Woodland, Odell, Valtchev & Young, 1994). The HTK speech recognition tool developed by the P.C.woodland team at the University of Cambridge

contributes significantly to the HMM-based discriminatory training for HMM and the phoneme duration model (Woodland & Young, 1993). The speech recognition framework based on the HMM model shows excellent performance and reliable stability over the next few years. It became the mainstream speech recognition model. These frameworks are mainly composed of acoustic models, language models, and lexicons. Nowadays, it is also widely employed in industry.

The deep neural network has been employed to the acoustic modelling to form the HMM-DNN speech recognition framework due to the development of deep neural network(Maas et al., 2017). The objective function of the DNN-based acoustic model is the probability of getting HMM state given a sequence of audio features, which is the opposite of the HMM-GMM acoustic model. With more and more research studies on the deep learning model, the generalization ability of built acoustic model becomes stronger and stronger. A classification network was directly created from the HMM-DNN acoustic model (Georgescu, Cucu & Burileanu, 2019). The output of the network is no longer the states of HMM, but the sequence of text label called tokens, such as the alphabetic label or subword label of English, or a single Chinese character label of Chinese, which is the original idea of end-to-end speech recognition. This approach eliminates the need for a pronunciation dictionary and lowers the threshold for building a language recognition system.

Due to the continuous development of deep learning, the end-to-end speech recognition frameworks have shown exemplary performance in high-resource languages. However, it is hard to perform well in low-resource datasets for speech recognition, such as Chinese-English Code-Switch environment (C. H. Wu, Shen & Yang, 2014). Because the deep neural network is a data-driven model, it is challenging to train an excellent end-to-end speech recognition framework with a small amount of data. In this thesis, we optimize the end-to-end speech recognition framework to perform well in encoding and decoding, making them perform better with a small number of training

data, thereby improving the recognition accuracy with less training corpus data.

Considered more applications, our daily communications often involve a mix of languages, academically known as Code-Switch model. For example, in New Zealand, a lot of Chinese-mixed-English words while speaking Chinese. These situations are one of the critical challenges faced by speech recognition technology. The main technical difficulties include the phenomenon of non-native accents formed by the influence of the subject language looks often, the difference of phoneme composition between different languages brings great difficulties to the modelling of mixed acoustics, and the data of labeled mixed speech training is exceptionally scarce.

The traditional phonetic framework is based on the basic modelling unit of different language recognition. The linguistic information is different, for example, Chinese phonetic consonants and English phonemes. This technical framework relies heavily on the specific linguistic knowledge and is difficult to be expanded to multi-language recognition. The end-to-end frameworks apply a unified network for modelling, requiring less manual dictionary editing, and depending more on data than linguistic information. Accordingly, we have a great interest in using end-to-end thinking to solve this kind of speech recognition issue.

For example, Chinese-English-mixed speech is a small corpus compared to a single language, and experiments show that the performance of the end-to-end Chinese-English speech recognition framework is close to that based on HMM-DNN(Shi, Feng & Xie, 2020), making it possible to deploy the end-to-end framework for the online system for Chinese-to-English mixed speech recognition in the future(Shan et al., 2019). This work also indirectly demonstrates that the end-to-end framework has great potentiality not only for speech with a large number of labeled corpus but also for speech with a small number of labeled dataset. Accordingly, the research work for speech recognition based on the end-to-end framework of a small number of labeled corpus has particular values in bother academic research and industrial applications.

1.2 Research Questions

In Section 1.1, we are particularly interested in the end-to-end framework for small labeled datasets and multi-language speech recognition. Hence the purpose of this thesis is to investigate whether the end-to-end framework in speech recognition performs better on small label datasets than traditional speech recognition systems in multi-language environments. Firstly, we create a speech recognition system based on HMM-DNN for the purpose of comparisons. Secondly, the system with an end-to-end framework is the main objective of our experiments. The focus of this thesis is on the end-to-end framework of CTC and attention hybrid model, which compares speech recognition tasks of English, Chinese, and Code-Switch speech with traditional speech recognition. An end-to-end framework for speech recognition with better performance in small labeled datasets and multi-language speech recognition is obtained through experiments. Therefore, the main questions of this thesis are as follows:

(1) What is the speech recognition performance of the end-to-end framework on small labeled datasets?

(2) What are the differences in speech recognition performance on multi-language datasets?

(3) Does the end-to-end framework with a hybrid CTC/Attention model have better speech recognition performance than traditional speech recognition models on small labeled multi-language datasets?

To sum up, the core idea of this thesis is to apply the proposed end-to-end framework to improve the accuracy of speech recognition based on small labeled datasets. The traditional HMM-GMM model combines with deep learning to form a speech recognition framework used in practical applications. Therefore, in this thesis, we take HMM-DNN as the comparison object and employ the CTC/Attention hybrid model in

the end-to-end framework to find out whether the recognition accuracy of the end-to-end framework based on small labeled datasets performs better than the HMM-DNN model. Throughout experiments based on our low-resource datasets, we explore how to optimize such a model for better performance in Code-Switch environment or small labeled datasets.

1.3 Research Contributions

The focus of this thesis is on speech recognition of small labeled datasets and multi-language environment in the end-to-end framework. On the basis of related work, four experiments will be implemented with the datasets and multiple models. The corresponding results will also be analysed. The contributions of this thesis include:

(1) Investigating the performance of a speech recognition system on small labelled datasets;

(2) Exploring an end-to-end framework for speech recognition on multi-language datasets;

(3) Comparing the end-to-end speech recognition framework based on CTC/Attention hybrid model with the traditional speech recognition model on small labeled multi-language datasets.

Moreover, the experimental results will be compared and analysed to evaluate the main objective and exploration of this thesis.

1.4 Objectives of This Thesis

Firstly, data preparation is introduced and conducted. Meanwhile, the acoustic modelling units of multiple languages and various evaluation methods are also depicted,

in which the datasets contain Mandarin dataset, English dataset, and mixed Mandarin-English dataset. Secondly, HMM-DNN based speech recognition model is introduced for the purpose of comparisons. The objectives of this thesis include a comparison model based on HMM-DNN, the design and implementation of our experiments based on the end-to-end framework with hybrid CTC/Attention model. Finally, in this thesis, we will analyse and discuss the experimental results and evaluate the outcomes with small labeled datasets and Code-Switch model in speech recognition.

1.5 Outline of This Thesis

The outline of this thesis is shown as the following description:

In Chapter 2, our focus is on literature review based on the previous related work. After the basic review of ASR development history, acoustic models, language models, feature extraction and various models will be detailed and investigated. Moreover, the end-to-end framework in speech recognition will also be explored.

In Chapter 3, we will conduct the experiments including data preparation and design of four experiments for speech recognition. The corresponding results of each experiment will also be depicted.

In Chapter 4, we will implement the analysis an discussion of the experimental outcomes. The limitation of the proposed model will also be discussed.

In Chapter 5, we will provide the conclusion as well as our future work.

Chapter 2

Literature Review

This chapter contains seven parts. In the first part, we introduce the development history of ASR. In the second part, preprocessing and feature extraction are discussed. In the third part, we examine the language models of ASR from the perspectives of the n-gram and neural network language model, in which the neural network language model is divided into recurrent and feedforward neural networks. In the fourth part, we describe the acoustic model in the ASR system. The GMM, HMM, DNN, CNN and LSTM models are also investigated. In the final part, we explore the end-to-end speech recognition framework, including attention encoder-decoder, RNN transducer, and CTC.

2.1 ASR

There are three main development steps of speech recognition models. The first step is the rule-based model, such as Shoebox and Harpy created by IBM and CMU in 1962 and 1976, respectively (Jason & Kumar, 2020). The second step is the statistical-based models such as Large Vocabulary Continuous Speech Recognition (LVCSR) and Hidden Markov Model (HMM). HMM-Gaussian Mixture Model (GMM) had been the dominant framework in the field of speech recognition until the application of deep learning techniques in 2006 (D. Wang, Wang & Lv, 2019). The third step is Deep Learning-based models such as Deep Neural Networks (DNN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM)-RNN, and bidirectional LSTM (BiLSTM). Considered LSTM-RNNs are sensitive to the static data, which may lead to delays with respect to features arise, BiLSTM appears as a special architecture operating the input sequence in both directions (Liu, Chen, Hu, Tang & Zou, 2019).

Bayes' theorem was early applied to convert speech recognition problems into given text conditions to calculate the probability of corresponding audio features multiplied by the prior probability of that text. Speech recognition systems have transited from isolated word research to the small speech recognition system and the LVCSR on the basis of HMM (Woodland et al., 1994). The HTK speech recognition tool contributes significantly to the HMM-based discriminatory training for HMM and the phoneme duration model (Woodland & Young, 1993). The speech recognition framework based on the HMM model performed well over the next few years, which became the mainstream speech recognition model.

However, there are three drawbacks to the utilization of this framework. Firstly, the framework is optimized separately by each module, which is prone to local optimization problems. Secondly, this framework requires a dictionary, and the construction of

a dictionary requires linguistic knowledge. There are many languages in the world. If a developer creates a speech recognition system for a language, the linguistic knowledge is needed to be understood. Finally, since each module has been optimized separately and then linked up hierarchically, this process is very complex and prone to loss of performance between models, making the whole model not optimal.

In 2006, an unsupervised method was employed to pre-train the Deep Belief Network (DBN), which solved the problem that gradient descent is sensitive to the initial value (G. E. Hinton, Osindero & Teh, 2006). Deep neural network has been applied to the acoustic modelling to form the HMM-DNN framework in speech recognition (Maas et al., 2017). The objective function of DNN-based acoustic models is the probability of getting HMM state given a sequence of audio features. In 2009, DNN was applied to the TIMIT phoneme recognition task and achieved good recognition performance (Mohamed, Dahl & Hinton, 2009). In 2012, CNN was applied to the LVCSR system, which normalized the data to obtain higher speech recognition performance (Abdel-Hamid, Mohamed, Jiang & Penn, 2012).

In 2014, the LSTM model is applied in the LVCSR system to solve the gradient disappearance or gradient explosion issues of traditional RNN model (Sak, Senior & Beaufays, 2014; Sak, Senior, Rao & Beaufays, 2015).

Moreover, an end-to-end speech recognition framework based on RNN and Weighted Finite State Transducer (WFST) decoding technology was proposed in 2015. This end-to-end framework directly utilises analogue signals as input, which improves the recognition rate of the system (Miao, Gowayyed & Metze, 2015). In 2017, a new method of end-to-end speech recognition was proposed, which employs the CTC model in a multi-task learning framework to improve the robustness of the system and achieve quick converge, thereby alleviating the problem of data alignment (Kim, Hori & Watanabe, 2017).

However, most of the previous research work on speech recognition focused on

performing better in recognition of a single language or a single task. The comparison of speech recognition between different languages and the work of speech recognition in a multilingual or code-switch environment are relatively rare.

2.2 Preprocessing and Feature Extraction

Preprocessing is the basic step of voice signals before feature extraction. It eliminates the effects of aliasing and high-order harmonic distortion on the quality of the voice signal which is caused by the speaker's vocal organ and the voice-collect equipment (Pacheco & Seara, 2008). This step ensures the high quality of the acoustic signal, and provide it for the feature extraction (Ibrahim, Odiketa & Ibiyemi, 2017). Preprocessing includes spectral enhancement (Reindl, Zheng, Meier, Schwarz & Kellermann, 2012), medium-duration analysis window (Mitra, Franco, Graciarena & Vergyri, 2014) and voice activity detection (Mousazadeh & Cohen, 2013).

Feature extraction means to grasp useful features from quasi-stationary speech signals relying on short-term amplitude spectrum (Gupta & Gupta, 2016). Feature refers to a compact representation of raw input speech signals. Feature vectors mean the quality of classifier in training and the knowledge in testing, which are computed from an estimated spectrum (Kadyan, Mantri & Aggarwal, 2020). Digitally recorded speech refers to the discrete-number representation corresponds to the changes of air pressure, which are produced during people talking (Mansikkaniemi, 2010).

In addition to the vital phonemes information, various excess information existing in the raw speech signal, such as gender and mood, is necessary to further process the speech signal and extract the essential features for recognizing phonemes (Mansikkaniemi, 2010). The subword pronunciation verification in ASR provides adequate performance from speech utterances taken from the impaired speech domain (Saz et al., 2009).

As one of the powerful feature extraction methods in this step, Mel-Frequency Cepstral Coefficient (MFCC) is easy to compute and extract the features. The frequency bands carry significant information while analysing speech.

Considering human ears is sensitive to the changes in pitch at low frequencies, and the Mel frequency enhances the low-frequency details of the acoustic signal, the phonetic differences will be more narrow and more comprehensive at lower frequencies (Picone, 1993). Accordingly, the relationship between the Mel frequency and the actual Hz frequency can is close to the actual human auditory system, leading to better-characterized performance of the speech signal.

MFCC performs better than linear frequency cepstrum coefficients (LFCC) by providing perceptually relevant information about the short-time speech spectrum (Davis & Mermelstein, 1980).

As two traditional feature extraction approaches, MFCC and GFCC face high variance with leakage of spectral information and low-performance mismatch in training and testing when classifying large numbers of the acoustic feature vector. In order to resolve these issues, the PCA-based multiwindowing and modelling classifier on the basis of differential-evolution HMM perform better in word recognition compared with the traditional and fused feature extraction systems (Kadyan et al., 2020).

2.3 Language Models

Language models are regarded as a critical component in building ASR systems, which are widely applied to natural language processing, such as LVCSR and statistical machine translation (Schwenk, Bougares & Barrault, 2014). Language modelling aims to model text and estimate the probability of the token sequence given context leading to the importance of dealing with sequential data prediction problem while creating language models for speech recognition (Mikolov, Karafiát, Burget, Černockỳ &

Khudanpur, 2010).

The language model utilizes the probability of collocation between words, namely, the probability of occurrence is different due to the deep nature in the context. Since speech recognition imports the two fields of audio and text, a language model is needed from the text perspective.

There are two main approaches to train language models. The first approach is the conventional count models, which depends on the relative frequencies of n-gram counts and smoothing methods such as backing-off to shorter contexts and interpolation (S. F. Chen & Goodman, 1999). The second approach is the Neural Network-based Language Models (NNLM), including the Feedforward Language Model (FFLM) and Recurrent Network Language Model (RNNLM).

2.3.1 n-gram Language Model

Before the development of neural networks, statistical modelling methods could not effectively model the dependence of long sentences. This problem has been simplified through n-gram modelling (Bellegarda, 1998). n-gram modelling obtains generalization by concatenating the short overlapping sequences that appeared in the training dataset (Bengio, Ducharme, Vincent & Janvin, 2003).

More specifically, the n-gram model refers to solve the probability of the n-th word if the length of the current word sequence is reduced to n, with the given current $n-1$ words. Usually, n is a relatively low model order due to the rapid increase of the size of language models with growing n (Siivola & Pellom, 2005), which enables FFLM to be directly integrated into the traditional decoding of speech recognition (Schwenk & Gauvain, 2002).

The n-gram utilised in this thesis is 3-gram, namely, if predict the current word w_n, it only depends on the two preceding words w_{n-2} and w_{n-1}. For count-based

language models, using relative frequencies is a simple way to calculate the Maximum Likelihood (ML) estimate as shown in Eq. (2.1).

$$P_{ML}\left(w_n \mid w_{n-2:n-1}\right) = \frac{C\left(w_{n-2:n}\right)}{C\left(w_{n-2:n-1}\right)} \tag{2.1}$$

where $C(\cdot)$ represents the number of sequences obtained from the training set, $C(w_{n-2:n})$ stands for the number of sequences composed of the three words w_{n-2}, w_{n-1}, and n, $C(w_{n-2:n-1})$ refers to the number of sequences composed of the two words w_{n-2} and w_{n-1}.

The problem of data sparseness is a considerable issue in statistical language model modelling. Assume that there are $20,000$ words, there are $20,000^3$ of words combined with a length of 3 in 3-gram. However, it is impossible to have so many word order combinations in the training set, since some of them that appear in the test set do not turn up in the training set. With the combinations of word order, such as $w_{n-2:n}$ (namely, $w_{n-2}w_{n-1}w_n$) never appears in the training set, the speech recognition system will not have a chance to output correctly with a result of $P_{ML}(w_n|w_{n-2:n-1}) = 0$. Accordingly, smoothing methods are usually applied to avoid zero probability and alleviate the data sparseness issue.

The term smoothing refers to the method to adjust the maximum likelihood estimate of probabilities to produce more uniform distributions and more accurate probabilities (S. F. Chen & Goodman, 1999), which is able to generally prevent zero probabilities.

Based on previous work (Mikolov, Deoras, Kombrink, Burget & Černockỳ, 2011), linear interpolation based on the trained models of various n-gram orders is a simple smoothing method. Accordingly, for 3-gram, the final conditional probability of a given history can be estimated by weighing and combining the trigram, bigram, and unigram counts through the parameter λ_n:

$$\hat{P}(w_n \mid w_{n-1:n-2}) = \lambda_1 P_{ML}(w_n \mid w_{n-1:n-2}) + \lambda_2 P_{ML}(w_n \mid w_{n-1}) + \lambda_3 P_{ML}(w_n) \quad (2.2)$$

where $P_{ML}(w_n \mid w_{n-1:n-2})$, $P_{ML}(w_n \mid w_{n-1})$, and $P_{ML}(w_n)$ are calculated through Eq. (2.1). The weights λ_n of each model should be non-negative and constrained to $\sum_{n=1}^{3} \lambda_n = 1$, which makes $\hat{P}(w_n \mid w_{n-1:n-2})$ to be a valid probability result.

2.3.2 Neural Network Model

Compared with the traditional n-gram language models, neural network-based language models have better performance from language modelling benchmarks (Raju, Filimonov, Tiwari, Lan & Rastrow, 2019). More specifically, based on previous work (Bengio et al., 2003), the feedforward language model (FFLM) and the recurrent network language model (RNNLM) can solve the curse of dimensionality by learning a distributed representation for getting the exponential number of each training sentence in the semantic contexts and learning the probability function for word sequences simultaneously.

Feedforward Neural Network Language Model

The parameters of the feedforward neural network language model (FFLM) are used to present word feature vectors, and the negative log loss is minimized by jointly adjusting the word feature vectors and other parameters (equivalent to minimizing perplexity). Similar to n-gram, FFLM depends on previous words. The output probability of the n-gram FFLM depends on the word feature vector of the preceding word. Under the condition of a single sample (namely, the batch size equals to 1), the feedforward neural network structure of 4-gram is as follows:

w_{t-3}, w_{t-2}, and w_{t-1} are the input of the neural network. After one-hot encoding, each word enters an embedding layer (namely an embedding matrix) as $C \in R^{|v| \times m}$. $|v|$ is the row of the matrix which refers to the size of the vocabulary. The column of the matrix is m, which is the degree of embedding vector of each word. $C(w_{t-3})$, $C(w_{t-2})$, and $C(w_{t-1})$ represent the vector after embedding.

The activation vector **x** of the word integral layer is formed by putting these three embedding vectors together.

$$\mathbf{x} = (C(w_{t-3}), C(w_{t-2}), C(w_{t-1})), \mathbf{x} \in R^{3m} \tag{2.3}$$

If the number of neurons in the hidden layer is h, then the probability of the current w_t is y, which is the logarithmic probability without normalization.

$$y = xW^T + tanh(d + xH^T)U^T + b \tag{2.4}$$

where parameters b and d are bias vectors, b is the output layer bias of size $b \in R^{|V|}$, d is the hidden layer bias of size $d \in R^{|h|}$. x refers to an input vector with a length of 3m. W, U, and H are matrices, and T is the transpose of the matrix. $tanh$ is the activation function. The matrix $W \in R^{V \times 3m}$ is the weight matrix directly connected from the embedding layer to the output layer. The matrix $U \in R^{|V| \times h}$ is the weight matrix from the hidden layer to the output layer. The matrix $H \in R^{h \times 3m}$ is the weight matrix from the embedding layer to the hidden layer.

The normalized probability P of the current word w_t is calculated from the input vector y through the softmax activation function.

$$P\left(w_t \mid w_{t-1}, w_{t-2}, w_{t-3}\right) = \frac{\exp y_{w_t}}{\sum_i \exp y_i} \tag{2.5}$$

Different from the statistical-based n-gram language model, the parameters of the n-gram FFLM model increase linearly with n, rather than exponentially.

Language Model Using Recurrent Neural Networks

Feedforward neural depends on the preceding word, which cannot learn long sentence dependent information. With the development of deep learning, recurrent neural networks (RNNs) emerged, which address the issue of long sequence dependence. RNN is defined as a type of artificial neural network that classify, cluster, and make predictions about various data, such as text, language, speech, video, time-series, and DNA genomes data (Manaswi, Manaswi & John, 2018).

Compared with traditional feedforward neural networks such as multilayer perceptions using static classifiers only by considering fixed-size input windows irrespective of surrounding context, RNN is more effective and suitable to transcribe connected time-series such as speech transcription because of its hidden network layers (Eyben, Wöllmer, Schuller & Graves, 2009). Different from the feedforward network using fixed-length context, RNN does not take use of a limited size of context, which contains cache models that can encode temporal information implicitly for contexts with arbitrary lengths (Mikolov et al., 2010). The recurrent connections allow information to cycle inside networks for a long time adapting to the past inputs (Boden, 2002). The utilization of a continuous vector space for word representation and deep learning methods better represents the relationship between words. Accordingly, compared to feedforward neural networks, recurrent neural networks reflect the contextual dependency spanning over a fixed number of predecessor words (Sundermeyer et al., 2013). In order to compare FFLM and RNNLM more clearly, equation (2.4) is simplified as

$$y = tanh(d + xH^T)U^T + b \tag{2.6}$$

where $tanh(d + xH^T)$ is noted as M,

$$y = MU^T + b \tag{2.7}$$

The neural networks predict the current word depending on the word at the previous moment. The current output of the output layer is determined by the joint of the saved output M of the hidden layer at the previous moment and the current hidden layer. The current output of the hidden layer must depend on the output of the hidden layer at the previous moment, and the output of the hidden layer at the first previous moment ought to depend on the output of the hidden layer at the second previous moments. Accordingly, it comes up with a recurrent calculation with infinite historical information in theory leading to RRN:

$$M_t = \tanh(d_n + xH_t^T + M_{t-1}W_{hh}) \tag{2.8}$$

where d_n is the bias vector of the hidden layer, $x \in R^{1\times 3m}$ is the embedding feature vector of the preceding word. $H_t^T \in R^{m\times h}$ is the transposition of the weight matrix from the embedding layer to the hidden layer at the current moment. $M_{t-1} \in R^{1\times h}$ is the output vector of the hidden layer at the previous moment. $W_{hh} \in R^{h\times h}$ refers to the weight of hidden layers, which represents how much of the current output of the hidden layer depending on the previous output of the hidden layer.

As for the application of the RNN language model in the bilingual Code-Switch environment, Yilmaz, Heuvel and van Leeuwen (2018) enhanced the recognition performance in a Frisian-Dutch mixed language recognition by creating Code-Switch text using RNN language models, adding transcribed Code-Switch speech data, and translating Dutch text extracted from the transcriptions of large Dutch speech corpora.

Furthermore, there are two main methods based on previous studies on the Code-Switch speech recognition (J. Y. Chan, Cao, Ching & Lee, 2009). The first method aims to divide the input speech signals into homogeneous language segments through language boundary detection (LBD) so as to determine language identity of each segment based on language-specific phonological and acoustic properties through a corresponding monolingual speech recognizer (C.-H. Wu, Chiu, Shia & Lin, 2005). The focus of second method is on processing multilingual utterances together by implementing a cross-lingual ASR system, in which the design of the acoustic models, language models as well as lexicon should be prepared as multilingual lexicon, such as Mandarin-Taiwanese (Lyu, Lyu, Chiang & Hsu, 2006), English-Chinese (You et al., 2004) and Frisian-Dutch (Yilmaz et al., 2018).

2.4 Acoustic Models

The most likely word sequence based on previous acoustic features follows the step of speech input, feature extraction, decoding network, and text output, which are four components of a speech recognition system.The acoustic model estimates the likelihood of the word sequence based on the observed acoustic features (Cui et al., 2020). Acoustic models estimate the uttered probability of a particular phoneme in a recorded audio segment trained based on pre-recorded speech (Mansikkaniemi, 2010).

Speech recognition models contain the parameterisation of the input noise signal as well as training and testing of the features through classification techniques (Waris & Aggarwal, 2018). Large Vocabulary Continuous Speech Recognition (LVCSR) systems apply multiple decoding and restoring passes containing several speaker adaptation passes. The performance is improved by using “cross-pollinating” diverse acoustic models with multiple design parameters (e.g., input features, acoustic modelling paradigm, phonetic context, discriminative training criterion) through cross-adaptation

and system combinations (Saon & Chien, 2012). The utilization of acoustic features derived using Multi-Layer Perceptrons (MLPs) for ASR system has been shown to perform well, which expands the traditional short-term spectral-based feature vector such as perceptual linear prediction (PLP) (J. Park, Diehl, Gales, Tomalin & Woodland, 2011). If generate and utilise these MLP features, rapid system training, system adaptation, and system combination are three considerable aspects (J. Park et al., 2011).

The distributed training of acoustic models, DNN acoustic models have unique characteristics such as shallow, light computational load, and heavy communication load. The centralized and decentralized training from the perspectives of synchronous PSGD and Asynchronous PSGD separately has been conducted (Cui et al., 2020).

For the low-resource speech recongition, the acoustic model requires various resources such as dictionaries and phonetic questions, which may be unavailable under the conditions such as low-resource datasets, which leads to restrict or delay the deployment of ASR (Miao et al., 2015).

2.5 Gaussian Mixture Models (GMM) and Hidden Markov Model (HMM)

Hidden Markov Model (HMM) refers to a recognition method based on statistics, including extracting acoustic features, acoustic model, language model, decoder, and other postprocessing fused technologies (W. Chan, Jaitly, Le & Vinyals, 2016). HMM normalizes the temporal variability, and GMM computes the emission probabilities of HMM states (Miao et al., 2015).

By considering the non-stationary process with the range of frequency and time

of the speech signals, HMM models have poor robustness performance since they focus on the time-dimension analysis (Y. Li, Pi & Xiao, 2018). Moreover, the context-dependent state model performs higher accuracy compared to context-independent states both in HMM-GMM systems and HMM-DNN hybrid systems (Senior, Heigold, Bacchiani & Liao, 2014). A theoretical framework of maximum a posteriori adaptation of the hidden activation function parameters is proposed in context-dependent-HMM-DNNs to reduce the mismatch between training and testing (Z. Huang, Siniscalchi & Lee, 2016).

2.6 Deep Neural Network (DNN)

The acoustic models are based on GMM-HMM for a long time until the development of deep learning. Compared with GMMs, the disadvantage of DNNs is the difficulty to make good use of large cluster machines to train them on massive datasets and find better ways of parallelizing the fine-tuning of DNNs (G. Hinton et al., 2012). In HMM-DNN model, DNNs are employed to classify speech frames into clustered context-dependent states as acoustic models that improve the performance of speech recognition (Miao et al., 2015). HMM-DNN-based acoustic model has gradually become mainstream in speech recognition due to its good recognition performance (LeCun, Bengio & Hinton, 2015).

However, the training of DNNs still depends on GMM models to obtain the initial frame-level label, which increases the complexity of speech recognition. Building GMM models contains multiple stages, such as context-dependent states and each stage involves various feature processing techniques (Miao et al., 2015).

Accordingly, a GMM-free bootstrapping DNN acoustic model was explored to reduce the complexity of the ASR system (Senior et al., 2014). The context-dependent trees are created with DNN alignments, which is well-matched to its features and DNN

model. In 2014, another GMM-free online-training algorithm of a context-dependent DNN model was proposed, optimising the DNN parameters and alignment by using flat starting a model from scratch to avoid GMM acoustic model. Compared with a context-independent GMM bootstrapped system, the proposed algorithm reduces the recognition error rate by 24% and decreased the rate by 16% compared to a context-dependent GMM bootstrapped system (Bacchiani, Senior & Heigold, 2014).

In the low-resource LVCSR environment, the utilization of DNN when extracting the speech features can solve the problem of insufficient training data (Thomas, Seltzer, Church & Hermansky, 2013). The combination of DNN bottleneck features and MFCC for speech recognition under low noise conditions can achieve better recognition performance (Imseng, Motlicek, Garner & Bourlard, 2013).

Moreover, DCNN and DFCNN become popular frameworks in speech recognition because if the convolution layer numbers of CNN become more extensive, the recognition performance of CNN will be better (Y. Li et al., 2018).

2.6.1 Convolutional Neural Network (CNN)

CNN refers to an advanced method with advanced features such as weight sharing, local filters, and pooling that normalizes speaker variance using local filters in the convolution layer (Waris & Aggarwal, 2018). Local filters, weight sharing and activation function, and pooling are the three main CNN properties (Passricha & Aggarwal, 2020).

Different from the standard neural network using only fully connected layers, CNN is an advanced version with a particular structure of the standard neural network, which is made from numerous feature extraction levels, and each degree includes a convoluted layer, a nonlinear transformation layer, and a pooling layer (Newatia & Aggarwal, 2018). Moreover, CNN-TDNNF contains two CNN layers followed by the

factorized time-delay neural networks architecture, useful in low-resource conditions for Code-Switch environment (Biswas, De Wet, van der Westhuizen & Niesler, 2020).

2.6.2 Long Short-Term Memory (LSTM) Network

It is challenging to train standard RNNs to solve long-term temporal dependencies requirement because of the gradient varnishing while learning long-term dependencies (Bengio, Simard & Frasconi, 1994), namely, the gradient of the loss function decays exponentially with time (Manaswi et al., 2018). In order to solve this problem, the long short-term memory network (LSTM) as a variant of RNN, using a set of gates to control when information enters memory, which can not only solve the issues of vanishing and exploding gradients but also learn the dependence of long sentences by storing information over a long period (Manaswi et al., 2018).

There are a number of recurrently connected memory blocks in the LSTM hidden layer, each of which contains at least one recurrently connected memory cells and three multiplicative gate units: The forget gate, input gate, output gate ("Learning precise timing with LSTM recurrent networks", n.d.).

Based on previous studies on long-term or short-term dependency of sequential data (Eyben et al., 2009; Sak et al., 2014), the following equations show how the LSTM network calculates the network unit activation.

$$f_t = sigmoid\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right) \tag{2.9}$$

$$i_t = sigmoid\left(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i\right) \tag{2.10}$$

$$c_t = f_t c_{t-1} + i_t \tanh\left(W_{xc}x_t + W_{hc}h_{t-1} + b_c\right) \tag{2.11}$$

$$o_t = sigmoid\left(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o\right) \tag{2.12}$$

$$h_t = o_t \tanh(c_t) = sigmoid(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \tanh(c_t) \quad (2.13)$$

where f_t, i_t, o_t and c_t are input gate, forget gate, output gate and current cell activation vectors respectively, all of which with the same size as the hidden vector $h \in R^{h\times 1}$ representing the output information of LSTM at each moment. The vector x represents the embedding vector of the preceding word with the size $x \in R^{a\times 1}$. W refers to weight matrices. For example, W_{xi} stands for the matrix of weights from the input gate to the input, and W_{hf}, W_{hi}, and W_{ho} means the weight matrix of the corresponding gate at the hidden state, with the size $R^{h\times h}$. b is bias vectors, $t-1$ is the previous cell state.

The forget gate can selectively forget some cell state information. The input gate ignores new information and learns new information. The cell state determines the output of the output gate, which is corresponding to the output of the hidden layer of the feedforward neural network. The output gate is the output of LSTM, which is determined by the current cell state.

By combining the effect of those gates and adjusting the weights of the gates during training, the network learns word-level semantic context storing and retrieving information over a long period (L. Huang, Xu, Sun & Yang, 2017).

However, LSTM-RNNs are sensitive to static data, which leads to target delays. Therefore, BiLSTM appears based on an exceptional architecture that operates the input sequence in both directions to make decisions (Passricha & Aggarwal, 2020).

For example, a BiLSTM-based teacher-student (BiLSTM-TS) model is applied to use the factual data information during training and solve the data mismatch problem, which reduces to 4% average word error rates compared to a conventional model (Liu et al., 2019). BiLSTM can also be used in the hybrid acoustic model to achieve computational efficiency. For example, a hybrid acoustic model with three BiLSTM, two CNN, and three fully-connected layers along with max-out neurons and dropout achieve an improvement of 5.8% compared to a CNN-based system and 10% compared

to a DNN-based system (Passricha & Aggarwal, 2020).

2.7 The End-to-End Framework for Speech Recognition

The end-to-end speech recognition system refers to directly transduce the input sequence of acoustic feature to the output sequence of token such as phonemes, characters, and words (J. Li, Zhao, Hu & Gong, 2019). The end-to-end model is divided into three categories based on the different alignment methods: Connectionist Temporal Classification (CTC), attention encoder-decoder (AED), and RNN Transducer (RNN-T), which have been widely utilized in large-scale speech recognition systems (J. Li et al., 2019). End-to-end models are more suitable for on-device applications than conventional speech recognition systems because there are fewer parameters by folding the acoustic, pronunciation and language models into one neural network (B. Li et al., 2020).

2.7.1 Connectionist Temporal Classification (CTC)

CTC refers to an objective function, which can automatically infer the alignment in the speech label. Accordingly, the end-to-end system can maximize the possibilities of the output sequence by using a softmax output layer and summing over all possible input sequences efficiently (Zhang et al., 2019). However, when introducing the CTC target function to infer speech-label alignments, the incorporation and utilization of language models and lexicons in decoding and the shortage of a shared experimental platform for benchmarking are two considerable issues (Graves, Fernández, Gomez & Schmidhuber, 2006).

Compared to the basic extraction of high-dimensional speech features in terms of

depth, MCNN is able to improve speech recognition performance by extracting additional detailed speech features in terms of width as an augmentation of the CNNs. Accordingly, an end-to-end MCNN-CTC ASR system based on DCNN and CTC objective function was proposed, which performs better than DCNN-CTC from the perspectives of extracting more valuable features (Zhang et al., 2019). The proposed system extracts sufficient speech features with more robust nonlinear capability by extending more comprehensive network structure and the capability of taking an end-to-end training manner due to the CTC loss.

It is proved that the increase of the network depth and the number of hidden cells lead to better performance based on the experimental result of an end-to-end framework trained by CTC for an LVCSR task (Eyben et al., 2009). As for the Code-Switch speech recognition, the end-to-end framework in CTC models was firstly applied for the Code-Switch task by introducing an additional LID classifier to adjust the posteriors of initial CTC model (K. Li, Li, Ye, Zhao & Gong, 2019).

2.7.2 Attention Encoder-Decoder Model

Attention-based encoder-decoder model such as LAS (Listen, Attend and Spell) contains three main components: Encoder as an acoustic model, attender as alignment model, and decoder as language model (W. Chan, Jaitly, Le & Vinyals, 2015), which subsume the acoustic, pronunciation and language model components into a single neural network without a lexicon or a separate text normalization component (Chiu & Raffel, 2017).

Applied to a task for Google Voice Search, the proposed model achieves a WER of 5.6%, while a hybrid HMM-LSTM model achieves 6.7% WER. Throughout testing the same models on a dictation task, the proposed model reaches up to 4.1%, and the HMM-LSTM system attains 5% WER. The decoding process of sequence-to-sequence

(S2S) models with soft attention can incur a quadratic time and space cost, which is regarded as a challenge for online sequence transduction (Chiu & Raffel, 2017).

In order to address the online streaming challenge of the attention-based model, monotonic-chunkwise attention was proposed, which splits input sequences into numerous small chunks (Chiu & Raffel, 2017). Triggered attention equipped with the CTC-based classifier performs well to control the activation of the attention-based decoder (Moritz, Hori & Le Roux, 2019).

As for the application of the attention-based model in the Code-Switch environment, Shan et al. (2019) proposed three improvement approaches:

(1) Introducing multi-task learning;

(2) The utilization of word pieces rather than graphemes as English modelling units can not only reduce the modelling unit gap between mixed language, but also capture more context information and reduce the time of decoding;

(3) Applying transfer learning with a large size of monolingual data.

2.7.3 RNN Transducer

Recurrent neural network transducer (RNN-T) refers to a streaming, all-neural, sequence-to-sequence architecture, which jointly trains acoustic and language model components from transcribed acoustic data (Rao, Sak & Prabhavalkar, 2017).

In order to conduct speech recognition by using the end-to-end framework, (Rao et al., 2017) trained the entire neural network with the RNN-T loss and directly output the recognised transcript as a sequence of graphemes. They find that the recognition performance can be improved by utilising subword as the modelling unit, capturing more extended context and reducing substitution errors. The proposed RNN-T system contains a 12-layer LSTM encoder with a two-layer LSTM decoder trained with 30,000 subword units as output targets, which achieves a WER of 8.5% on voice-search and

5.2% on voice-dictation tasks (Rao et al., 2017).

A novel RNN-T model achieves stable training by the utilization of large batch size and word piece targets as well as a layer normalisation. The training speed is also improved by the utilization of a time-reduction layer, meanwhile reducing memory footprint and increasing computation speed by quantising network parameters. The novel RNN-T model improves WER by over 20% compared to a traditional CTC embedded model (He et al., 2019).

Chapter 3

Methods

In this chapter, we mainly introduce our research methods, and give a detailed description of the design of the comparative experiment, including preparing data, the description of pronunciation dictionary and language model, and the comparative experiment process of speech recognition system under different models. In this chapter, we will provide the theoretical basis and experimental design of speech recognition on small labeled datasets and explain the working principle of the speech recognition model and optimization process in each experimental description. We will also implement the comparative experiments of two speech recognition systems based on a Chinese dataset, an English dataset, and a Code-Switch dataset.

3.1 Introduction

In order to complete the main issues of this thesis, the method is divided into two parts. One part is about exploring speech recognition with traditional HMM method in small label corpus and the establishment of a comparative system based on this model system. The other is to investigate the application of CTC and attention mixed model based on the same corpus. The best performance is found through our experiments, and the results are comparable to the conventional model.

In the traditional model aspect, we implement an HMM-DNN speech recognition system on the basis of the traditional HMM speech recognition system of Kaldi. The application principle and function of the acoustic model, language model, and lexicon are described in detail. Then the components are trained and optimized based on the extraction of input features, the evaluation method of the language model, and the evaluation strategy of the speech recognition system. Finally, the optimized system is used to get the experimental results on three datasets.

In the aspect of end-to-end framework, it is divided into two parts. Firstly, the network model of attention end-to-end framework and four attention mechanisms will be compared and analyzed. Simultaneously, the application of transformer in attention speech recognition system is explored to determine how to apply transformer to speech recognition. The second part is the application of CTC, combined with the CTC model to build CTC and attention hybrid model, to implement an end-to-end speech recognition framework. Finally, the training and optimization process on Chinese, English, and mixed Chinese English datasets will be described in detail, and the experimental results will be presented.

3.2 Data Preparation

3.2.1 Language Features

Considering English and Mandarin are worldwide-used languages with low-resource environment of bilingual Code-Switch corpus, it is important to select appropriate recognition units in acoustic modelling, which convert the feature vector sequence to the speech recognition unit by recognizing the speech recognition unit corresponding to the feature vector sequence (Senior, Sak & Shafran, 2015).

English is an Indo-European language, while Mandarin is a Sino-Tibetan language (J. Y. Chan, Ching, Lee & Meng, 2004). Based on the Oxford Dictionaries, English is written in a Latin alphabet (namely, the Roman alphabet) containing 26 letters and nearly 170,000 words. The modelling units of English includes phone, subword and character (W. Wang et al., 2020).

In the field of Chinese speech recognition, there are various available acoustic modelling units, including Chinese characters (word), syllable (syllable), semi-syllable (initial/final), phoneme (phone), which is generally based on phonetic knowledge or data-driven generation (Zenkel, Sanabria, Metze & Waibel, 2017). Mandarin contains more than 6,000 characters, 60 phonemes, 408 atonal syllables and 1,302 toned syllables (Lin, Lee & Ting, 1993). Each syllable encompasses initials, finals and tones, including 22 initials and 39 finals of syllables.

Meanwhile, there are many homophones and polyphones in Chinese (Zheng, Yang & Dang, 2020), which may need high-level non-acoustic context knowledge for speech recognition. The small flexible unit may lead to the difficulty to calibrate the dataset. By contrast, the limitation of flexibility as well as the high requirement of lexicon and Out of Vocabulary (OOV) scenarios (Ueno, Inaguma, Mimura & Kawahara, 2018) are significant issues though the large unit equipped with high recognition performance.

The syllable unit can meet both the performance requirement and flexibility (Qu, Haghani, Weinstein & Moreno, 2017). Moreover, the utilization of syllable with tone effectively increases the recognition accuracy compared to Chinese characters and syllable with initial/final tone (Fu, Li & Zi, 2020).

3.2.2 Corpus

Based on the language features in Section 3.2.1, three speech datasets are applied to the experiment, including dataset Alpha (Mandarin), dataset Beta (English), and dataset Gamma (Mandarin-English).

We create the three datasets through the iFlytek InterPhonic toolkit. It is a text-to-speech software developed by iFLYTEK, which converts text into male or female voices. The iFlytek InterPhonic is based on an advanced large corpus and a phonetic prosody description, the sound quality of the synthesized $.wav$ format file is comparable to that of a real person (iFLYTEK Co., Ltd, 2020).

Converting the transcript text to synthesised acoustic speech is a solution to control the variables. For example, the speaker's accent, the speaker's emotion, and the noisy environment are consistent. The synthesised acoustic speech in our experiment focuses on the recognition results of different language environment without considering other variables.

Dataset Alpha (Mandarin)

Alpha (Mandarin) is a Mandarin speech dataset. We create this dataset through the iFlytek InterPhonic toolkit based on the transcript files of the THCHS-30 corpus.

THCHS-30 is an open-source Chinese corpus released by the Center for Speech and Language Technology at Tsinghua University China. THCHS-30 contains over 30 hours of Mandarin speech footage recorded by a single carbon microphone at a

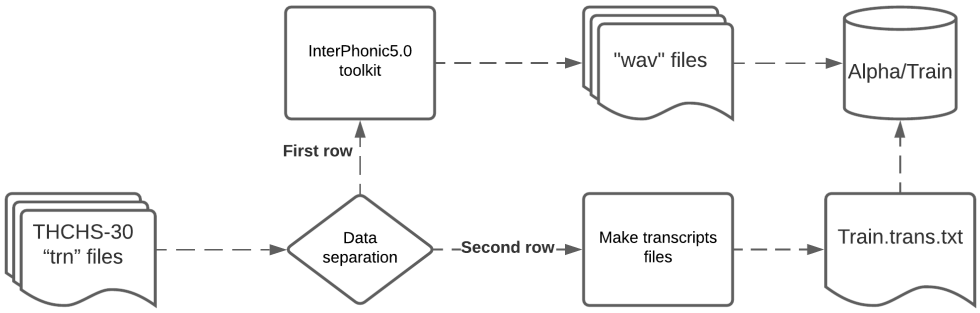


Figure 3.1: The Process of Creating Alpha-Train

silent office with a sampling rate of 1.6 kHz and a sample size of 16 bits (Dong Wang, 2015). As shown in Table 3.1, the “Train” dataset of THCHS-30 contains 29.23-hour 10,893 utterances from 8 male speakers and 22 female speakers. The “Test” dataset of THCHS-30 contains 6.24-hour 2,496 utterances from 1 male speaker and 9 female speakers.

In the THCHS-30 corpus, there are a number of $.trn$ transcript files corresponding to the wav acoustic files. Each $.trn$ file contains three rows, which are the correct labeled transcripts of the corresponding acoustic sentence. The first row is the Chinese character of the corresponding acoustic sentence. The second row is the Chines Pinyin with the four tones, corresponding to the Chinese characters. The third row is the syllable initial-final with tone, which corresponds to the Chinese Pinyin.

As shown in Fig. 3.1, we select 8,000 $.trn$ files of the THCHS-30 corpus. The first row (Chinese characters) of each selected $.trn$ file are collected as the input of the iFlytek InterPhonic toolkit. The selected Chinese characters are synthesised into 8,000 $.wav$ files sampled at 16 kHz. The second row (Chinese Pinyin with tones) of each selected .trn file is collected together as one text file as our labeled transcript for training. Through this operation, we create the “Train” subset of Alpha (Mandarin).

Similarly, we create the “Dev” subset and the “Test” subset of Alpha (Mandarin) through the same operation process. The “Dev” subset is synthesised from other 1,000

$.trn$ files of the THCHS-30 corpus except for the selected 8,000 trn files. The "Test" subset is synthesised from other 1,000 $.trn$ transcript files except for the selected 9,000 $.trn$ files.

Namely, the synthesised $.wav$ acoustic files in the dataset Alpha (Mandarin) are split into three groups as shown in Table 3.2: 1-hour acoustic files as the testing subset called Alpha-Test, 1-hour acoustic files as the development subset called Alpha-Dev to train the rapid reaction and performance evaluation, the other 10-hour acoustic files as the training subset called Alpha-Train.

Table 3.1: The Statistics of THCHS-30

Dataset	Speaker	Male	Female	Utterance	Time(hour)
Train	30	8	22	10893	27.23
Test	10	1	9	2496	6.24

Table 3.2: The Statistics of Database Alpha (Mandarin)

Dataset	Time(hour)	Speakers	Male	Female	Percentage of THCHS-30
Alpha-Train	10	2	1	1	29.878%
Alpha-Dev	1	2	1	1	2.988%
Alpha-Test	1	2	1	1	2.988%

Dataset Beta (English)

Dataset Beta (English) is an English speech dataset. We create this dataset through the iFlytek InterPhonic toolkit based on the transcript files of the LibriSpeech corpus.

The original LibriSpeech corpus is a large English corpus containing 1,000 hours of speech sampled at 16 kHz based on the audiobooks of LibriVox project (Panayotov, Chen, Povey & Khudanpur, 2015). LibriSpeech is divided into data subsets based on recording quality and WER as shown in Table 3.3. Both the dev-clean subset and test-clean subset contain 5.4-hour utterances from 20 male speakers and 20 female speakers, in which "clean" refers to the speakers with the low-WER recording quality.

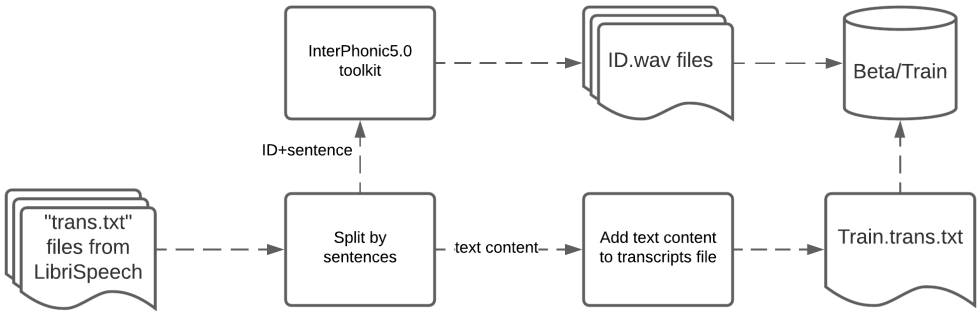


Figure 3.2: Process of Creating Beta-Train

The train-clean-100 subset is selected from the rest of the clean data, including 100.6-hour utterances from 125 female speakers and 126 male speakers.

In the LibriSpeech corpus, there is a $.txt$ transcript file corresponding to the $.wav$ acoustic files. Each row of the txt transcript file consists of a unique ID and a capitalized English sentence. Each unique ID corresponds to one unique $flac$ acoustic file. The capitalized English sentence is the correct labeled transcript of the corresponding acoustic sentence with the same ID.

As shown in Fig. 3.2, we select 8,000 lines of the English sentence from the $.txt$ transcript file in the LibriSpeech as the input of the iFlytek InterPhonic toolkit. The selected English sentences are synthesised into 8,000 $.wav$ files sampled at 16 kHz. The selected text contents are employed as the labeled transcript file for training. Through this operation, we create the "Train" subset of Beta (English).

Similarly, we create the "Dev" subset and the "Test" subset of Beta (English) through the same operation process. The "Dev" subset is synthesised from other 1,000 rows of the $.txt$ transcript files in the LibriSpeech corpus except for the selected 8,000 rows. The "Test" subset is converted from other 1,000 rows of the $.txt$ transcript files except for the selected 9,000 rows.

Namely, the synthesised $.wav$ acoustic files in the dataset Beta (English) are divided into three groups as shown in Table 3.4: 10-hour acoustic files synthesised as the

training subset called Beta-Train, 1-hour audio files as the development subset called Beta-Dev to train the rapid reaction and performance evaluation during training, and 1-hour audio files as the testing subset called Beta-Test.

Table 3.3: The Statistics of LibriSpeech

Dataset	Male	Female	Time(hour)
dev-clean	20	20	5.4
test-clean	20	20	5.4
test-clean-100	126	125	100.6

Table 3.4: Dataset Beta (English)

Dataset	Time(hour)	Speakers	Male	Female	Percentage of LibriSpeech-clean
Train	10	2	1	1	8.977%
Test	1	2	1	1	0.898%
Dev	1	2	1	1	0.898%

Dataset Gamma (Mandarin-English)

Dataset Gamma (Mandarin-English) is a mixed Mandarin-English bilingual speech dataset. We created this dataset through the iFlytek InterPhonic toolkit based on the transcript files of the TAL-CSASR courpus.

The original TAL-CSASR corpus is the audio in English class teaching environment (TAL Education Group, 2019). This dataset contains 587 hours of audio from over 200 teachers, sampled at 16KHz and 16 bit, which is created by Beijing Century TAL Education Technology Co., Ltd. in 2019. Considered the teachers are Chinese, who teach bilingually in both Chinese and English, the TAL-CSASR corpus is used in our experiments on speech recognition in the Code-Switch environment.

In the TAL-CSASR corpus. there is a transcript file called label. Each row of the transcript file consists of a unique ID and a mixed Mandarin-English sentence including Chinese characters and Capitalized English words. Each unique ID corresponds to

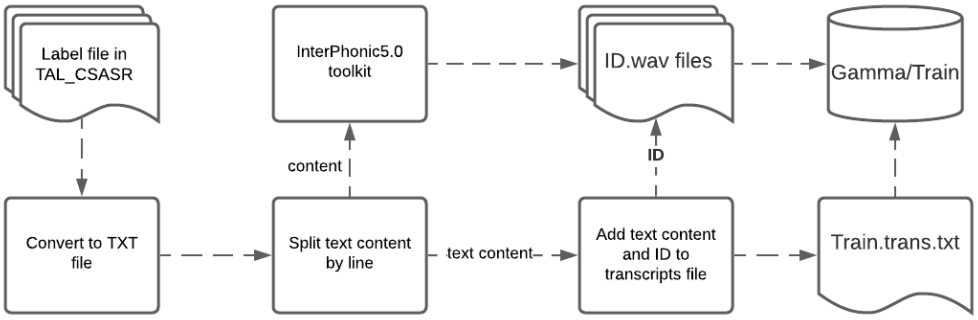


Figure 3.3: The Process of Creating Dataset Gamma

one unique $.wav$ acoustic file. The Mandarin-English sentence is the correctly labeled transcript of the corresponding acoustic sentence with the same ID.

As shown in Fig. 3.3, we select 8,000 rows of the mixed Mandarin-English sentences from the transcript label file in the TAL-CSASR corpus. The selected mixed Mandarin-English sentences are synthesised through the iFlytek InterPhonic toolkit. The synthesised acoustic $.wav$ files are sampled at 16 kHz. The selected Mandarin-English sentences are also used as our labeled transcript file for training. Through this operation, we create the "Train" subset of Gamma (Mandarin-English).

Similarly, we create the "Dev" subset and the "Test" subset of Gamma (Mandarin-English) through the same process. The "Dev" subset is synthesised from the other 1,000 rows of the transcript label file in the TAL-CSASR corpus except for the selected 8,000 lines. The "Test" subset is synthesised from the other 1,000 lines excepted for the selected 9,000 lines.

Namely, the synthesised $.wav$ acoustic files in the dataset Gamma (Mandarin-English) are categorised into three groups as shown in Table 3.5: 10-hour training subset called Gamma-Train, 1-hour audio files as the development subset called Gamma-Dev to train the rapid reaction and performance evaluation, and 1-hour audio files as the testing subset called Gamma-Test. All the subsets are recorded from 1 male speakers and 1 female speakers.

Table 3.5: Dataset Gamma (Mandarin-English)

Dataset	Time(hour)	Speakers	Male	Female
Train	10	2	1	1
Dev	1	2	1	1
Test	1	2	1	1

3.2.3 Lexicon and Language Model

The lexicon applied to the dataset Alpha (Mandarin) comes from the 9,225-row $lexicon.txt$ file of the THCHS-30 corpus. The lexicon used in the dataset Beta (English) comes from the CMU English dictionary (Panayotov et al., 2015). The dataset Gamma (Mandarin-English) is from CMU, and a Chinese dictionary (Hsiao, Fuhs, Tam, Jin & Schultz, 2008), consisting of 33,000 English words and about 6,000 Chinese words. The phoneme set consists of 252 phonemes, including 213 Chinese phonemes and 39 English phonemes. The data-driven approach is employed to model the pronunciation probability of words, and the maximum cross-entropy training criterion is applied to build grammar with a 3-gram language model. The language model is derived from the training set transcription of the dataset.

3.2.4 Acoustic Model

As a generative model for speech recognition, the HMM is usually jointly utilised with a Gaussian mixture model (GMM) or a DNN to model acoustic data (Y. Huang, Yu, Gong & Liu, 2013; L. Li et al., 2013; Z.-J. Yan, Huo & Xu, 2013; Georgescu & Cucu, 2018).

By considering the method performing sequence discriminative training of the acoustic model without frame-level cross-entropy pre-training (Povey et al., 2016), the lattice-free version of the Maximum Mutual Information (MMI) (LF-MMI) performs better compared to the conventional cross-entropy training of DNNs (Hadian, Sameti,

Povey & Khudanpur, 2018), we apply the LF-MMI to build the CNN-TDNN-F acoustic model. The standard objective function applied in Maximum Likelihood training is as Eq. (3.1).

$$F_{ML}(\lambda) = \sum_{r=1}^{R} \log p_\lambda \left(X_r \mid S_r \right) \tag{3.1}$$

where Eq. (3.1) illustrates the likelihood of the observation of training data given the correct-transcription HMM, λ refers to all HMM parameters, R stands for the total number of all training utterances r, S_r is the correct transcription of the input sequenceX_r.

Based on the previous work on MMI (McDermott, Hazen, Le Roux, Nakamura & Katagiri, 2006), which is the most popular discriminative training criteria that equals to the posterior probability of the correct sentence S_r, the MMI objective function is written as Eq. (3.2).

$$F_{MMI}(\lambda) = \sum_{r=1}^{R} \log \frac{p_\lambda \left(X_r \mid S_r \right)^\kappa P \left(S_r \right)^\kappa}{\sum_s p_\lambda \left(X_r \mid S \right)^\kappa P(S)^\kappa} \tag{3.2}$$

Different from S_r corresponding to the numerator graph specific to a word sequence in transcription, S refers to the denominator graph modelling all possible word sequences (Madikeri et al., 2020). $P(S)$ is the language model probability for sentence S, which contains scales and word insertion penalties. κ is the probability scale as the model parameter.

In place of the word language model, the utilization of an n-gram phoneme language model makes the computation feasible, which integrates acoustic information encoding all possible word sequences into an HMM graph as the denominator graph as M_{den}.

Accordingly, the denominator part of Eq. (3.2) is rewritten as $p_\lambda(X_r|M_{den})$. Similarly, the numerator part of Eq. (3.2) is denoted as $p_\lambda(X_r|M_{num})$. Therefore, Eq. (3.2)

is shown as Eq. (3.3).

$$F_{MMI}(\lambda) = \log \frac{p_\lambda \left(X_r \mid M_{num}\right)}{p_\lambda \left(X_r \mid M_{den}\right)} \tag{3.3}$$

Moreover, there are single task or multitask model of LF-MMI when training multi-language datasets, depending on whether languages share the output layer or not (Mao & Zhang, 2019). There is a separated output layer of each language preceded by a pre-final layer and a corresponding objective function in the multitask architecture.

There are two sources of the neural network input of the acoustic model. One is the 40-dimensional Fbank feature obtained through the inverse discrete cosine (IDCT) transformation from the 40-dimensional MFCC feature. The other is a 200-dimensional vector obtained through a linear transformation from the 100-dimensional vector feature. Those are spliced to obtain a 240-dimensional feature sending to the network, composed of 6 layers of 2-dimensional convolutional neural networks, and nine layers of singular value decomposition time-delayed neural networks from bottom to top.

The objective function of acoustic model applies LF-MMI, but the posterior probability from the forward and backward calculation of the molecular graph is applied to the cross-entropy objective, which is used as the standard term of the LF-MMI objective function.

As for the low-rank matrix decomposition of time-delayed neural network, the number of neurons in the time-delayed neural network layer is 1,536, the number of neurons in its bottleneck layer is 160. The network takes use of 6 epochs, the initial learning rate is 0.00025, the terminated learning rate is 0.000025, the batch size is 64, and the leak HMM coefficient is 0.1.

3.2.5 Acoustic Modelling Units

Dataset Alpha (Mandarin)

Based on previous work (Zou, Jiang, Zhao, Yang & Li, 2018), especially various acoustic modelling approaches for Mandarin speech recognition, such as character, syllable with tone and syllable initial/final based context-dependent phoneme as shown in Table 3.6, in this experiment, we choose the syllable with tone approach as the acoustic modelling unit of dataset Alpha (Mandarin).

The syllable-based approach performs better than the phoneme-based approach in intractable variations and long temporal dependency scenarios (H. Wu & Wu, 2007), it also outperforms than the character-based approach by avoiding the out-of-vocabulary condition.

Table 3.6: An Example of Mandarin Acoustic Modelling Units

Modelling unit	Example
Character	绿 是 阳 春 烟 景
Word	绿 是 阳春 烟景
Syllable with Tone	lv4 shi4 yang2 chun1 yan1 jing3
Context-dependen Phoneme	sil-l+v4 l-v4+sh v4-sh+i4 sh-i4+y i4-y+ang2 y-ang2+ch ang2-ch+un1 ch-un1+y un1-y+an1 y-an1+j an1-j+ing3 j-ing3+sil

Dataset Beta (English)

Based on previous work (Goussard & Niesler, 2010; Razavi & Doss, 2015; Smit, Virpioja & Kurimo, 2021), especially the various acoustic modelling approaches for English speech recognition, such as word, character and subword as shown in Table 3.7, in this experiment, we choose subword approach as the acoustic modelling unit of dataset Beta (English).

Table 3.7: An Example of English Acoustic Modelling Units

Modelling unit	Example
Word	Do you like the newest music festival
Character	D o y o u l i k e t h e n e w e s t m u s i c f e s t i v a l
Subword	Do you li ke the n e w est mu si c fe s ti val

Dataset Gamma (Mandarin-English)

The acoustic modelling units of Mandarin consist of character and syllable. The acoustic modelling units of English include character and subword. Accordingly, the four combinations of acoustic modelling units for the Mandarin-English Code-Switch environment are listed in Table 3.8. Our experiment applies Syllable-Subword to the modelling unit.

Table 3.8: An Example of Mandarin-English Acoustic Modelling Units

Modelling unit	Example
Character-Character	那个 m u s i c f e s t i v a l 听起来不错
Character-Subword	那个 _mu si c _fe s ti val 听起来不错
Syllable- Character	na4 ge4 m u s i c f e s t i v a l ting1 qi3 lai2 bu2 cuo4
Syllable- Subword	na4 ge4 _mu si c _fe s ti val ting1 qi3 lai2 bu2 cuo4

3.3 Speech Recognition Based on HMM-DNN

Based on previous work on HMM-DNN (Kanda, Takeda & Obuchi, 2013; Maas et al., 2017), the overall idea of the HMM-DNNs model refers to the utilization of DNN to train the classification model $P(s \mid x)$, then $\frac{P(s|x)}{P(s)}$ is applied to convert the output probability distribution required by HMMs, namely, replaced by using GMMs, after that, combines HMMs to build a complete speech recognition model. More specifically, the overall training and recognition process of the HMM-DNN model is summarized as the following steps:

(1) Utilise all the training data (X, S) to train the GMMs-HMMs model, and label the corresponding optimal state sequence for each X according to the Viterbi decoding algorithm (Seshadri & Sundberg, 1994), in which each frame corresponds to a state label $(x, state)$;

(2) Use $(x, state)$ marked in the previous step to train the DNN model. The input is the current frame including its left and the right context. The output is the score of the states in the HMM which is optimized by cross-entropy loss. There are a number of tricks that are employed in this step such as Sequence Discriminative Training (SDT) (Veselỳ, Ghoshal, Burget & Povey, 2013) and Speaker Adaption Training (SAT) (Abdel-Hamid & Jiang, 2013);

(3) According to the DNN in the previous step, we re-estimate the transition probability parameters of the HMM, then relabel the state, and train the DNN, we repeat the step until achieving convergence;

(4) After trained DNN and HMMs, the recognition process is as same as that of the GMM-HMM model (Yu & Deng, 2016).

There are various DNN models, including Feedforward DNN(FFDNN), Context-Dependent DNN(CDDNN), Time-Delay NN(TDNN), RNN, as well as tricks that are used in training such as pre-training initialization, SDT, SAT. Considering the good performance of TDNN in the low-resource speech recognition environment (Biswas, Menon, van der Westhuizen & Niesler, 2019; Fathima, Patel, Mahima & Iyengar, 2018), in this section, the focus is on the utilization of TDNN-F as DNN in the HMM-DNN model, especially the factorized TDNN (TDNN-F).

3.3.1 Description of The TDNN-F Neural Network

Time Delay Neural Networks (TDNNs), namely one-dimensional CNNs, refers to an efficient neural network architecture, which was firstly proposed by Waibel, Hanazawa,

Hinton, Shikano and Lang in 1989. TDNN contains a typical parameter matrix, rectified linear unknit and batch normalization. As shown in Eq. (3.4), the frames of a continuous temporal window will be multiplied through corresponding weights before inputting the sum to a nonlinear function.

$$\mathrm{y}_t = sigmoid[\sum_{n=-N_1}^{N_2} \mathrm{W}_{n^{\mathrm{x}}t+n} + \mathrm{b}] \tag{3.4}$$

where x_{t+n} refers to the frame if time is $t+n$, namely, the $n-th$ frame after the current frame. N_1 and N_2 refer to the respective numbers of historical frames and future frames. However, the utilization of the continuous temporal window in the conventional TDNN may lead to a large model size even with a small window. In order to improve the performance and stability of training without losing any modelling power, Povey et al. and (2018) proposed a factored form of TDNNs (TDNN-F) by compressing the layers of TDNN through Singular Value Decomposition (SVD), factorizing the matrices into products of two smaller pieces, and training them from a random start with one of the two factors of each matrix constrained to be semi-orthogonal.

Furthermore, semi-orthogonal refers to a generalization to non-square matrices of orthogonal (Povey et al., 2018). For example, M is semi-orthogonal if $\mathbf{MMT} = \mathbf{I}$ or $\mathbf{MTM} = \mathbf{I}$. From the perspective of TDNN, TDNN-F introduces a bottleneck layer between the two feedforward layers, in which the linear bottleneck dimension is less than the hidden layer dimension. For example, transforming the original parameter matrix $\mathbf{M}_{800\times2300}$ into factorized matrices is $\mathbf{M} = \mathbf{AB}$, where $\mathbf{A}_{800\times300}$, $\mathbf{B}_{300\times2300}$, then the linear bottleneck dimension 300 is smaller than the hidden layer dimension 800. From the perspective of one-dimensional CNN, if the columns of $\mathbf{M}$ correspond to 5 frame offsets of the previous layer spliced together, TDNN-F will introduce a $5\times1\times800$ convolutional layer with semi-orthogonal parameters, in which the convolution kernel is 5×1, and the number of filters is 800.

Moreover, the example of Kaldi applies the neural network structure of $1536 \times 160 \times 1536$ to deal with the corpora of swabs and librispeech, in which the middle layer dimension is 160. Besides increasing the factorized convolution layer (Povey et al., 2018) there are several other improvements of TDNN-F compared to TDNN. For example, the utilization of across time dropout method can avoid overfitting and improve the result by about 0.2% or 0.3% absolute. Factorizing the final layer can improve the effect even if the dataset is small. Moreover, the utilization of skip-connection can ensure the information transfer to the deep layers, which helps to alleviate the problem of vanishing gradients and improve the result.

3.3.2 Experiments

Experimental Environment and Setup

We utilise Kaldi in this experiment. Kaldi refers to an open-source toolkit for speech recognition licensed under the Apache License v2.0 that is available on SourceForge, which aims to have modern and flexible code that is easy to understand, modify and extend (Povey et al., 2011). The Kaldi-based RNN language model aims to construct a language model and re-score the first decoded lattice. The input of RNN language model is an 800-dimensional word embedding vector. The time-delayed neural network and the long and short-term memory network with projection appear alternately. There are six network layers in total. The dimension of LSTM cell with projection is 800, the projection dimension of the cyclic part is 200, and the non-projection dimension of the cyclic part is 200. Moreover, Table 3.9 shows the details of the experimental hardware information and operating system.

This experiment takes use of the PyTorch deep learning framework based on LAS in the RNN language model. PyTorch is the dynamic deep learning neural network engine of ESPNet provided the architecture with a single neural network to perform

Table 3.9: Hardware Equipment of Experiment

OS Name	Ubuntu20.04
OS Type	64-bit
Memory	32 GiB
Processor	Inter(R)Core i9-9900k CPU @ 3.60GHZ x 16
GPU	GeForce RTX 2080Ti 11GiB x 2
Disk Capactiy	1.5T

the end-to-end framework in speech recognition (Watanabe et al., 2018). The training parameters are shown in Table 3.10. The network structure of RNN language model contains two-layer unidirectional LSTM. The cell size is 1024, using regularized dropout, the dropout rate is 0.5. The stochastic gradient descent (SGD) optimizer is applied, and the batch size is 16. The early-stop mechanism is applied to improve the generalisation ability of this model, speed up the training process, and reduce the risk of overfitting (Ji, Zhang & Wu, 2020) with the stop threshold 3, training 20 epochs. The language model is employed to assist the analysis and comparisons in the end-to-end framework.

Table 3.10: Training Parameters

Parameters	Value
Neural Network	TDNN-F
RNNLM LSTM layers	2
Cell size	1024
Dropout rate	0.5
Batch size	16
Optimizer	Stochastic gradient descent
Epochs	20

Evaluation Matrix

The most straightforward approach to evaluate the performance of speech recognition is by running and calculating the Word Error Rate (WER) (Mansikkaniemi, 2010).

WER is employed for the English dataset, which is calculated through Eq. (3.5).

$$R = \frac{I_E + D_E + S_E}{N} \times 100\% \tag{3.5}$$

where I_E refers to the insertion number of English word, D_E means the deletion number of English word, S_E stands for the substitution number of English word, and N is the total number of English word in the standard correct sentence. However, characters are considered instead of words in the Chinese THCHS-30 corpus. Similarly, the character error rate (CER) is calculated through Eq. (3.6), in which the suffixes ended with M take the place of E.

$$R = \frac{I_M + D_M + S_M}{N} \times 100\% \tag{3.6}$$

Besides character recognition, syllable recognition is another version of transcriptions for the Mandarin dataset, in which the recognition performance is measured by using syllable error rate (SER). The result of SER is based on how much syllable string is returned by using a recognition engine that differs from a correct transcription, and evaluates the effectiveness of adaptation methods of syllable decoding-based ASR system(C. Li, Chen & Xu, 1999).

Based on the previous work to avoid the homophone issue, which refers to the same pronunciation with different meaning, in this thesis, we choose to decompose the Chinese words and characters into spelled sounds Pinyin.

Experimental Results

The experiment results of the "Test" subset of datasets Alpha, Beta, and Gamma in the TDNN-F model are shown in Table 3.11. The CER of dataset Alpha (Mandarin) in the TDNN-F model reaches 10.91%. The WER of dataset Beta (English) in TDNN-F

attains 18.23%. The WER of dataset Gamma (Mandarin-English) is up to 25.62%. It is clear that the CER of Mandarin is lower than the WER of English in speech recognition. Monolingual speech recognition performs better than mixed bilingual speech recognition. Moreover, as shown in Table 3.12, the dataset with a larger size of data performs better than the dataset with a smaller size. As shown in the comparison Table 3.13, the TDNN-LSTM model (J. Li, Shan, Wang & Li, 2018) performs 9.55% CER in Mandarin recognition. The TDNN model (Povey et al., 2016) performs 4.28% WER in English recognition.

Table 3.11: Experiment Results

Model	Dataset	WER/CER
TDNN-F	Alpha (Mandarin)	10.91%
TDNN-F	Beta (English)	18.23%
TDNN-F	Gamma (Mandarin-English)	25.62%

Table 3.12: Beta (English):Comparison of Different Models

Model	Dataset	WER
TDNN	1,000h LibriSpeech	4.28%(Povey et al., 2016)
TDNN	960h LibriSpeech	4.83%(Peddinti, Povey & Khudanpur, 2015)
TDNN-F	Beta(English)	18.23% (this experiment)

Table 3.13: Comparison of Different Models

Model	Dataset	Mandarin	English	Code-Switch
(J. Li et al., 2018)	THCHS-30	9.55%	-	-
(Povey et al., 2016)	LibriSpeech	-	4.28%	-
TDNN-F	Alpha	10.91%	-	-
TDNN-F	Beta	-	18.23%	-
TDNN-F	Gamma	-	-	25.62%

3.3.3 Summary

On the basis of the HMM framework, we select a specific network TDNN-F as the acoustic model in this section, which is a semi-orthogonal low-rank matrix factorization delay network. In this section, we outline the network structure, explain the n-gram-based language model, acoustic model, pronunciation lexicon, and configuration of RRN language model based on the Kaldi version. experimental verification, the resultant analysis and comparison of three multilingua datasets are also carried out.

3.4 RNN-CTC Model for Speech Recognition

The idea of CTC networks is to align speech directly with corresponding text. The motivation is that speech recognition is a monotonic audio-to-text alignment process that is chronologically sequential. The model shows excellent performance in practical experiments (Graves et al., 2006). Therefore, we take CTC as the basis to prob the alignment ideas and algorithms and test their performance in Chinese, English and Chinese mixed speeches.

Ins the speech recognition, we solve the problem that the target label and each input frame need an alignment by the utilizing CTC and deep neural network. We need to know which input frame corresponds to which output character and how to split the boundary of input frames corresponding to different output characters. In different cases, this boundary is blurred, and the data that needs to be tagged frame by frame is much more demanding than tasks that only require simple text output. Traditional acoustic model training for speech recognition, for each frame of data, the corresponding labels for data are needed to know before training to achieve practical training. Therefore the alignment preprocessing of speech needs to be prepared before training. The process of voice alignment itself requires repeated iterations to ensure

more accurate alignment, which is a time-consuming work.

Compared with the traditional acoustic model, the acoustic model training using CTC as the loss function is a complete end-to-end acoustic model training. It does not require pre-alignment of the data. Only one input sequence and one output sequence are needed to be trained. This eliminates the need for data alignment and labeling, CTC directly outputs the probability of sequence prediction without external postprocessing. Moreover, CTC introduces blank (i.e., the frame has no predictions), a spike in a speech segment corresponding to each predicted classification, and blank in other locations that do not spike. For a speech, the final output of the CTC is a spike sequence, regardless of how long each phoneme lasts (Z. Chen, Deng, Xu & Yu, 2016).

3.4.1 Principle of CTC

In speech recognition, CTC is connected to RNN. Therefore, the CTC is treated as RNN and CTC as a whole unless otherwise specified. In order to analyze whether the CTC is suitable for this thesis, we apply formal language to describe the thought of CTC in speech recognition. It is assumed that the characteristic sequence of output and playback audio is X. In order to fit this project, the input X may include Chinese and English. X = {x_1, x_2, x_3, x_4,$\cdots$, x_n}, the corresponding text sequence is Y, Y = {y_1, y_2, y_3, y_4, $\cdots$, y_m}. The task of speech recognition is to find an accurate mapping from X to Y. In order to achieve this alignment, there are two challenges. Firstly, the length of X is different from that of Y, n may not equal to m. Secondly, the correspondence between X and Y is uncertain. CTC can solve these two problems. It is able to give all possible alignment relations and calculate the probability of these alignment relations. The calculation and execution of the loss function in the algorithm will be demonstrated and tested in the later part of this thesis.

3.4.2 Score Function

Targeting the goal, we train a model to get the maximum accuracy of sequence x to sequence y alignment. In order to achieve this goal, the probability $P_{CTC}(Y \mid X)$ must be calculated. The gradient descent method is utilised to calculate the parameters of the model. The equations are shown as Eq. (3.7) and Eq.(3.8).

$$P_{frame}(X) = \sum_{i=1}^{n} P(Frame_n|X) \tag{3.7}$$

$$P_{ctc}(X \mid Y) = \sum_{i=1}^{n} P(x_i|x_{i-1}, Y)P_{frame}(X) \tag{3.8}$$

where $P_{frame}(X)$ represents the probability that each frame will be output, $P(x_i|x_{i-1}, Y)$ is the probability of connection between characters or between words. Both of them are calculated by the Forward-Backward Algorithm. Then in the prediction phase, CTC finds the output Y_{max} corresponding to the maximum probability for each input X. The mathematical description is as provided in Eq. (3.9).

$$Y_{max} = \arg\max_{Y} \log P_{ctc}(Y \mid X) \tag{3.9}$$

If the sequence length is slightly longer, the calculation amount of all probability distributions is enormous, so a faster algorithm is needed. The outcomes of related work (Kim et al., 2017) and (Ruder, 2017) inspired us. After combining the attention model, we utilise the proposed multitask learning method to accelerate the convergence of prediction calculations. This content will be introduced later in the CTC/Attention hybrid model. In this chapter, we focus on the accuracy of CTC based on different speech test datasets and do not optimize its convergence rate.

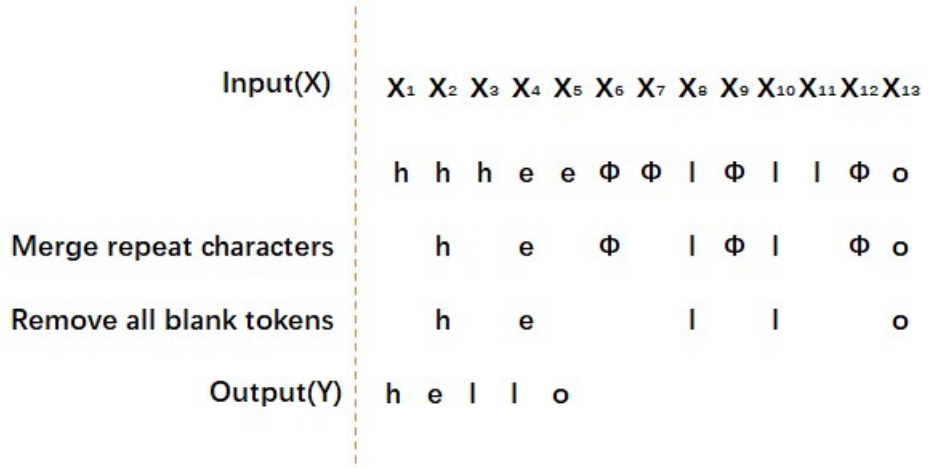


Figure 3.4: CTC Alignment Example

3.4.3 Alignment

CTC introduces the concept of blank on the alignment issue, we denote it as Φ, which represents a placeholder and does not correspond to any characters, so it should be deleted in the final output. The alignment process of the word hello is shown in Fig. 3.4.

If there are two identical characters in a word, in order to handle this case, we insert Φ between these two characters. In this way, it is possible to distinguish alignments such as “good” and “god”. This is critical in English speech recognition. It distinguishes the concatenation of Chinese and English if Chinese and English are mixed. The rules have the following characteristics. Firstly, the alignment of input X and output Y is monotonic. If X is prior to the input component X_{i+1} corresponding to the next time slice, Y either stays still or moves to the next. The output component Y_{j+1} corresponds to the time slice. Secondly, there is a one-to-many relationship between input and final output. Namely, multiple-input components may only correspond to one output component. Therefore, the length of output X must not be greater than Y.

3.4.4 Beam Search

For an input sequence X, the aligned output sequence is called a path. The summation on the right side of the equal sign in Eq. (3.8) refers to the summation of all possible alignment path probabilities corresponding to the input sequence X, and these alignment sequences are mapped to the correct output sequence Y (Sim, Narayanan, Bagby, Sainath & Bacchiani, 2017). The purpose of this model is to maximize the probability. The training model here mainly refers to the encoder part of CTC training.

Two types of coding networks will be applied in this chapter. One is the transformer, and the other is the RNN encoder. In the experiment part, two models will be described in detail. The function of the encoder is to estimate the probability of the current label in the current time step. The output label of the next time step is independent on the output label of the current time, which is determined by the CTC condition independent assumption. Therefore, the network can learn the context information if the RNN or the decoder of the transformer output the current probability. This is very important for the CTC distribution algorithm to work in the task of speech recognition.

At the prediction phase, given an input X, we calculate the output sequence corresponding to the maximum probability. If it is assumed that the time slices are independent of each other, then only the character with the highest probability corresponding to each time slice is used as the predicted value. Then the sequence is formed, and the final result is obtained by de-duplication and other processing. However, this does not consider that multiple sequences are aligned corresponding to the same output result. For example, suppose $[s, s, \Phi]$ and $[s, s, s]$ have respective probabilities lower than $[s, i, s]$, and the sum of their probabilities is higher than the latter. The alignment result of the former is $[s]$, while the alignment result of the latter is still $[s, i, s]$. Clearly, the output $[s]$ is mUCH reasonable than $[s, i, s]$.

In order to avoid this problem, we manipulate an algorithm called Beam Search (Drexler

& Glass, 2019). There is a parameter B in the training parameters to specify the number of prefix sequences retained each time. If $B = 3$ is set, the three prefix sequences with the highest probability are selected each time. For example, the three characters with the highest probability are selected when the time step is 1, and the three characters with the highest probability are also selected when the time step is 2. There are nine combinations. The alignment may correspond to the same output, so the prefix sequences with the same result should be merged (the probabilities should also be added), and then the 3 with the highest probability should be selected as the next output, and so on.

3.4.5 Experiment

Based on the working principle and application method of CTC, in this section, we provide the experimental content, which will compare the recognition performance of the cyclic neural network and CTC model with different speech datasets. In this section, we describe the experimental content through four parts: Data preparation and feature extraction, network structure, experimental process and experimental results.

Data and Feature

In the experiment of CTC, the data is from the corpus introduced in Section 3.2.2, the "Test" datasets of three different languages are used to evaluate the model. Considering that our Chinese-English mixed datasets is a relatively small datasets, we adopt data enhancement methods to improve the effect of network training. Firstly, we perform feature extraction on the audio data, obtain 40-dimensional Fbank and 3D pitch, and then utilise SpecAugment (D. S. Park et al., 2019) to enhance the features. Finally, the features of 40D Fbank and 3D pitch are obtained. The length of frame is 25 milliseconds and the shifting of frame is 10 milliseconds.

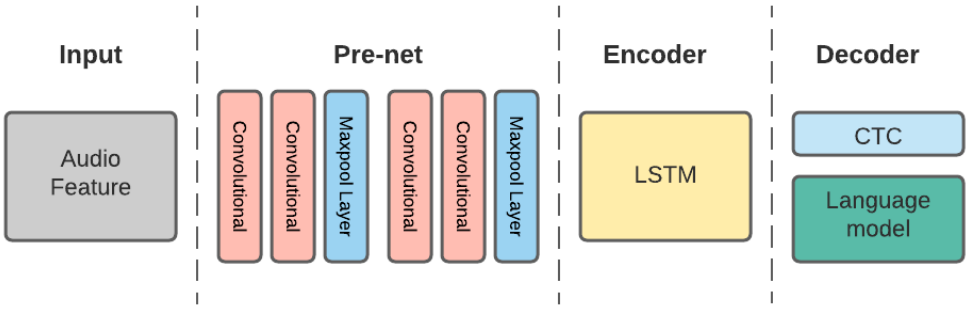


Figure 3.5: Structure of CTC Model

Structure of CTC Model

In our experiment, we implement an encoding network constructed with a recurrent neural network structure. Firstly, the audio features are fed into a pre-input network consisting of a 4-layer CNN and two max-pooling layers. This pre-network is inspired by the VGG structure (C.-Y. Li, Yuan & Lee, 2020). Then the processed features are sent to the encoder, which is composed of three-layer LSTM. The cell dimension of LSTM is 512. The Adadelta algorithm (Zeiler, 2012) is utilised for optimization. The Adadelta optimizer can effectively improve the training speed. Then we add the CTC layer, the CTC layer includes the fully connected layer and the Softmax layer. Finally, in the decoding stage, the trained language model is applied to assist the decoding. The structure is show in Fig. 3.5.

Training Process

The RNN-CTC model employs the same hardware equipped as the HMM-DNN. The parameters and network structure are shown in Table 3.14. The batch size is 16, the beam search width is 10, and a total of 30 epochs are trained. The objective function of CTC is as same as described in Section 3.4.2. However, the decoding stage needs to combine the score of the language model to determine the output result, the score

function of the CTC experiment model is Eq. (3.10), λ_{lm} is the weight of the language model, ranging from 0 to 1. We adjust λ_{lm} to obtain the optimal solution of each datasets.

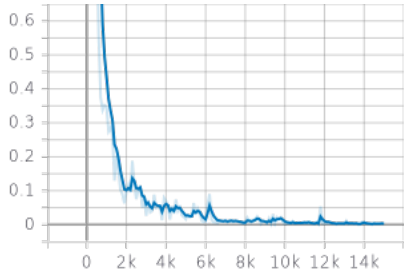
Table 3.14: Parameters of RNN-CTC Model

Parameters	Value
Encoder structure	CNN+LSTM
Encoder LSTM layers	3
Hidden size of LSTM	1024
Batch size	16
Beam search	10
Optimizer	Adadelta
Valid step	500
Language model weight	Between 0 and 1

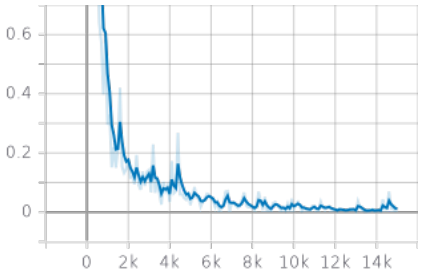
$$Y_{ctc-lm} = \arg\max_{Y}\{\log P_{ctc}(Y \mid X)(1 - \lambda_{lm}) + \lambda_{lm} \log P_{lm}(Y)\} \tag{3.10}$$

We take use of the three datasets to train and validate the CTC model. The indicators of training are Loss and WER. As shown in Fig. 3.6, from left to right, datasets are Alpha, Beta, and Gamma. The training process converges very quickly. If the Chinese training set is at 4,000 steps, the loss is close to zero. From the number of steps required for training, the dataset Gamma needs about 20,000 steps from the start of training to the completion of training. Moreover, the number of training steps in the English dataset is much less, only about 8,000 steps.

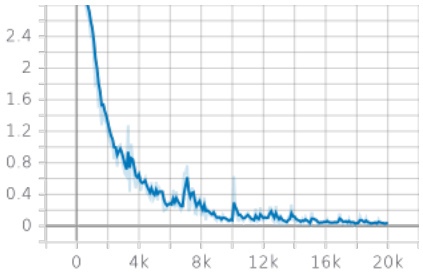
On the other hand, the evaluation standard CER and WER surface are divided into three parts. Amid training, two sets of corresponding WER values will be obtained. The final experimental result WER will be obtained by using the “Test” set for evaluation. Fig. 3.7 is the graph of WER obtained in the two subsets of “Train” and “Dev”. From (a) to (c), datasets are Alpha, Beta, Gamma. The red curve is the WER of the



(a) CTC loss of Alpha



(b) CTC loss of Beta



(c) CTC loss of Gamma

Figure 3.6: CTC Loss of Training

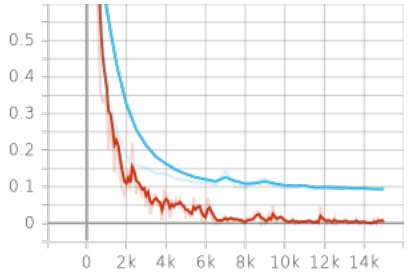
"Train" dataset, the blue curve is the WER of "Dev" dataset. The Chinese speech recognition has almost not pronounced fluctuations after 12,000 steps. The English dataset brought the WER close to zero after 12,000 iterations. Although the Code-Switch dataset has rapidly reduced to 10% from 7,000 steps to 9,000 steps, there is no noticeable improvement in the subsequent learning, which has been fluctuating around 10%.

3.4.6 Experimental Results

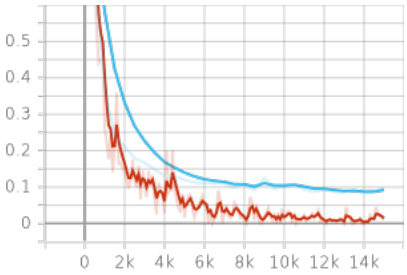
After trained the CTC model, the test results using the "Test" subset of each dataset are shown in Table 3.15.

Table 3.15: Results of CTC Model Experiment

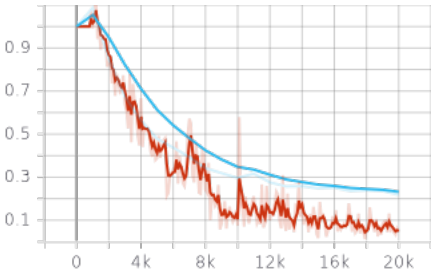
Model	Dataset	CER/WER(%)
RNN-CTC	Alpha(Mandarin)	11.12%
RNN-CTC	Beta(English)	20.51%
RNN-CTC	Gamma(Mandarin-English)	33.16%



(a) CTC WER of Alpha



(b) CTC WER of Beta



(c) CTC WER of Gamma

Figure 3.7: CTC WER of Training

From the test results of the three datasets, the 10-hour noise-free dataset for Mandarin speech training labeled with Pinyin reaches 11.12% CER if RNN-CTC and language models are applied to speech recognition. This is also the best performance in the three datasets. The WER of the model trained by using dataset Gama is 33.16% on the "Test" dataset. In contrast, the WER of the English dataset Beta is between the Alpha and Gamma that achieved 20.51%. From the overall results, the performance of RNN-CTC model in speech recognition based on the three datasets is not as good as that of HMM-TDNN-F.

3.4.7 CTC Summary

In this chapter, we mainly introduce the applications of the CTC algorithm in speech recognition and discus the theoretical basis of using the CTC algorithm based on our low-resource dataset. On the other hand, the design process of the CTC experiment was introduced. The training performance of the three datasets is compared. After

the experiments, it is found that the Chinese dataset labeled with Pinyin has a better training result and convergence speed under this model. In contrast, the model performance is worst in terms of training speed and WER results in the dataset Gamma, which is more complex in processing speech information. These results provide a research cornerstone for subsequent experiments under the CTC/Attention framework.

3.5 Speech Recognition Based on Attention Model

The goal of this chapter is to experiment with the applications of attention mechanism in speech recognition. Aiming at the research goals, we select a variety of attention models for hybrid model experiments, use the three datasets of Alpha, Beta and Gamma to train one of the completed attention models Local-Attention and evaluate the results. The main task is to train the attention model based on different training sets. The result of attention is utilised to compare with the results of the CTC section, we provide the experimental basis for the optimization of the CTC/Attention hybrid model.

3.5.1 Attention Mechanism

There is a description of the most basic and abstract process of attention to better describe the attention model we have chosen. While clarifying the main content and purpose of attention, we introduce the meaning of the essential variables. The attention model in the Seq2Seq framework is regarded as an alignment model between an element in the output Y sequence and each element in the input X sequence. The elements in X are seen as a series of $(Key, Value)$ data pairs. An element in Y is Query. By calculating the similarity or correlation between Query and each Key, the weight coefficient of corresponding Value of each Key is obtained. Then the Value is weighted and summed to get the final attention value.

Namely, the attention mechanism is to perform a weighted summation of the values of the elements in X, Query and Key are employed to calculate the weight coefficients of the corresponding Value. The essential idea is shown as Eq. (3.11), where Q is the query in the output sequence, K is the key in the input sequence, V is the value of the corresponding Key. Q, K, V are obtained by multiplying the input sequence by a matrix. $S(\cdot)$ is a function for calculating similarity or correlation. In the experiment, $S(\cdot)$ needs to be multiplied by Value after the operation of the softmax layer to get the final result.

Therefore, the whole process is divided into three stages. The first stage: Q and K are used as a specific similarity function to calculate the similarity S as Eq. (3.12), the normalized S with $softmax(\cdot)$ to obtain A as shown in Eq. (3.13). In the third stage, we multiply A and V correspondingly and then sum them to obtain the final attention Att as Eq. (3.14).

$$Att(Q_j, X) = \sum_{i=1}^{n} Simi(Q_j, K_i) \times V_i \tag{3.11}$$

$$S_i = Simi(Q, K_i) \tag{3.12}$$

$$A_i = softmax(Simi(Q, K_i)) \tag{3.13}$$

$$Att(Q, X) = \sum_{i=1}^{n} A_i \times V_i \tag{3.14}$$

Dot Product Attention Model (DPA)

The main idea of DPA is to calculate the similar function as a product. We get the connection between Q and K through multiplications. Then we multiply by V to get the final element as the output. Therefore, the utilization of DPA, Eq. (3.14) is further refined into Eq. (3.15).

On the other hand, considering that the input of speech recognition tasks is usually

```
class ScaleDotAttention(BaseAttention):

    def __init__(self, temperature, num_head):
        super().__init__(temperature, num_head)

    def forward(self, q, k, v):
        ts = k.shape[1]

        energy = torch.bmm(q.unsqueeze(1), k.transpose(1, 2)).squeeze(1)

        output, attn = self._attend(energy, v)

        attn = attn.view(-1, self.num_head, ts)

        return output, attn
```

Figure 3.8: Implementation of the SDPA Algorithm

a long sentence, it may increase the dimensions of Q and K, the multiplication will make the value take up too much space. Therefore, we apply the scaled-dot product method to prevent data overflow and improve computing efficiency (Vaswani et al., 2017). Scaled-dot product results from dividing the product of Q and K by using D to reduce the amount of data. D is the dimension number of K. After adding scaling, the dot product attention is recorded as SDPA, and its equation becomes Eq. (3.16). Fig. 3.8 shows the code of the implementation of the SDPA algorithm.

$$Att(Q_j, X) = \sum_{i=1}^{n} softmax(Q_j \times K_i) \times V_i \tag{3.15}$$

$$Att(Q_j, X) = \sum_{i=1}^{n} softmax(\frac{Q_j \times K_i}{\sqrt{|D|}}) \times V_i \tag{3.16}$$

Local Attention Model (Loc-Attention)

Attention model of the comparative experiment in this thesis is called local attention. It is called loc-attention model for short in this thesis. In speech recognition, if the

attention weight of current time step could be considered, it would strengthen the understanding of the context content, improve the recognition consequence (Chorowski, Bahdanau, Serdyuk, Cho & Bengio, 2015).

Global attention and local attention are both mechanisms that achieve the ideal results. The difference between the two is that the algorithms for generating the attention vector are different. Global attention needs to read all hidden states before performing calculations. However, local attention only focuses on a small part of the position information each time the attention vector is calculated. In this way, the calculation overhead is small, which is beneficial to improve efficiency (Luong, Pham & Manning, 2015).

By adding the vectors of the position information to Eq. (3.13), the equation of loc-attention model is obtained. As shown in Eq. (3.17), W is a weight matrix, H is the encoder hidden states. Fig. 3.9 shows the code of the implementation of the loc-attention algorithm.

$$Att(Q_j, X) = \sum_{i=1}^{n} softmax(\tanh(Q_j + K_i + WH_{i-1}) \times V_i \tag{3.17}$$

Multihead Attention

We inspired by previous work (Chorowski et al., 2015), multihead attention model is added to the experimental comparisons. Multihead attention model takes advantage of n different weight matrices to conduct linear transformations of Q, K, and V before the basic attention calculation, denoted as W^q, W^k and W^v. Then we calculate the attention to get the weights of n different angles, which are recorded as a sequence $HEAD$, $HEAD = \{head_1, head_2, \cdots, head_n\}$. The elements in $head$are concatenated and then linearly transformed with a new weight matrix W^0 to get the final attention

```
def forward(self, q, k, v):
    bs_nh, ts, _ = k.shape
    bs = bs_nh // self.num_head

    # Uniformly init prev_att
    if self.prev_att is None:
        self.prev_att = torch.zeros((bs, self.num_head, ts)).to(k.device)
        for idx, sl in enumerate(self.k_len):
            self.prev_att[idx, :, :sl] = 1.0 / sl

    # Calculate location context
    loc_context = torch.tanh(self.loc_proj(self.loc_conv(self.prev_att).transpose(1, 2)))
    loc_context = loc_context.unsqueeze(1).repeat(1, self.num_head, 1, 1).view(-1, ts, self.dim)
    q = q.unsqueeze(1)

    # Compute energy and context
    energy = self.gen_energy(torch.tanh(k + q + loc_context)).squeeze(2)
    output, attn = self._attend(energy, v)
    attn = attn.view(bs, self.num_head, ts)
    self.prev_att = attn

    return output, attn
```

Figure 3.9: Implementation of the Loc-Attention Algorithm

output.

Therefore, combined with the basic attention calculation equation before, multi-head attention model is expressed as Eq. (3.18), Eq. (3.19) and Eq. (3.20). We see from the equations that multihead attention model needs to cooperate with the previous two attention mechanisms to get the final output. We divide the multihead attention model into scaled-dot product multihead attention model and local multihead attention model. The corresponding models are abbreviated as SDPMHA and LMHA, respectively.

$$head_i = Att(QW_i^q, KW_i^k, VW_i^v) \tag{3.18}$$

$$HEAD = \{head_1, head_2....head_n\} \tag{3.19}$$

$$MH - Attention(Q, K, V) = concatenate(HEAD)W^0 \tag{3.20}$$

3.5.2 Our Experiments

In the experiments, the same training and evaluation datasets as the CTC experiment were taken. The composition of the framework of the experimental model is described in detail. The primary purpose of the attention experiment is to obtain the training and evaluation results of each attention model. We provide experimental data for the training optimization of the hybrid framework, comparing and analyzing the performance of four kinds of attention models based on three datasets.

Data and Feature

Our experiment utilizes three datasets that we designed. These three datasets are described in detail in Section 3.2.2. In order to obtain better training results based on low-resource datasets, we apply data augmentation by applying SpecAugment to enhance the features. The processed features include 40D Fbank and 3D Pitch with 25 milliseconds frame window and ten milliseconds shift length to process the frames of audio waveform.

Attention Model Composition

The attention model for speech recognition includes the module encoder and decoder. Because of the consideration of better retention of the information in the context sentence and the complexity reduction of the hybrid model, the coding network of the attention speech recognition takes use of the same LSTMs as the CTC. Among the known types of attention, in the design of this experiment, we mainly analyze and compare four attention mode, we select the local-attention model as the attention part. Finally, the speech model applies the same language model as CTC, we introduce λ_{lm} as the weight parameter of language model. The structure is shown in Fig. 3.10.

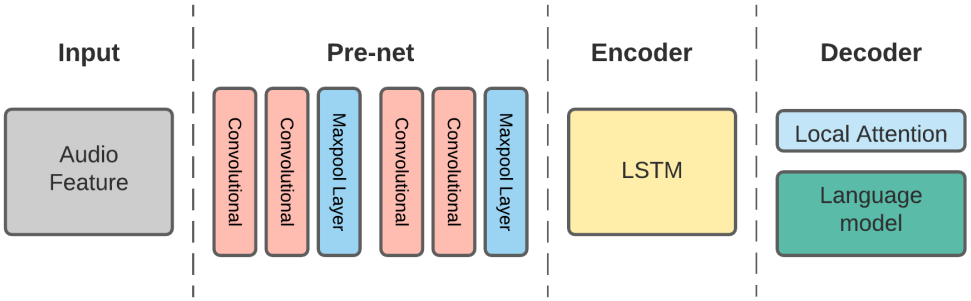


Figure 3.10: The Architecture of Attention Model

Training Process

The training attention model employs the same hardware equipment as the training HMM-DNN. The training parameters and network structure are shown in Table 3.16. We chose SDPA, loc-attention and LMHA as the experimental objects according to the convergence performance of the model during the experiment. SDPMHA will be experimented in a mixed model. The dimension of attention is 300. During local attention training, the attention convolution filter kernel size is 100, and the number of channels is 10. Because of the limitation of graphics card memory in the training device, the number of heads during multihead attention training is 2. The score function of the attention experiment model amid the training is Eq. (3.21). In the equation, X represents the input voice feature vector, and t represents the time step. y^*_{t-1} represents the previous ground truth. During the entire training process, every prediction output y_t is based on the ground truth sequence from time step 1 to $t-1$ of y $(y^*_1, \ldots, y^*_{t-1})$.

However, in the decoding stage, because there is no ground truth while outputting y_t, it is based on the prediction sequence of the model $(y_1, \ldots, y_{t-1})$. This process is expressed as Eq. (3.22). If this model is decoded, the predicted label should also be combined with the attention model score and the language model score. The score function is described as Eq. (3.23). Moreover, like CTC, λ_{lm} is the weight of the

speech model, and its value is between 0 and 1.0.

Table 3.16: Parameters of Training Base on Attention

Parameters	Value
Attention type	SDPA , Loc-attention and LMHA
Attention dimension	300
Loc-Attention kernel	10
Loc-Attention kernel size	100
Encoder structure	CNN+LSTM
Encoder LSTM layers	3
Hidden size of LSTM	1024
Decoder structure	LSTM
Batch size	16
Scaling factor	0.5
Optimizer	Adadelta
Valid step	1,000

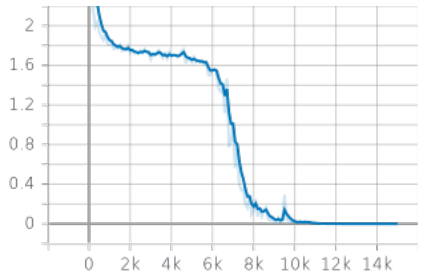
$$\prod_{t=1}^{T} P\left(y_t \mid y_1^*, \ldots, y_{t-1}^*, X\right) \triangleq P_{att}^*(Y \mid X) \tag{3.21}$$

$$P_{att}(Y \mid X) = \prod_{t=1}^{T} P\left(y_t \mid y_1, \ldots, y_{t-1}, X\right) \tag{3.22}$$

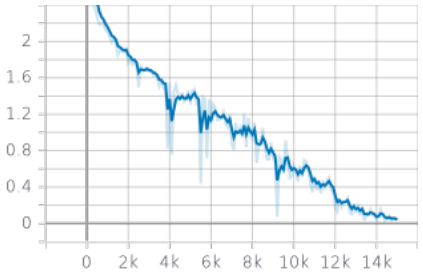
$$Y_{att-lm} = \arg\max_{Y} \{\log P_{att}(Y \mid X)(1 - \lambda_{lm}) + \lambda_{lm} \log P_{lm}(Y)\} \tag{3.23}$$

As shown in Fig. 3.11, the loss graphs of the loc-attention model trained on Alpha, Beta and Gamma datasets are illustrated. Fig. 3.11(a) shows the curve of the dataset Alpha training. From the perspective of the curve drop, the curve drops significantly after 6,000 steps of iterations. The loss dropped from 1.6 to 0.2 in 2,000 steps. The training situation of the dataset Beta is shown in Fig. 3.11(b).

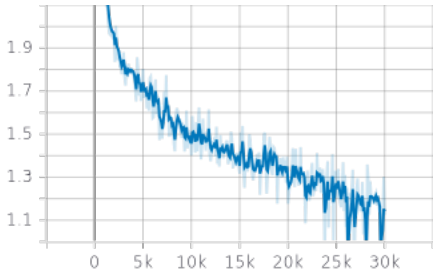
Although the loss curve drops faster in the early stage than in Fig. 3.11(a), there is no apparent acceleration in the later stage. The speed of convergence is also significantly slower than Alpha. On the other hand, the declining trend of loss in the dataset Gamma training is far less than that in the first two charts. After learning through 30 epochs, the loss remains over 1.



(a) Loss on Alpha



(b) Loss on Beta



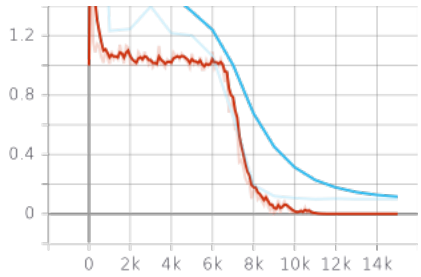
(c) Loss on Gamma

Figure 3.11: Loc-Attention Loss of Training

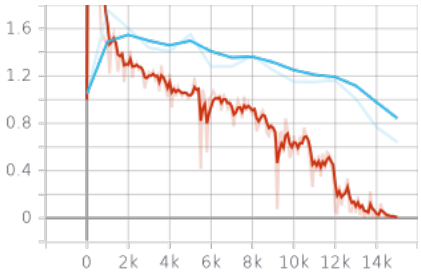
The graphs of CER and WER for training and evaluation of the three datasets are shown in Fig. 3.12. The blue curve in the figure is the CER and WER evaluation curve of the loc-attention model on the corresponding "Dev" subset during the training process. The red curve is the performance of the training set. Fig. 3.12(a) shows that if the dataset Alpha is used for training, the CER value decreases significantly as the training epoch increases. The SER value of the training set (red curve) has remained stable at 0 at about 11,000 steps. It shows that the model has completed the learning task on the training set at this time.

Although the dataset Beta completed the learning task after 14,000 steps, from the evaluation of the "Dev" subset, the WER is as high as 0.8. Finally, the WER in Fig. 3.12(c) shows no significant drop in either the training set or the "Dev" subset.

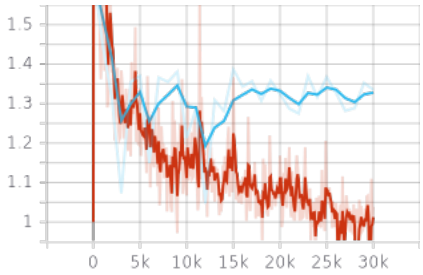
The nine subfigures in Fig. 3.13 ultimately show the alignment process of the loc-attention model amid training. These nine subfigures all come from the loc-attention model based on the dataset Alpha. From the first four subfigures, we see that the alignment of attention has not been changed significantly. However, starting from the



(a) CER of Alpha



(b) WER of Beta



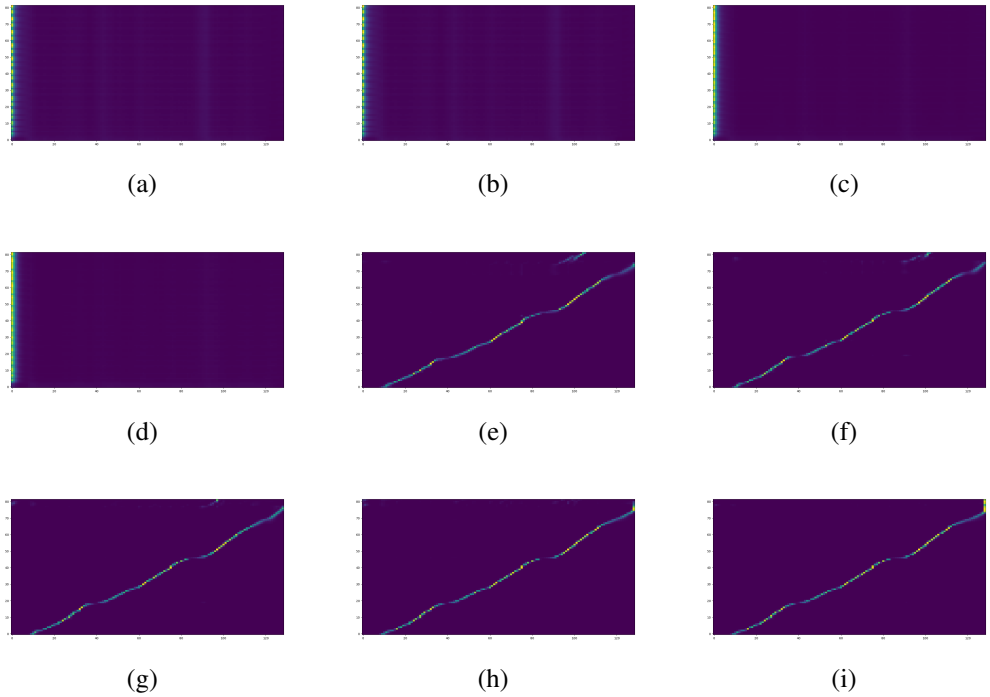
(c) WER of Gamma

Figure 3.12: Loc-Attention WER of Training

fifth subfigure, the alignment process is gradually transparent. It is evident from the color that the attention is getting deeper and deeper.

Fig. 3.14 is the training loss graph of the SDPA model. By considering the slow convergence of dataset Gamma in the single attention model during the loc-attention model training, dataset Gamma was not used in SDPA model training. Fig. 3.14(a) is the training curve on the dataset Alpha. From 3,000 steps to 6,000 steps, loss dropped rapidly from 1.6 to 0.2. After 11,000 steps, the learning process was completed.

In contrast, the training of dataset Beta has the problem of slow convergence. On the other hand, from the alignment process, SDPA has a situation where attention skips frames. Fig. 3.15 is the alignment process of the SDPA model based on the Alpha training dataset. In summary, if we use the "Test" subset to verify the attention model, we only verify the three models of loc-attention and the SDPA model trained on dataset Alpha.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

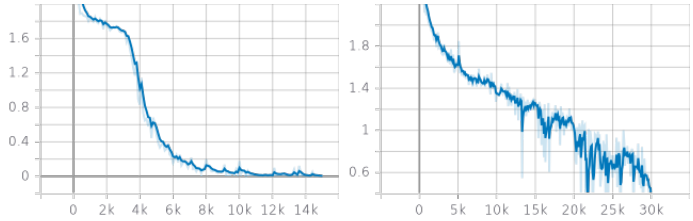
Figure 3.13: Loc-Attention Alignment Process on Dataset Alpha

3.5.3 Experiment Results

After training the four attention model, the test results based on the "Test" subset of each dataset are shown in Table 3.17. From these four experiment results, we eye that when only attention and language models are used to complete the speech recognition task, the effect is not good. Both Beta and Gamma trained models failed to meet the recognition task requirements. Their WERs is far greater than the requirements of speech recognition. Even on dataset Alpha, which performed the best in speech recognition tasks applying CTC, character error rate of 13.79% was achieved. However, this value is already the best of the four models.

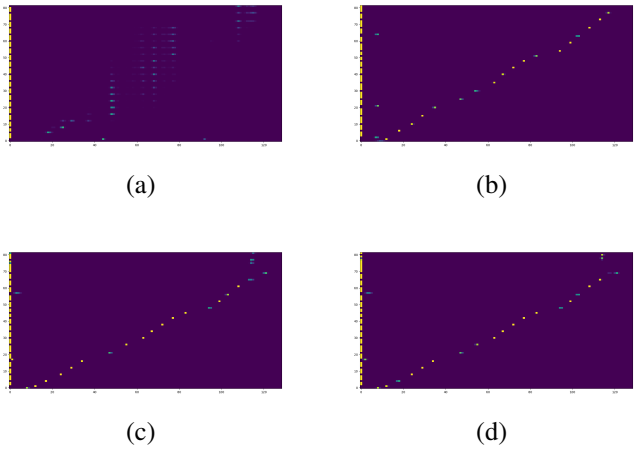
3.5.4 Attention Summary

In the experimental design based on attention speech recognition, we obtained four attention models for speech recognition by combining the three attention mechanisms.



(a) SDPA loss of Alpha (b) SDPA loss of Beta

Figure 3.14: SDPA Loss of Training



(a) (b)

(c) (d)

Figure 3.15: SDPA Alignment

Two models were tested on three datasets. The training process and experimental results are provided in details. The results did not meet speech recognition requirements, which also prompted us to utilise the CTC and attention mechanism hybrid model for speech recognition experiments.

3.6 Speech Recognition Based on CTC/Attention Hybrid

The scoring function of the hybrid structure model applies the advantages of the attention-based mechanism to evaluate the alignment of long sentences and employs the CTC model to force the voice and the corresponding text to be monotonously aligned. The

Table 3.17: Results of Attention Experiment

Model	Dataset	CER/WER
Loc-Attention	Alpha(Mandarin)	13.79%
Loc-Attention	Beta(English)	93.98%
Loc-Attention	Gamma(Mandarin-English)	127.43%
SDPA	Alpha(Mandarin)	24.11%

forward-backwards algorithm is implemented in CTC to speed up the process of processing this monotonous alignment.

3.6.1 Structure of Hybrid Model

The structure of the CTC/Attention hybrid model is divided into three main parts. The first part is pre-net, which is composed of Deep CNN inspired by the VGG structure. The second part is the encoder shared by CTC and Attention. The last part is a joint decoder, which is composed of a CTC decoder, an attention decoder, and a language model. The architecture of the CTC/Attention hybrid model is shown in Fig. 3.16.

3.6.2 Score Function of CTC/Attention Hybrid

The main idea of the CTC/Attention hybrid model is to utilise CTC to force the alignment of the eigenvectors of the audio frame to reduce a single tag which has been used for many times during decoding (Petridis, Stafylakis, Ma, Tzimiropoulos & Pantic, 2018). At the same time, CTC does not allow skipping label output under the same audio characteristics to avoid the frame skipping mentioned in the attention subsection. The score function of the hybrid model is described by using Eq. (3.24) based on the schematic diagram of the combination of CTC and attention formulas. Among them, O_{hybrid} is the prediction result of the model, Y is the text label sequence, X is the feature vector sequence corresponding to the audio frame. λ is the evaluation weight of the CTC model, $\log P_{ctc}(Y \mid X)$ is the score function of CTC, and $\log P_{att}(Y \mid X)$

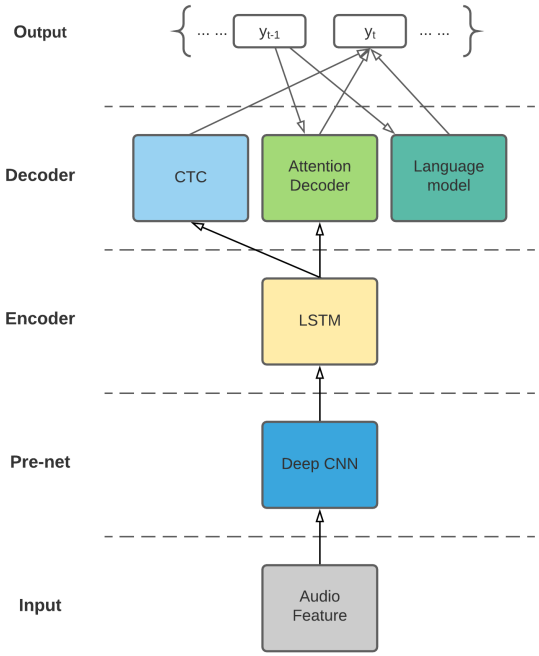


Figure 3.16: Structure of Hybrid Model

is the score function of attention.

$$O_{hybrid} = \log P_{ctc}(Y \mid X)\lambda + \log P_{att}(Y \mid X)(1 - \lambda) \tag{3.24}$$

The final output result of the CTC/Attention hybrid model needs to substitute the predictions of $\log P_{ctc}(Y \mid X)$, $\log P_{att}(Y \mid X)$ and the language model into the Beam search algorithm, calculate the optimal predictive text sequence. The final objective function is described by using Eq. (3.25). Y_{hybrid} is the final predicted text output by the hybrid model.

$$Y_{hybrid} = \arg\max_{Y}\{\log P_{ctc}(Y \mid X)\lambda + \log P_{att}(Y \mid X)(1 - \lambda) + \log P_{lm}\lambda_{lm}\} \tag{3.25}$$

3.6.3 Experiment

Experiment Environment and Setup

The hardware environment of this experiment is as same as the previous experiment. The implementation code uses the PyTorch deep learning framework to build a CTC-Attention hybrid experimental model based on LAS speech recognition. The main hyperparameters employed in the experimental model are shown in Table 3.18

Table 3.18: Parameters of CTC/Attention Model

Parameters	Value
Attention type	SDPA , Loc-attention, Multihead
Attention dimension	300
Loc-attention kernel	10
Loc-attention kernel size	100
Encoder structure	CNN+LSTM
Encoder LSTM layers	3
Hidden size of LSTM	1024
Decoder structure	LSTM
Decoder LSTM layers	2
Batch size	32
Scaling factor	0.5
Optimizer	Adadelta
Valid step	1,000
CTC weight	0.2
Decode beam size	8

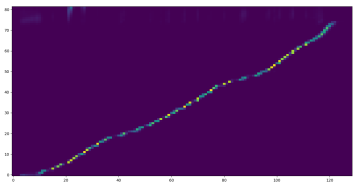
Multitask Learning

During the experiments, we applied multitask learning to train the CTC-attention hybrid model (Kim et al., 2017)(Qin, Zhang & Qu, 2019). Our focus is on a single model which may ignore the potential information that could improve the target task. By sharing the model parameters, the original task will be generalized. In our previous experiments, the training of CTC is faster than attention models in the experiment. In this case, multitask learning is more contributory to the attention model. It can improve

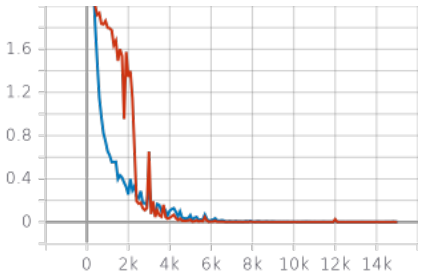
learning efficiency.

Optimization of the Training Process

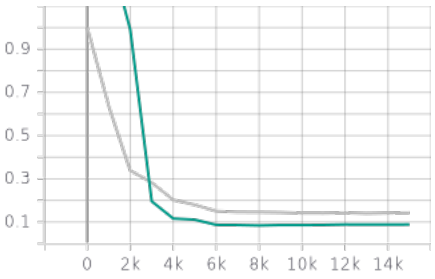
The parameters in our experiment are offered as the initial training parameters of the CTC/Attention model for conducting the first experiment. By adjusting only one parameter at a time, the influence of the parameters based on the model is explored to find the best combination of parameters. For the efficiency of the experiment, we summarize the previous independent experiments of CTC and Attention. We choose the faster convergence Alpha dataset as the tuning dataset. The tuning process will be tested from the two perspectives of the CTC weight and attention model. The first is the weight of CTC. We gradually increase the weight of CTC from 0.2 and obtain the optimal solution of CTC weight by evaluating the loss of training and the "Dev" set.



(a) Alignment



(b) Loss

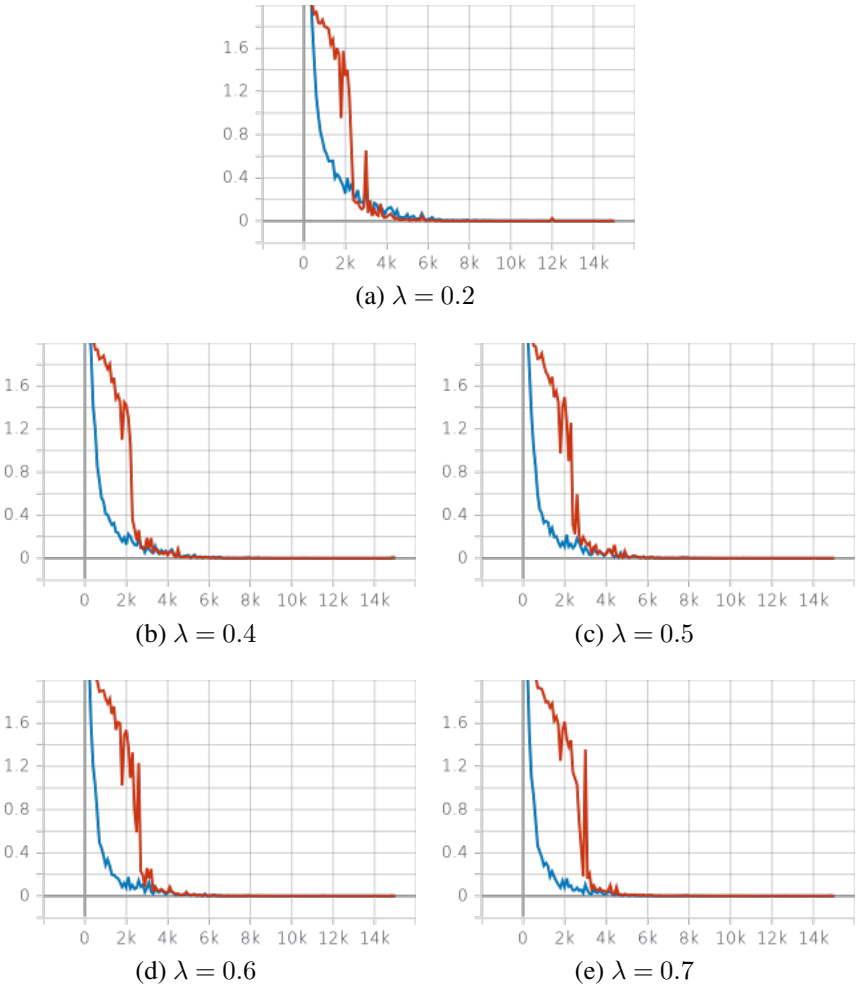


(c) Character Error Rate

Figure 3.17: Training with The CTC Weight of 0.2

Fig. 3.17 is the training result of the hybrid model if the CTC weight is 0.2. Fig. 3.17 (a), (b) and (c) are the alignment, loss graph, and CER graph. The red line is the loss of the attention model, the blue line is the CTC. In the CER diagram, the

green line is attention, and the grey line is CTC. From Fig. 3.17(b), compared with the case of attention model shown in Fig. 3.17(a), the loss in the hybrid model converges faster. From the previous 10,000 steps, the completion of project has been improved to 6,000 steps. On the other hand, though the CTC convergence rate has not changes significantly compared with the previous single training, it is more stable in the later training stage.



(a) $\lambda = 0.2$

(b) $\lambda = 0.4$

(c) $\lambda = 0.5$

(d) $\lambda = 0.6$

(e) $\lambda = 0.7$

Figure 3.18: The Comparisons of CER Based on "Dev" Subset with Various CTC Weights

In the experiment, five model were conducted for λ from 0.2 to 0.7. The loss is shown in Fig. 3.18. By judging from the loss of CTC, the convergence speed of the five experiments has been improved. While CTC improves the learning efficiency, the

convergence of attention does show relatively large fluctuations. If λ equals to 0.7, the loss of attention has a huge fluctuation in a short period.

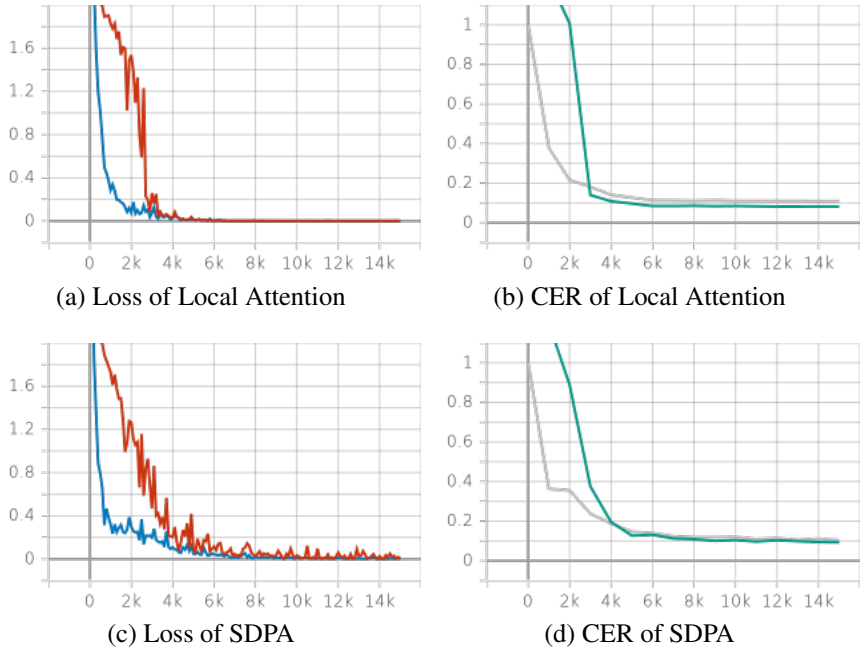
On the other hand, the CERs of the five experiments based on the "Dev" subset are listed in Table 3.19. If λ is less than 0.6, the CER obtained in the task with the increase of λ becomes smaller, from 0.0885 to 0.0832. However, if λ is equal to 0.6, CER starts increasing. Unlike the slight decrease in attention model, the performance of CTC on CER dropped from 0.1428 to 0.1104 as λ increases. The minima is reached if λ is equal to 0.6. Although λ is equal to 0.5, attention model has achieved the best results. Nevertheless, compared to the slight difference in attention, if λ is equal to 0.6, the CTC improvement is more remarkable. Combining the experimental results in Fig. 3.18 and Table 3.19, we see that λ equals 0.6 in the mixed model is the optimal solution.

Table 3.19: The Comparison of CER on "Dev" Subset with Different CTC Weight

λ	CER(Attention)	CER(CTC)
0.2	0.0885	0.1428
0.4	0.0852	0.1284
0.5	**0.0824**	0.1202
0.6	0.0832	**0.1104**
0.7	0.0851	0.1117

The second optimization solution is implemented to replace different types of attention model. For this reason, we take use of the four attention models to replace the local attention model in the current hybrid model. We design two comparable tests to achieve this goal. The first test aims to compare the local attention model and the SDPA model. We replace the attention module with SDPA. λ is 0.6 which is obtained from the optimal experiment, and other parameters remain unchanged. Fig. 3.19 is obtained after training. In Fig. 3.19 (a) and (b) are the local attention based on loss and CER. (c) and (d) are the performance of SDPA in these two evaluations. The convergence speed has been slightly promoted. However, whether it is the stability of learning or

the score on CER, the local attention model is even better.



(a) Loss of Local Attention (b) CER of Local Attention

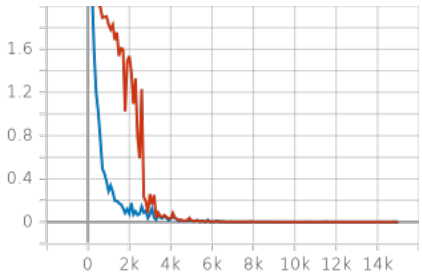
(c) Loss of SDPA (d) CER of SDPA

Figure 3.19: The Comparison of Attention Models

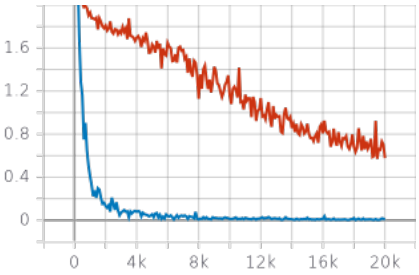
The second test is to compare whether the multihead attention model can achieve better results. The local attention performed better in the first test. The local attention in the hybrid model is replaced with SDPMHA and LMHA. Fig. 3.20 shows the loss of the three attention models. Moreover, the evaluation results of the three models on the "Dev" subset are listed in Table 3.20.

Although the local attention model is trained faster than LMHA from the loss figure, the LMHA model has achieved better overall results as shown in Table 3.20. It achieves 9.19% and 8.62% character error rates in attention model and CTC, respectively. Therefore, LMHA is considered to be the optimal solution among the four attention mechanisms.

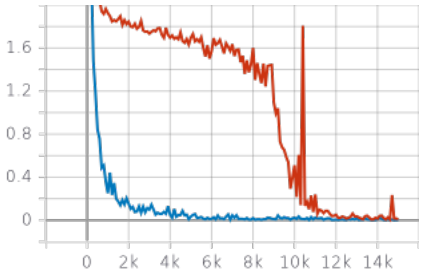
The final parameters in the multilanguage experiment are shown in Table 3.21.



(a) Loss of Local Attention



(b) Loss of SDPMHA



(c) Loss of LMHA

Figure 3.20: The Comparison of Multihead Attention Models

Table 3.20: The Comparison of Different Multihead Attention Models

Model	CER(Attention)	CER(CTC)
Local Attention	**0.0832**	0.1104
SDPMHA	0.8436	0.0961
LMHA	0.0919	**0.0862**

3.6.4 Experimental Results

We employ the data in Table 3.21 as the parameters for the hybrid model. Training and evaluation were carried out based on the three small datasets of Alpha, Beta, and Gamma, respectively. The final experimental results and the previous three experimental results are compared in Table 3.22.

Table 3.21: Parameters of CTC/Attention Model

Parameters	Value
Attention type	LMHA
Attention dimension	300
Loc-Attention kernel	10
Loc-Attention kernel size	100
Encoder structure	CNN+LSTM
Encoder LSTM layers	3
Hidden size of LSTM	1024
Decoder structure	LSTM
Decoder LSTM layers	2
Batch size	32
Scaling factor	0.5
Optimizer	Adadelta
Valid step	1,000
CTC weight	0.6
Decode beam size	8

Table 3.22: Comparison of Experimental Results(CER/WER)

Model	Alpha	Beta	Gamma
HMM-TDNN-F	10.91%	**18.23%**	**25.62%**
RNN-CTC	11.12%	20.51%	33.16%
CTC/Attention	**10.22%**	19.05%	26.11%

3.7 Summary

In summary, in this chapter, we describe the data collection, experimental design, and experimental results. In the data collection, the language features and the corpus structure are detailed. The composition and production process of the three low-resource corpora are described in detail. In practice, we have set up four sets of experiments to complete the comparison between the traditional model HMM-DNN and the CTC/Attention hybrid model based on a low-resource corpus. Each group of experiments detailed the experimental process and results of the model. We designed independent CTC and attention model to illustrate the working principle of the CTC/Attention hybrid. Finally, it plays an essential role in optimizing the CTC/Attention

hybrid model experiment. Our experiments show that the application of the local multihead attention model as the hybrid model can get the optimal solution, if the CTC weight is equal to 0.6. Finally, in this chapter, we summary all the verification results to provide data support for the analysis.

Chapter 4

Result Analysis

The experimental results will be analyzed and discussed in this chapter. Based on methodology and experimental results, three research questions are discussed. Moreover, its limitations and contributions are described.

4.1 Introduction

This chapter will analyze the experimental results from two aspects. One is to compare the performance of each experimental model from a different perspective of the framework. On the other hand, from different languages, we compare which language has a better recognition rate on small dataset. Finally, the limitations of this work are explained by analyzing the experimental results.

4.2 Comparison of Speech Recognition Frameworks

The experiments related to CTC and attention model demonstrate the advantages and disadvantages of the two models and also provide experimental data for optimizing the hybrid model. Throughout comparative experiments with various parameters, we concluded that if the CTC weight is 0.6, the overall output score of the model is the highest one.

The utilization of multi-head local attention improved the prediction accuracy of the entire model. From the perspective of the training process, the hybrid model of CTC and attention module has improved the efficiency and effectiveness of the training process. The experimental results show that the hybrid model obtained similar results to the traditional model in the three small datasets.

The first row in Table 3.22 shows that the traditional HMM-based speech recognition model achieved 10.91% error rate in Chinese and 18.23% in English. These two items are much higher than only implementing CTC or attention model in speech recognition. However, the CTC/Attention hybrid model obtained better results based on the Chinese dataset, with the error rate 0.069% lower than that of the traditional model. On the Code-Switch dataset,the result of the traditional speech recognition model of word error is the best of the three models, but the difference with CTC/Attention is

not obvious, and the obvious effect is improved by 3%. The evaluation results of all models in the Gamma dataset are worse than Alpha and Beta. That is owing to the complexity of mixed languages.

In contrast, there is a significant gap in the performance if the attention model works alone. The speech recognition was not completed based on the low-resource English and Code-Switch datasets, and the word error rate exceeded 90%. Moreover, through the experimental results, we see that the accuracy of using LMHA in the CTC/Attention hybrid model is the highest one, an error rate 10.22% was achieved based on the Chinese dataset.

4.3 Recognition Rate of Various Languages

From the language perspective, the experimental results show that the English dataset is more challenging to be used for model training than the Chinese dataset. In comparing the three languages, Chinese labeled with Pinyin performs better than English in speech recognition in both training and evaluation. The convergence in the training phase and the WERs in the verification phase consistently reflect that the Chinese dataset Alpha labeled with Pinyin is easier to be trained, the recognition accuracy is higher than English. The utilization of Pinyin effectively avoids the situation of one sound and multiple characters in Chinese. The three questions of this research have been answered based on the analysis of experimental results and data.

On the other hand, the WER of the HMM-TDNN-F model in the Gamma test dataset is 25.62%, 40% higher than that in the beta dataset. In contrast, the CTC/Attention model has a 37% gap between the two datasets. The results show that the difference in recognition accuracy of the CTC/Attention model in different language datasets is smaller. The difference of multi-language recognition is better than the traditional framework.

4.4 Contributions and Limitations

Throughout this thesis, we provide a research foundation for future exploration of speech recognition in different languages. If the corpus is insufficient, researchers can generate the corresponding corpus through our design method. The comparison of actual effects in different languages provides a reference for future research on the characteristics of speech recognition.

On the other hand, there are also limitations to this project. During the process of optimization, only three parameters are debugged. Moreover, the data utilised in this thesis only includes two languages and one code switch language environment.

4.5 Summary

In this chapter, we analyse the experimental results from two aspects and proposes a solution to optimize the model parameters. In contrast, CTC / attention model performs better in the Chinese dataset, while the traditional model performs better in English and mixed language. In terms of language, Pinyin labeled datasets are easier to be applied to model training. CTC / attention has less difference between different datasets in terms of the accuracy of different language recognition. Finally, contributions and limitations of this project are discussed.

Chapter 5

Conclusion

In this chapter, we will summarize the contributions of this thesis and provide the conclusion as well as envision our future work.

5.1 Conclusion

In this thesis, we investigate the speech recognition performance based on the CTC and attention hybrid model and HMM-TDNN-F model by using three small labeled datasets. Comparing with the performance of recognition task on different language datasets, we explore which language is more suitable for the speech recognition. In this research project, we create two labelled datasets and one low-resource Code-Switch corpus based on the labeled text files of the existing corpus and the iFlytek InterPhonic toolkit.

In terms of experimental comparison of models, the Kaldi speech recognition toolbox was employed to evaluate the performance of the HMM-TDNN-F model based on our datasets. The model achieves 10.91%, 18.23%, and 25.62% error rates on the Chinese, English, and Chinese-English Code-Switch datasets.

Furthermore, we have implemented a hybrid model of CTC and attention model based on PyTorch and ESPnet speech processing tools. By comparing the performance of the model with different CTC weights, the optimal CTC weight of the model is obtained as 0.6. The model adopts the optimal solution parameters to complete training and evaluation on our three datasets and achieves similar performance to the traditional model. The CER evaluation based on the Chinese dataset defeated the traditional model and reached 10.22%. Although the performance on the English and Code-Switch datasets was not as good as the traditional model, the gap remained within 3%.

5.2 Future Work

In future, more applications of the attention model will be added to our project to replace the current part of the recurrent neural network. We will implement the predetermined sample algorithm to improve the training effect. In addition, we will explore more languages for speech recognition by using the method of creating datasets (W. Q. Yan, 2021)(W. Q. Yan, 2019).

References

Abdel-Hamid, O. & Jiang, H. (2013). Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 7942–7946).

Abdel-Hamid, O., Mohamed, A.-R., Jiang, H. & Penn, G. (2012). Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 4277–4280).

Bacchiani, M., Senior, A. & Heigold, G. (2014). Asynchronous, online, GMM-free training of a context dependent acoustic model for speech recognition. In *Annual Conference of the International Speech Communication Association.*

Bellegarda, J. R. (1998). A multispan language modeling framework for large vocabulary speech recognition. *IEEE Transactions on Speech and Audio processing*, *6*(5), 456–467.

Bengio, Y., Ducharme, R., Vincent, P. & Janvin, C. (2003). A neural probabilistic language model. *The Journal of Machine Learning Research*, *3*, 1137–1155.

Bengio, Y., Simard, P. & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, *5*(2), 157–166.

Biswas, A., De Wet, F., van der Westhuizen, E. & Niesler, T. R. (2020). Semi-supervised acoustic and language model training for English-isiZulu code-switched speech recognition. *arXiv preprint arXiv:2004.04054*.

Biswas, A., Menon, R., van der Westhuizen, E. & Niesler, T. (2019). Improved low-resource Somali speech recognition by semi-supervised acoustic and language model training. *arXiv preprint arXiv:1907.03064*.

Boden, M. (2002). A guide to recurrent neural networks and backpropagation. *the Dallas project*.

Chan, J. Y., Cao, H., Ching, P. & Lee, T. (2009). Automatic recognition of Cantonese-English code-mixing speech. In *International Journal of Computational Linguistics & Chinese Language Processing* (Vol. 14).

Chan, J. Y., Ching, P., Lee, T. & Meng, H. M. (2004). Detection of language boundary in code-switching utterances by bi-phone probabilities. In *International Symposium on Chinese Spoken Language Processing* (pp. 293–296).

Chan, W., Jaitly, N., Le, Q. & Vinyals, O. (2016). Listen, attend and spell: A neural

network for large vocabulary conversational speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 4960–4964).

Chan, W., Jaitly, N., Le, Q. V. & Vinyals, O. (2015). Listen, attend and spell. *arXiv preprint arXiv:1508.01211*.

Chen, S. F. & Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, *13*(4), 359–394.

Chen, Z., Deng, W., Xu, T. & Yu, K. (2016). Phone synchronous decoding with CTC lattice. In *Interspeech* (pp. 1923–1927).

Chiu, C.-C. & Raffel, C. (2017). Monotonic chunkwise attention. *arXiv preprint arXiv:1712.05382*.

Chorowski, J., Bahdanau, D., Serdyuk, D., Cho, K. & Bengio, Y. (2015). Attention-based models for speech recognition. *arXiv preprint arXiv:1506.07503*.

Cui, X., Zhang, W., Finkler, U., Saon, G., Picheny, M. & Kung, D. (2020). Distributed training of deep neural network acoustic models for automatic speech recognition: A comparison of current training strategies. *IEEE Signal Processing Magazine*, *37*(3), 39–49.

Davis, S. & Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *28*(4), 357–366.

Dong Wang, Z. Z., Xuewei Zhang. (2015). *THCHS-30: A free Chinese speech corpus.* Retrieved from `http://arxiv.org/abs/1512.01882`

Drexler, J. & Glass, J. (2019). Subword regularization and beam search decoding for end-to-end automatic speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 6266–6270).

Eyben, F., Wöllmer, M., Schuller, B. & Graves, A. (2009). From speech to letters-using a novel neural network architecture for grapheme based ASR. In *IEEE Workshop on Automatic Speech Recognition & Understanding* (pp. 376–380).

Fathima, N., Patel, T., Mahima, C. & Iyengar, A. (2018). TDNN-based multilingual speech recognition system for low resource Indian languages. In *Interspeech* (pp. 3197–3201).

Fu, L., Li, X. & Zi, L. (2020). Research on modeling units of transformer transducer for Mandarin speech recognition. *arXiv preprint arXiv:2004.13522*.

Georgescu, A.-L. & Cucu, H. (2018). Automatic annotation of speech corpora using complementary GMM and DNN acoustic models. In *International Conference on Telecommunications and Signal Processing* (pp. 1–4).

Georgescu, A.-L., Cucu, H. & Burileanu, C. (2019). Kaldi-based DNN architectures for speech recognition in Romanian. In *International Conference on Speech Technology and Human-Computer Dialogue (SpeD)* (pp. 1–6).

Goussard, G. & Niesler, T. (2010). Automatic discovery of subword units and pronunciations for automatic speech recognition using timit. *PRASA*.

Graves, A., Fernández, S., Gomez, F. & Schmidhuber, J. (2006). Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *International Conference on Machine Learning* (pp. 369–376).

Gupta, K. & Gupta, D. (2016). An analysis on LPC, RASTA and MFCC techniques in automatic speech recognition system. In *International Conference-Cloud System and Big Data Engineering* (pp. 493–497).

Hadian, H., Sameti, H., Povey, D. & Khudanpur, S. (2018). Flat-start single-stage discriminatively trained HMM-based models for ASR. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *26*(11), 1949–1961.

He, Y., Sainath, T. N., Prabhavalkar, R., McGraw, I., Alvarez, R., Zhao, D., ... Pang, R. (2019). Streaming end-to-end speech recognition for mobile devices. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 6381–6385).

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

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

Hsiao, R., Fuhs, M., Tam, Y.-C., Jin, Q. & Schultz, T. (2008). The CMU-InterACT 2008 Mandarin transcription system. In *Annual Conference of the International Speech Communication Association.*

Huang, L., Xu, J., Sun, J. & Yang, Y. (2017). An improved residual LSTM architecture for acoustic modeling. In *International Conference on Computer and Communication Systems* (pp. 101–105).

Huang, Y., Yu, D., Gong, Y. & Liu, C. (2013). Semi-supervised GMM and DNN acoustic model training with multi-system combination and confidence re-calibration. In *Interspeech* (pp. 2360–2364).

Huang, Z., Siniscalchi, S. M. & Lee, C.-H. (2016). Bayesian unsupervised batch and online speaker adaptation of activation function parameters in deep models for automatic speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *25*(1), 64–75.

Ibrahim, Y. A., Odiketa, J. C. & Ibiyemi, T. S. (2017). Preprocessing technique in automatic speech recognition for human computer interaction:: An overview. *Annals. Computer Science Series*, *15*(1).

iFLYTEK Co., Ltd. (2020). *Online TTS WebAPI.* Website. (`https://global.xfyun.cn/products/online_tts`)

Imseng, D., Motlicek, P., Garner, P. N. & Bourlard, H. (2013). Impact of deep MLP architecture on different acoustic modeling techniques for under-resourced speech recognition. In *IEEE Workshop on Automatic Speech Recognition and Understanding* (pp. 332–337).

Jason, C. A. & Kumar, S. (2020). An appraisal on speech and emotion recognition technologies based on machine learning. *Language*, *67*, 68.

Ji, M., Zhang, L. & Wu, Q. (2020). Automatic grape leaf diseases identification via unitedmodel based on multiple convolutional neural networks. *Information Processing in Agriculture*, *7*(3), 418–426.

Kadyan, V., Mantri, A. & Aggarwal, R. (2020). Improved filter bank on multitaper framework for robust Punjabi-ASR system. *International Journal of Speech*

Technology, *23*(1), 87–100.

Kanda, N., Takeda, R. & Obuchi, Y. (2013). Elastic spectral distortion for low resource speech recognition with deep neural networks. In *IEEE Workshop on Automatic Speech Recognition and Understanding* (pp. 309–314).

Kim, S., Hori, T. & Watanabe, S. (2017). Joint CTC-attention based end-to-end speech recognition using multi-task learning. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 4835–4839).

learning precise timing with lstm recurrent networks. (n.d.).

LeCun, Y., Bengio, Y. & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.

Li, B., Chang, S.-y., Sainath, T. N., Pang, R., He, Y., Strohman, T. & Wu, Y. (2020). Towards fast and accurate streaming end-to-end ASR. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 6069–6073).

Li, C., Chen, J. & Xu, B. (1999). Regression class selection and speaker adaptation with MLLR in Mandarin continuous speech recognition. In *European Conference on Speech Communication and Technology*.

Li, C.-Y., Yuan, P.-C. & Lee, H.-Y. (2020). What does a network layer hear? analyzing hidden representations of end-to-end ASR through speech synthesis. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 6434–6438).

Li, J., Shan, Y., Wang, X. & Li, Y. (2018). Improving gated recurrent unit based acoustic modeling with batch normalization and enlarged context. In *International Symposium on Chinese Spoken Language Processing (ISCSLP)* (pp. 126–130).

Li, J., Zhao, R., Hu, H. & Gong, Y. (2019). Improving RNN transducer modeling for end-to-end speech recognition. In *IEEE Automatic Speech Recognition and Understanding Workshop* (pp. 114–121).

Li, K., Li, J., Ye, G., Zhao, R. & Gong, Y. (2019). Towards code-switching ASR for end-to-end CTC models. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 6076–6080).

Li, L., Zhao, Y., Jiang, D., Zhang, Y., Wang, F., Gonzalez, I., ... Sahli, H. (2013). Hybrid deep neural network–hidden Markov model (DNN-HMM) based speech emotion recognition. In *Humaine Association Conference on Affective Computing and Intelligent Interaction* (pp. 312–317).

Li, Y., Pi, S. & Xiao, N. (2018). Speech recognition method based on spectrogram. In *International Conference on Mechatronics and Intelligent Robotics* (pp. 889–897).

Lin, C.-H., Lee, L.-S. & Ting, P.-Y. (1993). A new framework for recognition of Mandarin syllables with tones using sub-syllabic units. In *IEEE International Conference on Acoustics, Speech, and Signal Processing* (Vol. 2, pp. 227–230).

Liu, Z., Chen, Q., Hu, H., Tang, H. & Zou, Y. (2019). Teacher-student learning and post-processing for robust BiLSTM mask-based acoustic beamforming. In *International Conference on Neural Information Processing* (pp. 522–533).

Luong, M.-T., Pham, H. & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Lyu, D.-C., Lyu, R.-Y., Chiang, Y.-c. & Hsu, C.-N. (2006). Speech recognition on code-switching among the Chinese dialects. In *IEEE International Conference on Acoustics Speech and Signal Processing Proceedings* (Vol. 1, pp. I–I).

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

Madikeri, S., Khonglah, B. K., Tong, S., Motlicek, P., Bourlard, H. & Povey, D. (2020). Lattice-free maximum mutual information training of multilingual speech recognition systems. In *Interspeech* (Vol. 2020).

Manaswi, N. K., Manaswi, N. K. & John, S. (2018). *Deep Learning with Applications Using Python*. Springer.

Mansikkaniemi, A. (2010). *Acoustic Model and Language Model Adaptation for A Mobile Dictation Service*. Aalto University.

Mao, X. & Zhang, Y. (2019). Time-delay recurrent neural network for cross-lingual speech recognition. In *Recent Developments in Intelligent Computing, Communication and Devices* (pp. 341–348). Springer.

McDermott, E., Hazen, T. J., Le Roux, J., Nakamura, A. & Katagiri, S. (2006). Discriminative training for large-vocabulary speech recognition using minimum classification error. *IEEE Transactions on Audio, Speech, and Language Processing*, *15*(1), 203–223.

Miao, Y., Gowayyed, M. & Metze, F. (2015). EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)* (pp. 167–174).

Mikolov, T., Deoras, A., Kombrink, S., Burget, L. & Černockỳ, J. (2011). Empirical evaluation and combination of advanced language modeling techniques. In *Annual Conference on the International Speech Communication Association.*

Mikolov, T., Karafiát, M., Burget, L., Černockỳ, J. & Khudanpur, S. (2010). Recurrent neural network based language model. In *Annual Conference of the International Speech Communication Association.*

Mitra, V., Franco, H., Graciarena, M. & Vergyri, D. (2014). Medium-duration modulation cepstral feature for robust speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 1749–1753).

Mohamed, A.-r., Dahl, G. & Hinton, G. (2009). Deep belief networks for phone recognition. In *NIPS Workshop on Deep Learning for Speech Recognition and Related Applications* (Vol. 1, p. 39).

Moritz, N., Hori, T. & Le Roux, J. (2019). Triggered attention for end-to-end speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 5666–5670).

Mousazadeh, S. & Cohen, I. (2013). Voice activity detection in presence of transient noise using spectral clustering. *IEEE Transactions on Audio, Speech, and Language Processing*, *21*(6), 1261–1271.

Newatia, S. & Aggarwal, R. (2018). Convolutional neural network for ASR. In *International Conference on Electronics, Communication and Aerospace Technology* (pp. 638–642).

Pacheco, F. S. & Seara, R. (2008). Dereverberation and denoising techniques for ASR applications. *Speech Recognition*, 81.

Panayotov, V., Chen, G., Povey, D. & Khudanpur, S. (2015). Librispeech: An ASR corpus based on public domain audio books. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 5206–5210).

Park, D. S., Chan, W., Zhang, Y., Chiu, C.-C., Zoph, B., Cubuk, E. D. & Le, Q. V. (2019). Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*.

Park, J., Diehl, F., Gales, M., Tomalin, M. & Woodland, P. C. (2011). The efficient incorporation of MLP features into automatic speech recognition systems. *Computer Speech & Language*, *25*(3), 519–534.

Passricha, V. & Aggarwal, R. K. (2020). A hybrid of deep cnn and bidirectional LSTM for automatic speech recognition. *Journal of Intelligent Systems*, *29*(1), 1261–1274.

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

Petridis, S., Stafylakis, T., Ma, P., Tzimiropoulos, G. & Pantic, M. (2018). Audio-visual speech recognition with a hybrid ctc/attention architecture. In *IEEE Spoken Language Technology Workshop* (pp. 513–520).

Picone, J. W. (1993). Signal modeling techniques in speech recognition. *Proceedings of the IEEE*, *81*(9), 1215–1247.

Povey, D., Cheng, G., Wang, Y., Li, K., Xu, H., Yarmohammadi, M. & Khudanpur, S. (2018). Semi-orthogonal low-rank matrix factorization for deep neural networks. In *Interspeech* (pp. 3743–3747).

Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., ... Schwarz, P. (2011). The Kaldi speech recognition toolkit. In *IEEE Workshop on Automatic Speech Recognition and Understanding*.

Povey, D., Peddinti, V., Galvez, D., Ghahremani, P., Manohar, V., Na, X., ... Khudanpur, S. (2016). Purely sequence-trained neural networks for ASR based on lattice-free MMI. In *Interspeech* (pp. 2751–2755).

Qin, C.-X., Zhang, W.-L. & Qu, D. (2019). A new joint CTC-attention-based speech recognition model with multi-level multi-head attention. *EURASIP Journal on Audio, Speech, and Music Processing*, *2019*(1), 1–12.

Qu, Z., Haghani, P., Weinstein, E. & Moreno, P. (2017). Syllable-based acoustic modeling with CTC-SMBR-LSTM. In *IEEE Automatic Speech Recognition and Understanding Workshop* (pp. 173–177).

Raju, A., Filimonov, D., Tiwari, G., Lan, G. & Rastrow, A. (2019). Scalable multi corpora neural language models for ASR. *arXiv preprint arXiv:1907.01677*.

Rao, K., Sak, H. & Prabhavalkar, R. (2017). Exploring architectures, data and units for streaming end-to-end speech recognition with RNN-transducer. In *IEEE*

Automatic Speech Recognition and Understanding Workshop (pp. 193–199).

Razavi, M. & Doss, M. M. (2015). An HMM-based formalism for automatic subword unit derivation and pronunciation generation. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 4639–4643).

Reindl, K., Zheng, Y., Meier, S., Schwarz, A. & Kellermann, W. (2012). On the impact of signal preprocessing for robust distant speech recognition in adverse acoustic environments. In *IEEE International Conference on Signal Processing, Communication and Computing (ICSPCC)* (pp. 131–135).

Ruder, S. (2017). An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Sak, H., Senior, A. & Beaufays, F. (2014). Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. *arXiv preprint arXiv:1402.1128*.

Sak, H., Senior, A., Rao, K. & Beaufays, F. (2015). Fast and accurate recurrent neural network acoustic models for speech recognition. *arXiv preprint arXiv:1507.06947*.

Saon, G. & Chien, J.-T. (2012). Large-vocabulary continuous speech recognition systems: A look at some recent advances. *IEEE Signal Processing Magazine*, *29*(6), 18–33.

Saz, O., Yin, S.-C., Lleida, E., Rose, R., Vaquero, C. & Rodríguez, W. R. (2009). Tools and technologies for computer-aided speech and language therapy. *Speech Communication*, *51*(10), 948–967.

Schwenk, H., Bougares, F. & Barrault, L. (2014). Efficient training strategies for deep neural network language models. In *NIPS Workshop on Deep Learning and Representation Learning*.

Schwenk, H. & Gauvain, J.-L. (2002). Connectionist language modeling for large vocabulary continuous speech recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing* (Vol. 1, pp. I–765).

Senior, A., Heigold, G., Bacchiani, M. & Liao, H. (2014). GMM-free DNN acoustic model training. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 5602–5606).

Senior, A., Sak, H. & Shafran, I. (2015). Context dependent phone models for LSTM RNN acoustic modelling. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 4585–4589).

Seshadri, N. & Sundberg, C.-E. (1994). List viterbi decoding algorithms with applications. *IEEE Transactions on Communications*, *42*(234), 313–323.

Shan, C., Weng, C., Wang, G., Su, D., Luo, M., Yu, D. & Xie, L. (2019). Investigating end-to-end speech recognition for Mandarin-English code-switching. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 6056–6060).

Shi, X., Feng, Q. & Xie, L. (2020). The ASRU 2019 Mandarin-English code-switching speech recognition challenge: Open datasets, tracks, methods and results. *arXiv preprint arXiv:2007.05916*.

Siivola, V. & Pellom, B. L. (2005). Growing an n-gram language model. In *European*

Conference on Speech Communication and Technology.

Sim, K. C., Narayanan, A., Bagby, T., Sainath, T. N. & Bacchiani, M. (2017). Improving the efficiency of forward-backward algorithm using batched computation in tensorflow. In *IEEE Automatic Speech Recognition and Understanding Workshop* (pp. 258–264).

Smit, P., Virpioja, S. & Kurimo, M. (2021). Advances in subword-based HMM-DNN speech recognition across languages. *Computer Speech & Language*, *66*, 101158.

Sundermeyer, M., Oparin, I., Gauvain, J.-L., Freiberg, B., Schlüter, R. & Ney, H. (2013). Comparison of feedforward and recurrent neural network language models. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 8430–8434).

TAL Education Group. (2019). *TAL CS Auto Speech Recognition Data Set.* Website. (`https://ai.100tal.com/dataset`)

Thomas, S., Seltzer, M. L., Church, K. & Hermansky, H. (2013). Deep neural network features and semi-supervised training for low resource speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 6704–6708).

Ueno, S., Inaguma, H., Mimura, M. & Kawahara, T. (2018). Acoustic-to-word attention-based model complemented with character-level CTC-based model. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 5804–5808).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., … Polosukhin, I. (2017). Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Veselỳ, K., Ghoshal, A., Burget, L. & Povey, D. (2013). Sequence-discriminative training of deep neural networks. In *Interspeech* (Vol. 2013, pp. 2345–2349).

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K. & Lang, K. J. (1989). Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *37*(3), 328–339.

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

Wang, W., Wang, G., Bhatnagar, A., Zhou, Y., Xiong, C. & Socher, R. (2020). An investigation of phone-based subword units for end-to-end speech recognition. *arXiv preprint arXiv:2004.04290*.

Waris, A. & Aggarwal, R. (2018). Acoustic modeling in automatic speech recognition-A survey. In *International Conference on Electronics, Communication and Aerospace Technology* (pp. 1408–1412).

Watanabe, S., Hori, T., Karita, S., Hayashi, T., Nishitoba, J., Unno, Y., … others (2018). ESPnet: End-to-end speech processing toolkit. *arXiv preprint arXiv:1804.00015*.

Woodland, P. C., Odell, J. J., Valtchev, V. & Young, S. J. (1994). Large vocabulary continuous speech recognition using HTK. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (Vol. 2, pp. II–125).

Woodland, P. C. & Young, S. J. (1993). The HTK tied-state continuous speech recogniser. In *European Conference on Speech Communication and Technology.*

Wu, C.-H., Chiu, Y.-H., Shia, C.-J. & Lin, C.-Y. (2005). Automatic segmentation and identification of mixed-language speech using delta-BIC and LSA-based GMMs. *IEEE Transactions on Audio, Speech, and Language Processing*, *14*(1), 266–276.

Wu, C. H., Shen, H. P. & Yang, Y. T. (2014). Chinese-English phone set construction for code-switching ASR using acoustic and DNN-extracted articulatory features. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *22*(4), 858–862.

Wu, H. & Wu, X. (2007). Context dependent syllable acoustic model for continuous Chinese speech recognition. In *Annual Conference of the International Speech Communication Association.*

Yan, W. Q. (2019). *Introduction to Intelligent Surveillance*. Springer.

Yan, W. Q. (2021). *Computational Methods for Deep Learning*. Springer.

Yan, Z.-J., Huo, Q. & Xu, J. (2013). A scalable approach to using DNN-derived features in GMM-HMM based acoustic modeling for LVCSR. In *Interspeech* (pp. 104–108).

Yilmaz, E., Heuvel, H. v. d. & van Leeuwen, D. A. (2018). Acoustic and textual data augmentation for improved ASR of code-switching speech. *arXiv preprint arXiv:1807.10945*.

You, S.-R., Chien, S.-C., Hsu, C.-H., Chen, K.-S., Tu, J.-J., Lin, J. S. & Chang, S.-C. (2004). Chinese-English mixed-lingual keyword spotting. In *International Symposium on Chinese Spoken Language Processing* (pp. 237–240).

Yu, D. & Deng, L. (2016). *Automatic Speech Recognition*. Springer.

Zeiler, M. D. (2012). Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

Zenkel, T., Sanabria, R., Metze, F. & Waibel, A. (2017). Subword and crossword units for ctc acoustic models. *arXiv preprint arXiv:1712.06855*.

Zhang, W., Zhai, M., Huang, Z., Liu, C., Li, W. & Cao, Y. (2019). Towards end-to-end speech recognition with deep multipath convolutional neural networks. In *International Conference on Intelligent Robotics and Applications* (pp. 332–341).

Zheng, Y., Yang, X. & Dang, X. (2020). Homophone-based label smoothing in end-to-end automatic speech recognition. *arXiv preprint arXiv:2004.03437*.

Zou, W., Jiang, D., Zhao, S., Yang, G. & Li, X. (2018). Comparable study of modeling units for end-to-end Mandarin speech recognition. In *International Symposium on Chinese Spoken Language Processing* (pp. 369–373).