Speech to Intent Mapping System For Low Resourced Languages

Yohan Karunanayake

188084V

Thesis submitted in partial fulfillment of the requirements for the
Degree of Master of Science (Research) in Computer Science and Engineering

Department of Computer Science & Engineering

University of Moratuwa
Sri Lanka

January 2020

DECLARATION

I declare that this is my own work and this dissertation does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: Date:

The above candidate has carried out research for the Masters thesis/dissertation under my supervision.

Name of the Supervisor: Dr. Uthayasanker Thayasivam
Signature of the Supervisor: Date:

Name of the Supervisor: Dr. Surangika Ranathunga
Signature of the Supervisor: Date:

ACKNOWLEDGEMENTS

It would never be possible to finish my dissertation without the encouragement, support, and supervision of various personalities, including my mentors, my friends, colleagues, and my family. At the end of this thesis, I would like to thank all those people who made this achievable and memorable experience for me.

First and foremost, I would like to thank my supervisors Dr. Uthayasanker Thayasivam and Dr. Surangika Ranthunga for the continuous support and guidance I received in every aspect while completing this research.

I would also like to thank my progress review committee, Dr. Peshala G. Jayasekara, and Dr. Charith Chitraranjan for their valuable insights and guidance. Their advice helped me to improve the state of my research work.

This work was funded by a Senate Research Committee (SRC) Grant of the University of Moratuwa. Further, I would like to thank LK Domain Registry for the support given for conference participation via travel grants.

Finally, I would like to express my sincere gratitude all of my friends. Not just academic work, I was able to enjoy my time while involving different activities and outings. It helped me to keep my life balanced. I thank my parents who have given me a very fortunate life and always believing, trusting and supporting me.

ABSTRACT

Today we can find many use cases for content-based speech classification. These include speech topic identification and speech command recognition. Among these, speech command-based user interfaces are becoming popular since they allow humans to interact with digital devices using natural language. Such interfaces are capable of identifying the intent of the given query.

Automatic Speech Recognition (ASR) sits underneath all of these applications to convert speech into textual format. However, creating an ASR system for a language is a resource-consuming task. Even though there are more than 6000 languages in the world, all of these speech-related applications are limited to the most well-known languages such as English, because of the high data requirement of ASR. There is some past research that looked into classifying speech while addressing the data scarcity. However, all of these methods have their limitations.

This study presents a direct speech intent identification method for low-resource languages with the use of a transfer learning mechanism. It makes use of three different audio-based feature generation techniques that can represent semantic information presented in the speech. They are unsupervised acoustic unit features, character and phoneme features. The proposed method is evaluated using Sinhala and Tamil language datasets in the banking domain. Among these, phoneme based features that can be extracted from Automatic Speech Recognizers (ASRs) yield the best results in intent identification. The experiment results show that this method can have more than 80% accuracy for a 0.5-hour limited speech dataset in both languages.

Keywords: Speech Intent Identification, Spoken Language Understanding, Low-Resource Languages.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AM	Acoustic Model
AMDTK	Acoustic Model Discovery Toolkit
ANN	Artificial Neural Network
ASR	Automatic Speech Recognition/Recognizer
CNN	Convolutional Neural Networks
CTC	Connectionist Temporal Classification
DBN	Dynamic Bayesian Network
DNN	Deep Neural Network
FNN	Feed-forward Neural Networks
GMM	Gaussian Mixture Models
GPU	Graphics Processing Unit
HLT	Human Language Technologies
HMM	Hidden Markov Model
LM	Language Model
LSTM	Long Short Term Memory
LVCSR	Large Vocabulary Continuous Speech Recognition/Recognizer
MFCC	Mel Frequency Cepstral Coefficients
NLU	Natural Language Understanding
RNN	Recurrent Neural Network
SVM	Support Vector Machine
WER	Word Error Rate

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

This thesis presents a methodology for automatic speech to intent mapping for low resource languages.

1.1 Speech Intent Identification

User interfaces that can understand spoken language such as smart speakers are becoming popular. This type of interfaces respond to the user's natural language and makes it simpler to interact. Amazon Alexa[1] and Google Home[2] are few such commercial devices that provide virtual assistants in homes. These devices can understand the intent of a given free-form speech command or commands. This is enabled via speech to intent mapping.

Here, speech to intent mapping involves associating the free-form speech commands with one or more predefined commands from a set of predefined commands [5]. Hence, users can express their need in natural language.

For better understanding, consider the following commands that we can express into a smart speaker in order to play music.

1. Play the "Blank Space" song.

2. Turn on the music player.

3. Put on the "Blank Space" song.

4. I would like to hear some music.

All these commands intend to play some music on the speaker. We can express this in different ways, and the speech intent identification system must be capable of figuring out the user's intent. In the first and third examples, "Blank

[1] https://developer.amazon.com/alexa

[2] https://en.wikipedia.org/wiki/Google_Home

Space" is a song name user has specified to play. Such entities present in free-form commands are called "slots", where the recognition system can incorporate while executing given commands.

Figure 1.1 illustrates the processing pipeline of a speech intent identification [6] system. First, it extracts speech features such as Mel Frequency Cepstral Coefficients (MFCC) from raw audio data. Then these features are fed into a sequence to sequence learner such as Hidden Markov Models (HMMs), Dynamic Bayesian Networks (DBNs) or Deep Neural Networks (DNNs) to convert speech feature sequence into a textual sequence [7]. Out of these, DNN based solutions provide human-level performance in recent works [1, 2]. These systems are called Automatic Speech Recognition (ASR) systems. Finally, this generated text is processed with Natural Language Understanding (NLU) techniques to identify intentions in the spoken content [8].

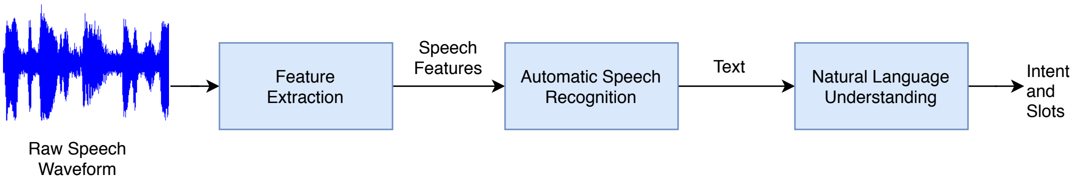

Figure 1.1: Commonly used processing pipeline for speech intent identification

Since this is a cascaded system, an error from a subsystem can propagate and affect the final result. Hence, to enable correct speech understanding, ASR subsystem must work with a very high level of accuracy [6]. Therefore ASR is one of the critical and essential components in this setup. Modern DNN based ASRs require large amount of transcribed and annotated speech data (more than 1000 hours) [1, 2] to provide high accuracy. Hence developing an impactful speech intent identification system for a low resource language is a challenging task. Here, "low-resourced languages" refers to the languages that have a limited presence on the internet and that lack electronic resources for speech and/or language processing [7].

1.2 Research Problem

Developing an impactful voice intent identification system for a low resourced language is a challenging task because of limited annotated speech data.

One obvious solution is to collect more speech data. However, creating an annotated speech corpus is a time and resource consuming task. Thus researchers have experimented with alternative solutions .

One method is developing a multilingual speech recognizer, which is trained on multiple languages and is capable of recognizing all of these languages. Usually this type of recognizer contains a shared model across all languages. Hence it reduces the amount of data need in each language [7]. Another such approach is to adapt ASR trained on a similar or different language to a new language using existing limited data [7].

However, none of these methods has achieved an acceptable accuracy when converting speech into text. As explained in section 1.1, higher accuracy of ASR is crucial for the pipeline of speech intent identification. Thus developing a speech intent identification system for low resource language is a difficult task.

1.3 Research Objectives

The objectives of this research are to

- Find and implement an effective and efficient approach to speech to intent mapping
- Explore the possibility of using high resource language data in the context of speech to intent mapping for low resource languages

1.4 Contributions

- Created a new intent annotated speech corpus for Tamil language in the banking domain

- Developed a novel methodology for direct speech intent identification with ASR character probability values

- Developed a novel methodology for direct speech intent identification with ASR phoneme probability values

1.5 Publications

- "Transfer Learning Based Free-Form Speech Command Classification for Low-Resource Languages" in Proceedings of the 57th Conference of the Association for Computational Linguistics: Student Research Workshop (ACL SRW) 2019 Jul (pp. 288-294).

- Sinhala and Tamil Speech Intent Identification From English Phoneme Based ASR in 2019 International Conference on Asian Language Processing (IALP) 2019 Nov.

1.6 Datasets

Section 5.1 provides a detailed explanation of the datasets. Further, we made the datasets accessible over the internet for the experimentation on low-resource speech[3].

[3] http://rtuthaya.lk/sinhala-tamil-speech-intent-dataset

Chapter 2

BACKGROUND

This chapter provides background theories and required details on implementing speech intent identification systems. Section 2.1 presents the information about speech feature extraction which is an important step in speech recognition. Following this, the next section provides details about ASRs. Section 2.3 presents theories and background information about Artificial Neural Network (ANN) based ASR technology.

2.1 Speech Feature Extraction

Feature extraction from speech signals is an important factor in the ASR pipeline. Speech or sound is produced by a continuous change of air pressure that emerges from a speaker's mouth, nose, and cheeks. Air pressure varies with time. Hence it can be called as a function of time. Microphones convert such fluctuating air presser waves into an electrical wave, fluctuation voltages or currents with time, in which speech processing dealt with as speech signal. For the storage and transmission purposes, this continuous signal will be converted into digital format via analog-to-digital converters.

It is difficult to directly use time-domain speech signal for speech recognition or related task. Hence speech signal is converted into an information-rich representation. This is known as speech feature extraction. The following are some of the commonly used hand-crafted features extraction methods for speech.

- Linear Predictive Cepstral Coefficients (LPCCs)
- Perceptual Linear Predictive (PLP) analysis
- Mel-Frequency Cepstral Coefficients (MFCCs)
- Mel-frequency Filter banks (FBanks)

Among these, MFCC is the most commonly used and effective feature for ASR. The figure 2.1 presents steps of the MFCC feature extraction process. The first step in MFCC feature extraction is Pre-emphasis. Here, filters are used to increase the energy in higher frequencies. In windowing process signal is brake into parts using a sliding window. Usually 25ms window size is used with 10ms sliding. Next using this windows, the time domain signal is converted into frequency domain via Discrete Fourier Transform (DFT). Then Mel Filter banks are applied to convert the signals into human perceiving scale. This is known as Mel Scale and it measure the pitches as perceived by humans with equal distances. Usually this has a log scale and in the next process log scale is applied. Finally cepstral coefficients are extracted by applying Inverse Discrete Fourier Transform (IDFT).

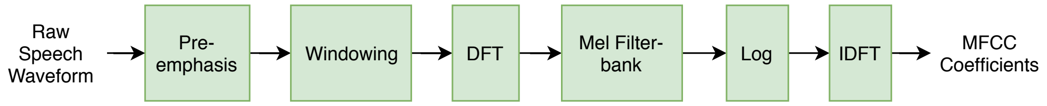

Figure 2.1: MFCC feature extraction process

Most of neural network based ASR models also use MFCC features as input. However, in recent studies, there were few attempts to do ASR from the raw speech wave. In the work of Ravanelli et al. [9], researchers demonstrate this and Lugosch et al. [4] use this to develop an end-to-end ASR system.

2.2 Automatic Speech Recognition

Automatic speech recognition or in short ASR involves predicting the most probable character (symbols of the target language) or word sequence for a given acoustic feature sequence. This can be modeled as follows.

$$\hat{W} = \arg\max_{W} P(W/X) \tag{2.1}$$

Here $\hat{W}$ represents the most probable character/word sequence. W and X represent the possible set of character/word and acoustic features. $P(W/X)$ is the probability of a character/word sequence for a given acoustic feature sequence. It is impractical to calculate $P(W/X)$ straight away, hence by applying Bayes' rule Equation 2.1 can be rewritten as follows.

$$P(W/X) = \frac{P(X/W)P(W)}{P(X)} \tag{2.2}$$

By combining Equation 2.1 and Equation 2.2 following result can be obtained. $P(X)$ is ignored since concern is about maximum value.

$$\hat{W} = \arg\max_{W} P(X/W)P(W) \tag{2.3}$$

Now, it is possible to calculate variables on the right side using a sample dataset. $P(W)$ is referred to as the language model (LM). LM can be created using a sample text corpus and it outputs the probability for a given sequence of words. $P(X/W)$ gives the probability of the sequence of speech features given the word sequence. This is referred to as an acoustic model (AM). Traditionally, ASR is enabled by Hidden Markov Models (HMM). In HMM models, the most probable word or character sequence is calculated through Viterbi decoding. Decoding is done using a weighted finite-state search graph. It uses the scores obtained from the AM, LM and a pronunciation lexicon. Figure 2.2 depicts the components and process of HMM-based speech recognition. However today these HMM-based ASR systems have been replaced by Artificial Neural Networks.

Word Error Rate - WER is the metric used to measure the accuracy of an ASR. WER is calculated from the Levenshtein distance at the word level. Following equation shows the calculation.

$$WER = \frac{S + D + I}{N} = \frac{S + D + I}{S + D + C} \tag{2.4}$$

where,

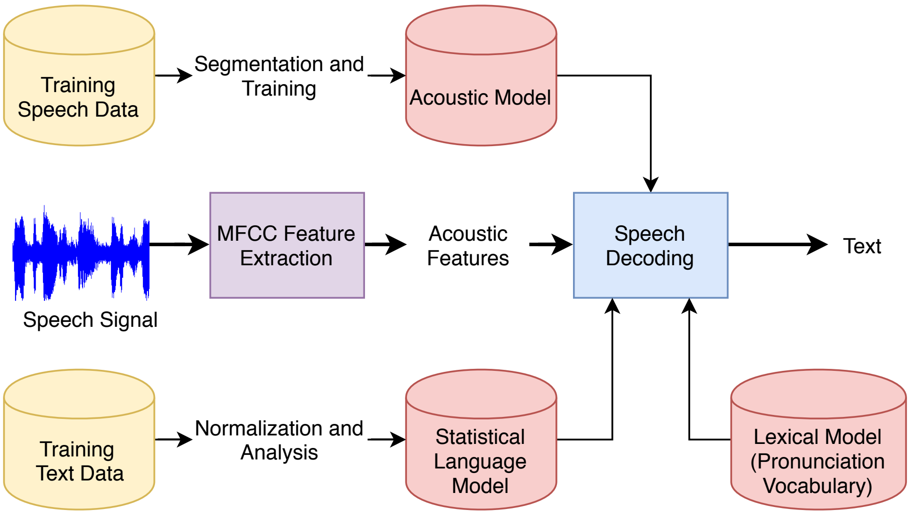

Figure 2.2: Components and processing stages of a HMM-based ASR system

S - number of substitutions

D - number of deletions

I - number of insertions

C - number of correct words

N - number of words in the reference

2.3 Artificial Neural Network In ASR

Artificial Neural Networks (ANN) are a set of algorithms that are inspired by biological neural networks in the animal brains. These algorithms can learn to perform a given task by considering examples. Hence ANN is an essential concept in the current Machine Learning paradigm. An artificial neuron is the fundamental element of ANN. Artificial neurons model the behavior of a biological neuron as a mathematical function.

ANNs are created by stacking multiple sets of neuron layers. Figure 2.3 illustrates a 3 layer fully connected neural network. Here, neurons are donated by the circles and, connecting lines represent the interconnections in between neurons. Every ANN has an input layer and an output layer which takes the input values and output the final results of the neural network. Intermediate layers are called hidden layers and the network in the figure contains only a single hidden layer.

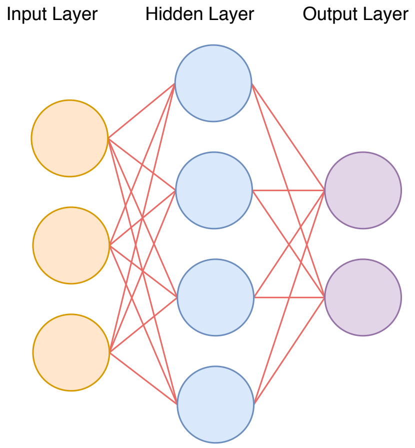

Figure 2.3: A 3 Layer Neural Network

ANNs with more than 3 layers is referred to as Deep Neural Networks (DNNs). Further, there are different types of neurons which have distinctive features. Often these are used as layers where single layer contains same type of neurons. Combining different types of neuron layers results different ANN/DNN architectures which has distinctive performances in different input data types.

In training, output of the ANN is extracted by passing input data through the network. Then error value is calculated between the ANN output and the actual output. The function used to calculate error can vary based on the application. Finally, an optimization algorithm is used to adjust the model parameters according to the error values.

When designing an ANN, it in necessary to define a set of parameters. This contains type of layer, number of neurons in each layer, number of layers, activation functions likewise. These are known as hyper-parameters and need to be selected in careful consideration. These parameters depend on various factors such as types of training data (images, set of numbers or time series), size of the data and more.

With the emergence of deep learning, researchers replaced heavily engineered ASR pipeline with DNN/ANN models. Since DNN had good discriminative power, in early days DNN models were used to recognize phones but leave the decoding part for HMM. These are known as hybrid ASR systems. Later all neural end-to-end models were introduced and produced state-of-the-art results in speech recognition tasks [10]. DeepSpeech [1], DeepSpeech 2 [2] and Wav2Letter [11] some of the remarkable end-to-end speech recognition models proposed in resent studies. This end-to-end all neural models can have different combination of ANN layer types such as Feed-forward, Convolutional, Recurrent layers.

2.4 Different ANN Layers used in ASRs

As described in above, there are different types of ANN layers used in ASR. Following subsections will present different ANN layer types and their functionality, that are discussed in this study.

Dense Layers

The dense or fully-connected layer has connections from every neuron in previous layer to each neuron in the current layer. Each connection is regularized by a separate weight value which is adjusted in the training process. Equation 2.5 represent the dense layer operation. x, y, W, b, f are respectively input vector, output vector, weight matrix, bias and activation function. Activation function and the number of node in a layer are the important hyper-parameters in the dense layer. ANN consist of dense layers usually known as Feed-forward Neural Networks.

$$y = f(W.x + b) \tag{2.5}$$

Recurrent Layers

In a recurrent layer or a Recurrent Neural Network (RNN), information flows not only in a feed-forward manner but also in a temporal sequence. Neuron in an RNN uses its previous output as input for its current calculation which is known as a recurrent connection. Hence, RNN is capable of modeling temporal sequences. Equations 2.6 to 2.9 represent the mathematical operations in a RNN.

Figure 2.4 illustrate this information in a diagram. Here, the left diagram shows time unfolded operations.

$$a^{(t)} = Vh^{(t-1)} + Ux + b \tag{2.6}$$

$$h^{(t)} = \tanh\left(a^{(t)}\right) \tag{2.7}$$

$$o^{(t)} = Wh^{(t)} + c \tag{2.8}$$

$$y^{(t)} = \mathrm{softmax}(o^{(t)}) \tag{2.9}$$

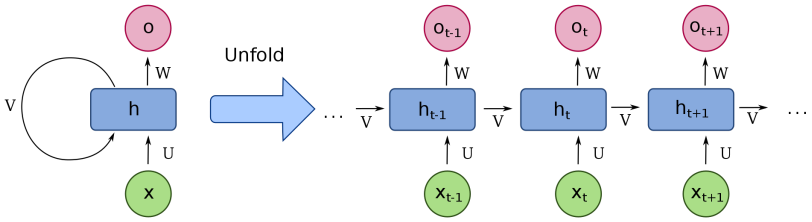

Figure 2.4: Recurrent layer architecture and it's unfolded steps

Source: `https://en.wikipedia.org/wiki/Recurrent_neural_network`

In practice, there are variants of RNNs. Some of them are as follows.

LSTM : Long Short Term Memory networks or LSTMs are introduced to overcome the issues of long-term dependency modeling in RNN and has been successful in many applications such as machine translation, text generation, image captioning and may more [12]. Further LSTMs are largely used in speech recognition models. Figure 2.5 represents an LSTM neuron and its internal operations.

Bi-RNN or Bi-LSTM : All the above discussed recurrent network captures information only from the past and the present. Often some applications depend on the whole input sequence. As an example, in speech recognition, the correct construction of the current phoneme depends on the next set of phonemes. Bidirectional RNN can address these issues. Bi-RNN combines two RNNs, one captures information of the forward flow and, another for the backward flow.

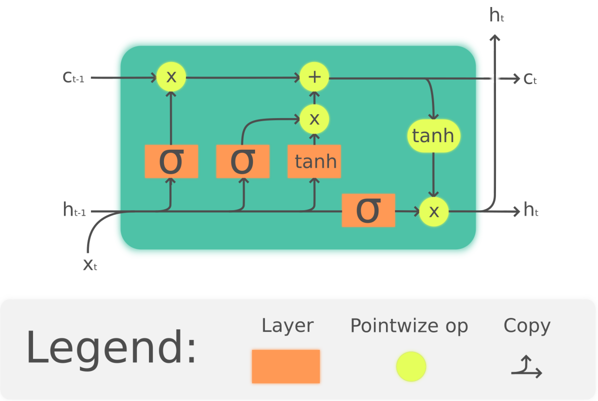

Figure 2.5: LSTM neuron and its internal operations

Source: `https://en.wikipedia.org/wiki/Long_short-term_memory`

Convolutional an Pooling Layers

Convolutions layer is designed to operate on 2D structures like images or speech signals. This layer performs a convolutional operation on this 2D data using another smaller 2D data matrix which is known as filter or kernel by sliding in both dimensions. The result of this convolution operation is known as a feature map. For a given layer user need to define the number of kernels, size of the kernel and stride or sliding window size. Further, convolutional layers are often used with subsampling layers which are known as pooling layers. Each feature map is then subsampled typically with mean or max-pooling over contiguous regions by sliding. Hyper-parameters for the pooling layers are the size of the pooling region and the sliding length. Generally ANN with convolutional and pooling layers are called Convolutional Neural Networks (CNNs) Figure 2.6 illustrates an information flow of a CNN with two convolutional and pooling layers.

In CNN, there are two variants based on the convolution filter movements which is known as 1D CNN and 2D CNN. In 1D CNN, convolution filters move along only on 1 dimension while in 2D filters move along the two dimensions. Figure 2.7 depicts the operations of these two variants. Usually, 1D CNN is mostly used to process 1D sequence data like signals and 2D CNN is used to process data which can vary with both directions such as 2D images.

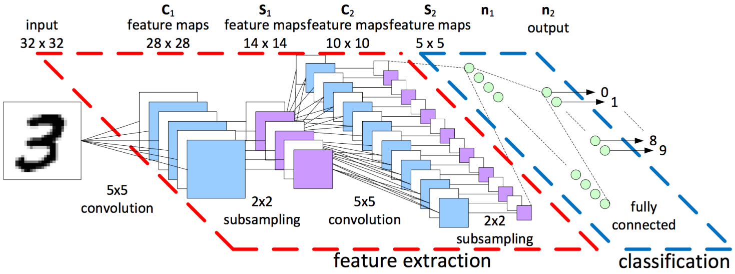

Figure 2.6: CNN with 2 convolutional and pooling layers

Source: `https://becominghuman.ai/how-should-i-start-with-cnn-c62a3a89493b`

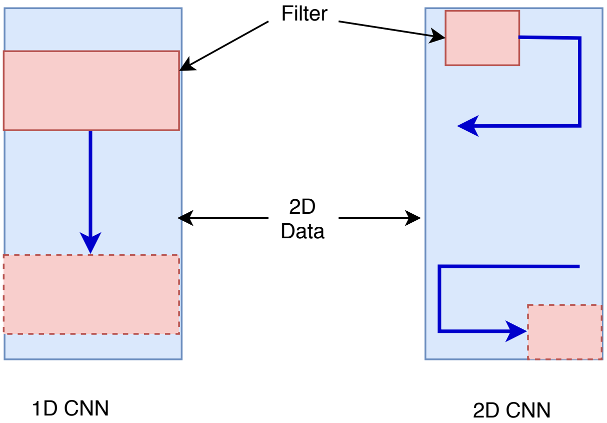

Figure 2.7: 1D vs 2D CNN operation

Chapter 3

LITERATURE SURVEY

This chapter present literature related to speech intent identification and critical analysis of it with respect to low-resource languages. The first section is dedicated to give brief overview about speech recognition because of the relevance. The second section will introduce work related to speech intent identification and the next section will discuss existing approaches for low-resource speech intent identification.

3.1 Automatic Speech Recognition

Automatic Speech Recognition (ASR) is aimed to convert human speech into a textual representation and it has been an intense research field for a long time. ASR is the core technology for various fields including command and control, transcription of recorded speech, conversational agents and, many more. The appearance of different elements makes the ASR system further complex. However, a robust ASR system must be capable of handling different kinds of variabilities which present in the input speech. Background noises, diverse speakers, speaking style are the factors that can affect the performance of the ASR system.

Many impotent concepts like Hidden Markov Models (HMMs), Gaussian Mixture Models (GMMs), Mel-Frequency Cepstral Coefficients (MFCCs) features, discriminative training, and speaker adaptation methods have been developed with the time. Among these, HMM was a prime technology for building ASRs in the past. CMU SPHINX [13] and Kaldi [14] are some of the ASR toolkits still in the use which are based on HMMs. Even though training an HMM-based ASR system for a new language is straightforward, it requires expert knowledge in various arias like signal processing, statistics models, the phonology of target language [7].

The interest became high for the ASR and related technologies in the past few years. This was due to the improved demand for speech-related applications and the advancement of deep learning methods. The availability of large speech datasets like Librispeech [15], Mozilla Common Voice[1] that contain 1000+ hours of speech data makes it easy to conduct further research. The availability of GPU computing expedites the development of ASR on these large datasets.

DeepSpeech [1], Wav2Letter [11] are recently developed Deep Neural Network (DNN) based model which have achieved human-level recognition accuracy. Further, these models are called Large Vocabulary Continuous Speech Recognizer (LVCSR) since it can recognize almost all the utterances in a trained language. This is enabled by a scoring algorithm that uses both ASR output and an LM. LMs are trained on text corpus to predict the likelihood of a given character sequence. When LM is trained on large text corpus it can correct minor errors done by the ASR model and expand the possible recognizable vocabulary in the combined system. Unlike HMM models these DNN based ASR systems can be trained on a new language as long as having enough data and computation power.

Table 3.1 present some of the recently introduced end-to-end ASR models and their performances in different datasets. Following two subsection introduce two of these ASR models which is used in this study.

Model	Trained Language	Dataset	WER
Wav2Letter [11]	English	LibriSpeech	7.2%
Listen, Attend and Spell [16]	English	Google Voice Search Task	Clean:10.3% Noisy: 12.0%
DeepSpeech [1]	English	Switchboard Hub5'00	16.0%
DeepSpeech2 [2]	English Mandarin Chinese		Eng:5.15% Man:7.93%

Table 3.1: Existing end-to-end ASR models

DeepSpeech Model

DeepSpeech (DS) [1] was introduced as all neural end-to-end speech recognition model to replace the traditional speech processing systems which needed carefully engineered processing pipelines. Further, DS had good performance with noisy

[1] https://voice.mozilla.org

speech recognition. As above described, the availability of multiple GPUs and large datasets made it possible to train this type of large RNN.

Let an utterance x and corresponding transcript y which is taken from a training set $X = (x^{(1)}, y^{(1)}), (x^{(2)}, y^{(1)}), ...$ Here, each $x^{(i)}$ is a sequence of audio feature vectors with the length of time $T^{(i)}$. The goal of the ANN model is to transform the given input sequence x into a sequence of character probabilities of the transcription y, with $\hat{y}_t = \mathbb{P}(c_t|x)$ where c_t represents the possible character set including other special characters - (to represent spaces, silents) in the training language.

DS has a simpler neural network architecture and consists of 5 layers. The first 3 layers are simple feed-forward/dense layers. Then, there is the bidirectional recurrent layer which tries to capture the temporal features of the speech both in the forward and backward direction. The final layer is another feed-forward/dense layer which output probability values. Beam search is applied to these probability values while incorporating LM to produce final text output. Figure 3.1 illustrate the ANN layer arrangement of the DS model.

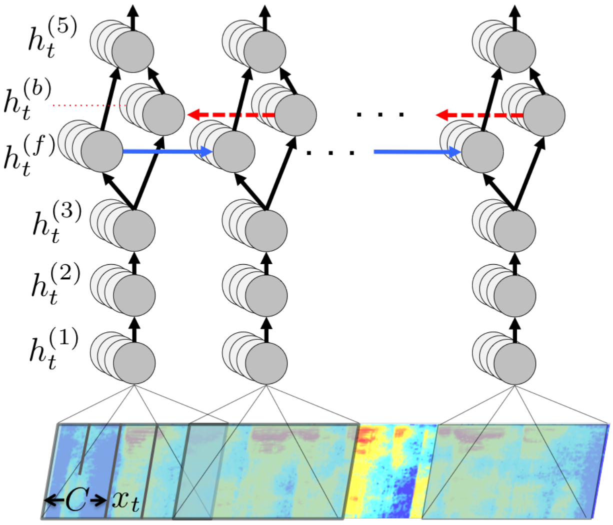

Figure 3.1: DeepSpeech Model Architecture; Source: [1]

To train the DS the requirement is a large speech corpus and computing power. Here, DS model does not require exact alignment of the speech transcripts. For that, the model uses Connectionist Temporal Classification (CTC) loss function [17] to calculate the error in the training phase.

DeepSpeech2 Model

DeepSpeech2 (DS2) was introduced as an improved version of the previous DS model. It is designed to handle a wide variety of speech including noisy environments, accents, and different languages. To facilitate this DS2 has a different layer arrangement. Figure 3.2 illustrates the architecture of the DS2 model. Further this model also outputs character probability values for a given audio feature sequence as in DS.

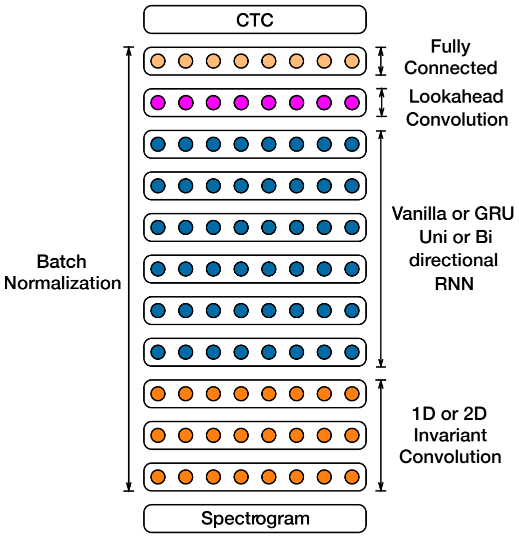
CTC
Fully Connected
Lookahead Convolution
Vanilla or GRU Uni or Bi directional RNN
Batch Normalization
1D or 2D Invariant Convolution
Spectrogram

Figure 3.2: DeepSpeech 2 Model Architecture; Source: [2]

3.2 Low-Resource Automatic Speech Recognition

While ASR technology growing, one major issue was the need for annotated speech corpus. In the past audio clips need to have the exact alignment of the phonemes or characters for training. Creating a speech transcript with this information requires highly skilled people and this is a time-consuming task. However, with the introduction of the Connectionist Temporal Classification (CTC) [17] need of exact alignment of the transcript was eliminated. But this increased the amount of training data [18, 1, 2]. Because of these reasons, researchers have given some focus to the low-resource ASR development.

Even it is difficult to obtain transcribed speech data, speech recordings can be obtained by news broadcasts, recording of the speech or conference talks, television programs and many more. When there is such untranscribed data, unsupervised or lightly-supervised approaches have been used to develop ASR systems [19, 20, 21]. In a situation where some previous knowledge or data of the targeted language is accessible, such as audio recordings, pronunciation dictionaries and LMs, the unsupervised methods are very beneficial to preserve time and costs by developing an ASR system for a new language [21].

However, with the shift from the HMM-based model to DNN based model, researchers have improved the accuracy of low-resource languages. They have used data of high resource languages to achieve this [7, 22]. In this method first, they train an ASR model using high resource language data. Then this trained model is retrained on another language. This reduces the amount of needed data to achieve higher accuracy since the initial model is trained on a similar task. However, this method still requires at least 300 hours of speech data to get a reasonable accuracy value [22].

Moreover, researchers have proposed multilingual speech recognizers to overcome the limited data issue [3, 23]. Figure 3.3 demonstrate an architecture of such DNN based multilingual ASR model. In these models, part of the model is shared across all languages as in figure 3.3. In training, this shared part is trained to generate language-independent feature representation. since there are multiple language datasets, amount of data need from each language is low.

3.3 Speech Intent Identification Systems

Speech intent identification systems became popular with the rice of speech-enabled user interfaces and conversational agents. These technologies enable natural language based human-machine interactions. Snips Voice Platform [24], Google Home, Amazon Alexa are such commercial level systems which provide control over speech commands.

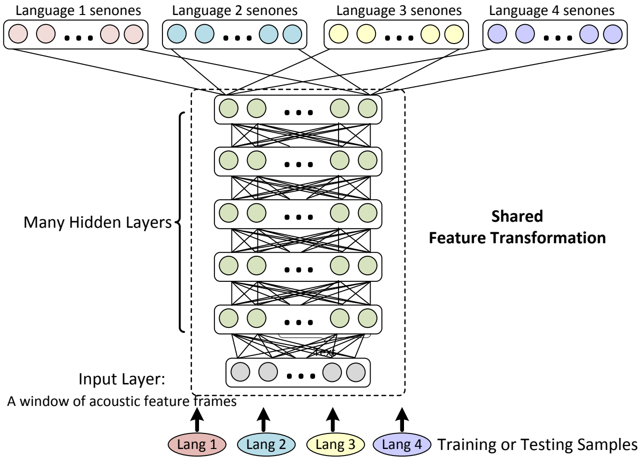

Figure 3.3: Architecture of DNN based multilingual speech recognizer; Source: [3]

These systems use separately trained ASR and NLU subsystems to identify intent in the speech. In this way, text transcripts obtained from the ASR model, are used as input for a separate text classifier (See Figure 1.1). In these cascaded systems, ASR needs to generate accurate transcripts for a given speech query. An erroneous transcript can divert the final results of this system [25, 26].

In this conventional method, ASR is trained on with the objective function fo reducing the incorrectly predicted words (Word Error Rate - WER). Hence we can not assume that the 1-best hypothesis from the ASR is always correct. Further, sometimes ASR tends to produce misspelled words. As a solution for this researchers have used n-best outputs from the ASR. Further NLU subsystem is separately trained to predict the intent from text data. Yaman et al. [25] presented a joint optimization technique for this issue. Then, He and Deng [27] advanced this effort by producing a generalized framework.

Today with the use of the neural network, researches try to develop end-to-end models. These models are trained in an end-to-end manner and try to infer the semantic meaning directly from audio features. Serdyuk et al. [28] and Haghani et al. [29] introduced such end-to-end models. These models are capable of identifying the domain, intent and the slot presented in the given speech query. Further, Chen et al. [30] present a similar system for call center speech data classification. These end-to-end models are large and contain more than 50 millions of parameters [29]. To train modes with such a large number of parameters, it is essential to have large datasets. Further to optimize the model convergence researches have utilized pre-training strategies [29, 30].

In speech intent identification, the aim is to identify the intent in a given spoken utterance. Snips Voice, Google Home, Amazon Alexa like systems try to identify the intent of an issued command. In addition to that, identifying the intent of call center calls can also be categorized into the speech intent identification. It is possible to find another related task that tries to identify the topics of the speech [31, 32, 33]. Here, the topics are similar to intents, but they vary in a broad range and try to represent the whole subject presented in the spoken content.

3.4 Low-Resource Speech Intent Identification

In the literature, there are two categories of approaches which can be identified for low-resource speech intent identification. Here, the major barrier is developing an ASR system with limited data. Hence some researchers have given focus to develop ASR systems while others focusing on finding alternative ways. Following subsections present these details.

3.4.1 ASR Based

As discussed in section 3.2, researchers have focused on developing ASR systems optimized for languages with smaller speech corpus. One method is to adapt or retrain of an ASR trained in high resource language. Another way is multitask-learning with multiple smaller speech datasets.

Wiesner et al. [33] propose a speech topic classification methodology that provides promising results when there is very limited training data. They use a multilingual speech corpus and universal phones [34] to develop a low-resource ASR system. The text output of the ASR is fed into a text classifier to identify the corresponding topic. Here, the topics can be described as intents and there are 11 different types. Some of them are “Evacuation”, “Food Supply”, “Urgent

Rescue", "Medical Assistance" and, "Shelter". However, obtaining or creating a phoneme annotated speech corpus is a difficult task. Further, having a good understanding of the phonology of the targeted low-resource language is essential in this approach.

3.4.2 Feature Based

This approach uses features generated from the respective speech queries and does not rely on the text output of the ASRs. Using these features a classifier model is trained to detect the intent. The figure 3.4 illustrates this process. Few prominent feature generation techniques can be identified in the literature.

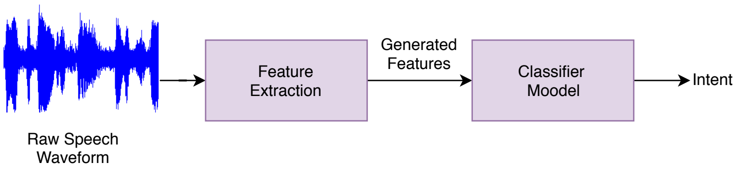

Figure 3.4: Feature based Speech Intent Identification Approach

Liu et al. and Wiesner et al. [31, 33], use features such as phone-like units discovered via acoustic unit discovery (AUD) [35, 36] or word-like units discovered via unsupervised term discovery (UTD) [37] in their work. Here, AUD and UTD are unsupervised feature generation techniques and does not require any transcript of the speech. But it is essential to have more and more data to get a better feature representation for the audio. Additionally, more data means it is essential to have more computation power to process all these data.

Buddhika et al. [38] presented an MFCC feature-based speech intent classifier suitable for low-resource languages. On top of these features, researchers have used different feature classification methods such as Support Vector Machines (SVM), and CNN to identify intents. Their approach achieves 74% classification accuracy for a 10 hour Sinhala dataset.

In another work, Chen et al. [30] propose an intent identification method for the English language queries using intermediate features of a pre-trained English ASR model. Here researchers use character probability values generated by the

ASR as features on a CNN based classification model to identify the intent and show good results. Utilizing this pre-training strategy Lugosch et al. [4] also present such similar work. In their work, they try to identify not only the intent but also slot values such as action, object, and location mentioned in the speech query and achieve better results using a 14.7-hour dataset. However, in both of these works, an ASR trained on large English corpus is used to identify intent on the same language.

3.5 Summary

This chapter provided the details about the past efforts on speech intent classification. Further, it presented some background information on the evolution of ASR and low-resource ASR. In general approach, speech intents are identified combining robust ASR and NLU subsystems. Moreover, there were some other proposed techniques to do the same task, which uses different features instead of ASR generated text. The secondly generated methods show better results and still need a reasonable amount of speech data.

Chapter 4

METHODOLOGY

As discussed in the previous chapter, there are different research attempts to identify low-resource speech intent. Among these, methods that use different features other than the ASR generated text showed comparably better performances in speech classification tasks. Further, this approach eliminates the issue of using the 1-best ASR output and also the need for robust ASR. Hence, this study focuses on feature-based intent classification approaches.

Methodology in this research can be divided into two steps. One is identifying a better feature generation technique for low resource speech that can represent semantics presented in a given spoken utterance. The next step is to identify a better feature classifier for the generated features. Figure 4.1 depicts the steps in presenting method.

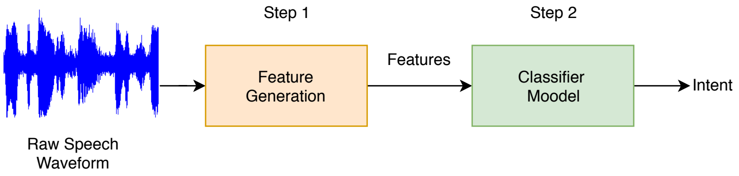

Figure 4.1: Methodology Steps

‘

In this chapter, Section 4.1 introduces two different feature generation approaches. Section 4.2 describes the classifier models used to identify intent from the features.

4.1 Feature Generation for Intent Identification

As described, feature generation is one of the important steps in the proposing methodology for speech intent identification. We experimented with two different feature generation methods.

4.1.1 Deep Neural Network Based ASR Feature Extraction

Chen et al. [30] use features generated via an ANN-based ASR to classify the intent of the speech. Few other research identifies not just intent but also domain, intent, and slots presented in the speech query [29, 4]. In all these works, features generated by the ASR models are used for the label prediction model. However, all the this was experimented on the English language using large datasets. Further, much attention is not given to the use of cross-language feature generation.

There are publicly available large datasets for high-resource languages like English. As described in Section 3.2, such large datasets can be used to develop ASR models for the resource-poor languages. For a very limited speech data, it is effective to use such pre-trained ASR as a feature generates rather than fine-tuning it to generate text (See Section 3.2).

In the machine learning paradigm, this technique is known as transfer learning, where we try to reuse a model trained on one task in another similar or related task [39]. Here, an ASR model trained to convert high resource speech into text is used on low resource speech intent identification. In some literature, authors referred to this as ASR model pre-training.

The following two subsections describe two different new feature extraction methods that are introducing in this study. These feature extraction methods use high-resource ANN-based ASRs.

Character Probability Features

Table 3.1 present some of the existing best performing ASRs. Among these, DeepSpeech, DeepSpeech2 models were selected to extract features in this study. These models have a good performance and have an open-source implementations [1] [2].

As described in Section 2.2, ASRs like DeepSpeech, DeepSpeech2 are trained to convert a given speech feature sequence into a most probable character se-

[1] DeepSpeech - `https://github.com/mozilla/DeepSpeech`

[2] DeepSpeech2 - `https://github.com/SeanNaren/deepspeech.pytorch`

quence. To facilitate this, these model provide sequence probabilities $y^{(i)}$, with $\hat{y}_t = \mathbb{P}(c_t|x)$ for a given acoustic feature sequence x. Here, c_t represents the possible character set in the training language. These values can be used to identify features. These character probability features are two dimensional. One dimension represents the different characters and the other represents the time steps of the sequence. Figure 4.2 provides a visualization of the extracted character probability feature.

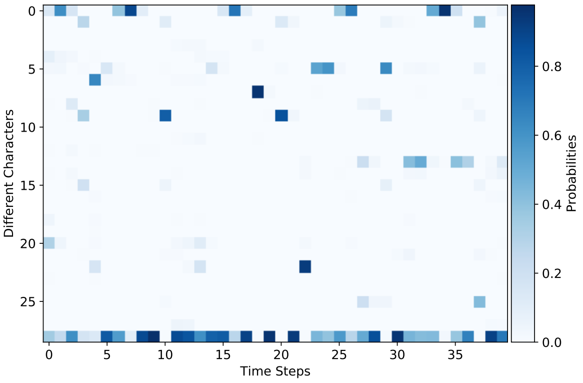

Figure 4.2: Visualization of Character Feature

To extract features, first, the ASR model is trained in high resource language. There are freely available speech datasets like LibriSpeech for high-resource languages. Such datasets can be used to train ASR models initially. In the training process, models learn to map high resource language audio features into the corresponding language characters. Then, features for the low-resource language is generated by feeding them as input for the trained model. After extracting the features, a feature classifier model is trained to identify the intents. Figure 4.3 illustrates the whole intent identification process based on this character feature extraction process.

Here, the ASR model in Figure 4.3 can be replaced by any ASR model that output character probability values (models like DeepSpeech and DeepSpeech2). Further, it is possible to extract features from any hidden layer. But, the character probability values generated from the final layer are used as features. These ANN-based ASR models use a stack of CNN and RNN layers (See Section 3.1 and 3.1).

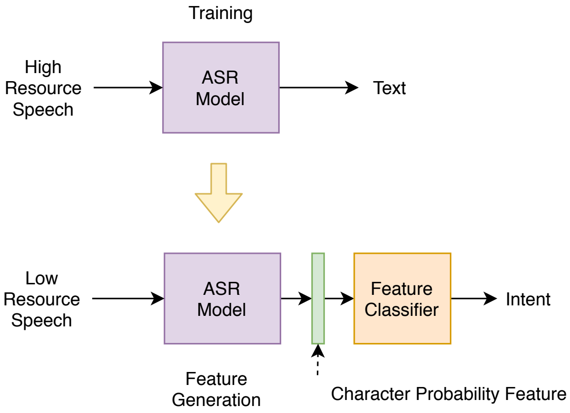

Figure 4.3: Character Feature Extraction

As described in Section 2.4 RNN are important to capture temporal information present in the audio feature sequences.

Phoneme Probability Feature

Phonemes represent a perceptually distinct unit of sound in a language, and it is used to present the articulation of the words in a language. Therefore phonemes can represent different sounds accurately compared to characters. Hence, the use of phoneme probability values features instead of characters have the likelihood of increasing the accuracy of intent classification. This scenario is exploited in this proposing method.

In order to have this, there should be a model that outputs a sequence of phoneme probabilities $p^{(i)}$, $\hat{p}_t = \mathbb{P}(p_t|x)$ when given a sequence of acoustic features $x^{(i)}$ where p_t represents the possible phoneme set.

All of the ASR models are trained to predict characters, and it is not so common to have ASRs that predict phonemes. However, Lugosch et al. [4] present an ASR model development method that uses phonemes as intermediate targets. In this method, authors use phoneme and word-level alignment rather than raw transcripts and CTC loss. Here, Montreal Forced Aligner [40] is used to generate these alignments. Hence, the ASR model proposed by Lugosch et al. is used to generate phoneme probability features.

Figure 4.4 represent the architecture of our phoneme feature generation ASR model, and Figure 4.5 depicts the phoneme feature extraction process. Phoneme probability features can be generated following this process instead of character probabilities. Figure 4.6 presents a visualization of phoneme probability features.

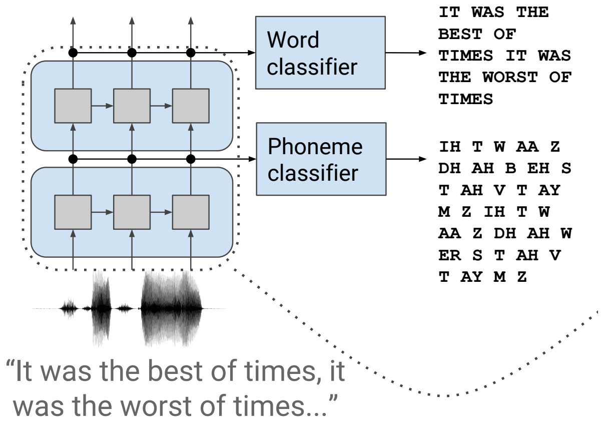
Word classifier
IT WAS THE BEST OF TIMES IT WAS THE WORST OF TIMES
Phoneme classifier
IH T W AA Z DH AH B EH S T AH V T AY M Z IH T W AA Z DH AH W ER S T AH V T AY M Z
"It was the best of times, it was the worst of times..."

Figure 4.4: Architecture of the phoneme based ASR; Source: [4]

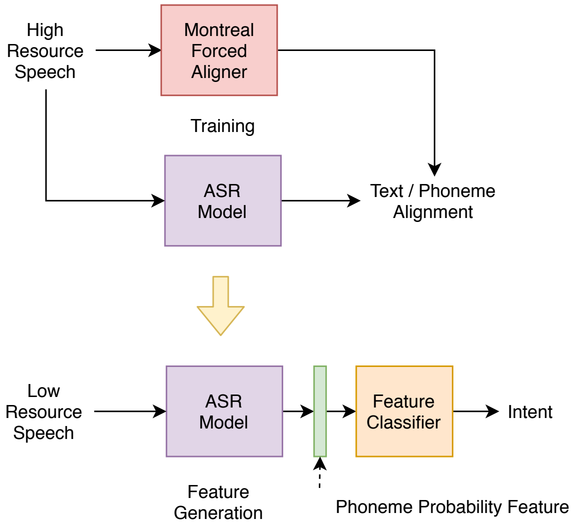

Figure 4.5: Phoneme Feature Extraction

4.1.2 Unsupervised Feature Extraction

Ondel et al. [35] presented an unsupervised method to automatically separate unlabeled audio data into distinctive sound units. These units are like phonemes, and segmentation accuracy increases with the available data amount. Kesiraju et al. [32] successfully used this method as a feature generation technique for topic

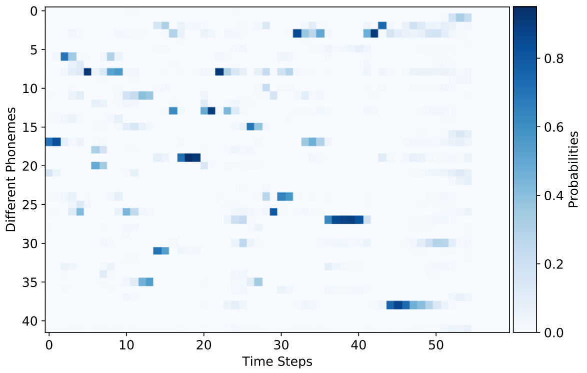

Figure 4.6: Visualization of Phoneme Feature

identification of spoken documents. The effectiveness of this methodology has experimented in this study for intent identification in limited data scenarios.

This method uses a Bayesian phone-loop model as in HMM [41] and tries to segment and cluster untranscribed speech data into the phoneme like units. However, there is no guarantee that the model is learning the exact phoneme like units. Each of these units is represented by left-to-right HMM and is embedded into a loop structure. This model combines a prior distribution over the parameters of the HMMs and has a prior distribution over the units modeled by a Dirichlet process. Here, the prior distribution or simply prior represents the beliefs about an uncertain parameter that is combined with the probability distribution before some evidence is taken into account.

The model assumes that N speech data samples are generated with only M sound components ($M \leq N$) and M is not a fixed number. Hence the model learns its complexity according to the data. The implementation of the model is available as a framework[3]. This is used to generate unsupervised features to identify intent. After training with a speech corpus, the model produces a number sequence for a given speech query. These numbers represent the M number of sound components model able to learn in the training process.

Notations : Let Q be the set of queries containing a vocabulary V, and consider each query belong to one intent from a set of intents I. Let q, w, i

[3] AMDTK - `https://github.com/iondel/amdtk`

be denoting speech queries, tokens in the vocabulary, and intents respectively. Vocabulary is determined by considering the 3-grams of the discovered phone-like units.

The following are the steps to generate features for the intent classifier.

1. Generate acoustic units using the AMDTK tool for speech queries

2. Select top N_i 3 grams per each intent with highest conditional probability of intent i given a 3-gram w using following equations. Here, f_{wi} is the number of 3-gram w that appeared in the query related to intent i, f_w is the total number of appearance 3-gram w in all the training queries. $P(i)$ is estimated using the training data.

$$P(i|w) = \frac{f_{wi} + |I|P(i)}{f_w + |I|} \tag{4.1}$$

3. Convert queries into vectors using smoothed TF-IDF (term frequency - inverse document frequency) representation (v_{wq}) as follows. Here, f_{wq} is the frequency of token w in the query q, and N_{qw} represents the number of documents in which the term w appears. This resulting vectors are normalized, such that the sum of the squares of elements equals to 1.

$$v_{wq} = f_{wq}.\log(\frac{|Q|}{1 + N_{qw}}) + 1 \tag{4.2}$$

4. Classify vectors to identify intent

4.2 Feature Classification for Intent Identification

Feature classification is the next important step in the presenting speech intent identification process. It is essential to identify a classifier. In the literature, past research has used simple classifier models such as SVM and some ANN-based classifiers such as FNN and CNN models [38, 31, 32]. In addition to that, some advance ANN-based models that use RNN or LSTM are also good at feature classification [42, 43]. However, these RNN or LSTM layer-based classifiers require more data. Hence, the following classifier models were tested for feature classification.

- Support Vector Machines (SVMs)
- Feedforward Neural Networks (FNNs)
- Convolutional Neural Networks (CNNs)

An SVM model takes different hyperparameters. Among these, the kernel is a special parameter, and it represents the type of the boundary function suck as linear or polynomial. All of the parameters were considered to obtain the best SVM classifier. In the CNN models as described in Section 2.4, there are two variations 1D vs 2D. Both of these models were evaluated with different combinations of CNN layers. In addition to that, model hyper-parameters are optimized to obtain the best results.

Figure 4.7 depicts the overall system arrangement with a pre-trained ASR model and CNN based feature classifier model. The model contains two CNN layers and two sub-sampling layers. After these layers, there are one Dense layer and one Softmax layer. The Softmax layer is responsible for converting Dense layer output into normalized probability values that represent the probability of belonging intent category.

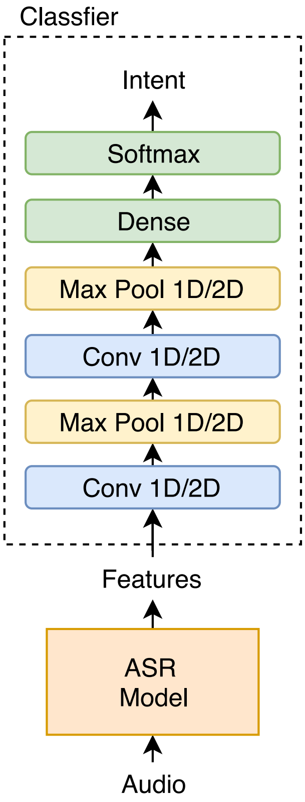

Figure 4.7: End-to-end system arrangement with CNN base classifiers

Chapter 5

EXPERIMENTAL SETUP

This chapter explains the experimental setup and the evaluation process of the methods presented in the Section 4.

This chapter is divided into two sections. First section introduces the datasets used in this study. Second section introduces the experiment details and parameters of the models.

5.1 Datasets

Intent annotated publicly available speech datasets are not so common even for English. It is even more difficult to find such datasets for other languages. Almost all the languages allow to express something in different ways. Hence, there can be several ways to express a particular intent. Buddhika et al. [38] used a Sinhala dataset collected by them. This dataset includes Sinhala speech data related to six different intents in the banking domain. Researchers have accounted this and each intent has some inflections. Data collection has been done by a crowdsourcing web platform [44]. Contributors can access this website via mobile phones to record audio. When accessed, they are asked to read a sentences form the dataset for recording. Since mobile phones are. used all the audios were recorded in the open environment and contain usual random background noises. Original dataset by Buddhika et al. [38], contained 10 hours of speech data from 152 males and 63 females students. All the contributors are in the age between 20 to 25 years. In this study, this dataset was re-evaluated since there were some miss-classified, too lengthy and erroneous speech queries. The final data set contained 7624 samples totaling 7.5 hours.

In addition to the Sinhala dataset, another Tamil dataset was collected for this research. It also contains similar intents presented in the banking domain similar to Sinhala speech data. Additionally, the Tamil dataset contains codemixed speech queries, which contain some English terms. This dataset was collected using voice-web [1] web-based speech data collecting platform. This is also developed to crowdsource speech data. The Tamil dataset contains 400 data samples totaling 0.5 hours of speech. There were 40 contributors including both male and female.

Table 5.1 provides the statistics of the Sinhala and Tamil datasets. In the table 5.1, "I" column represent the number of inflections and "S" column represent the number of samples in the dataset. Table 5.2 provides different inflections for Sinhala and Tamil language.

Table 5.1: Details of the datasets

(I-Inflections, S-Number of samples)

Intent	Sinhala		Tamil	
	I	S	I	S
1. Request Acc. balance	8	1712	7	101
2. Money deposit	7	1306	7	75
3. Money withdraw	8	1548	5	62
4. Bill payments	5	1004	4	46
5. Money transfer	7	1271	4	49
6. Credit card payments	4	795	4	67
Total	39	7624	31	400
Unique words	32		46	
Size in hours	7.5		0.5	

Dataset is publicly available[2].

5.2 Experimental Setup

As described in Section 3.3, Buddhika et al. [38] have presented some preliminary approach to speech intent classification, which uses MFCC as the features. This method is used as a benchmark in this study and performance is evaluated for both datasets.

[1] https://github.com/mozilla/voice-web

[2] http://rtuthaya.lk/sinhala-tamil-speech-intent-dataset/

Table 5.2: Selected Inflections

Intent No.	Language	Inflections
1	Sinhala	මගේ ගිණුමේ ශේෂය කීයද? (magē giṇumē śēṣaya kīyada) ගිණුමේ ශේෂය කීයද? (giṇumē śēṣaya kīyada) ශේෂය කීයද? (śēṣaya kīyada) මගේ ගිණුමේ ඉතිරිය කීයද? (magē giṇumē itiriya kīyada) ගිණුමේ ඉතිරිය කීයද? (giṇumē itiriya kīyada) ඉතිරිය කීයද? (itiriya kīyada) මට මගේ ගිණුමේ ශේෂය දැනගන්න පුලුවන්ද? (maṭa magē giṇumē śēṣaya dænaganna puluvanda) මට මගේ ගිණුමේ ඉතිරිය දැනගන්න පුලුවන්ද? (maṭa magē giṇumē itiriya dænaganna puluvanda)
2	Tamil	காசு deposit பண்ண வேண்டும் (kācu deposit paṇṇa vēṇṭum) காசு போட வேணும் (kācu pōṭa vēṇum) காசு போடோணும் (kācu pōṭōṇum) Cash போடோணும் (cash pōṭōṇum) Deposit போடோணும் (deposit pōṭōṇum) காசு வைப்பு செய்ய வேண்டும் (kācu vaippu ceyya vēṇṭum) Deposit காசு போடோணும் (deposit kācu pōṭōṇum)

AMDTK framework[3] implemented by Ondel et al. [35] is used to discover unsupervised acoustic units. The generated unit sequence is converted to vectors according to the process described under Section 4.1.2. The best performance is obtained when M is 67 and 43 in the acoustic unit discovery process respectively for Sinhala and Tamil languages. Using these units vocabulary size of 61 and 49

[3] AMDTK - `https://github.com/iondel/amdtk`

are selected for Sinhala and Tamil, respectively. SVM with a linear kernel is used to classify generated feature vectors. The performance is evaluated using 5-fold cross-validation.

The next method is feature transfer learning with ASR models. However, training an ASR model on a large dataset requires high computational power. Hence already trained openly available ASR models are used for the experiments. To get the character probability features, DeepSpeech (DS)[4] and DeepSpeech2[5] models. The DS model has been trained on the Common Voice American English corpus, and reports a 11% WER on the LibriSpeech clean test corpus. The DS2 model is trained on LibriSpeech data and report 12% WER. To get phoneme based probability values, the pre-trained ASR model of Lugosch el at. [4][6] is adapted. This model is trained on the LibriSpeech English corpus [4].

Then the probability features are extracted using the ASR models, which are used on the classifier models for training. For character probability features, a character set of $\{a, b, c, .., z, space, apostrophe, blank\}$ is used, since it is trained on English speech. The phoneme set had a set of 42 symbols that includes the ARPAbet English phoneme set (39 phonemes), and 3 non-speech annotations [4]. Since there is a limited amount of data, 5-fold cross-validation is employed to measure the overall classification accuracy. Models are trained up to the maximum accuracy without getting over-fitted into the training data set. Additionally, a Bayesian optimization-based algorithm is employed for hyper-parameter tuning [45]. The optimization algorithm is employed with 500 iterations to select the suitable hyper-parameters, which improves the overall accuracy. This was very significant for the CNN model parameters such as the number of filters and kernel sizes. For the SVM models, a linear kernel is used after experimenting with several different kernels types (Polynomial, Radial Basis Function (RBF)). Table 6.1 shows the final overall classification accuracy of different classifier models and a comparison between baseline method.

[4] https://github.com/mozilla/DeepSpeech

[5] https://github.com/SeanNaren/deepspeech.pytorch

[6] https://github.com/lorenlugosch/pretrain_speech_model

Further, the overall accuracy change with respect to the number of available training samples is evaluated. For that, a random sample with a particular size is drawn from the dataset and intent classification accuracy is evaluated via 5-fold cross-validation. This process is carried out 20 times to get the average accuracy for the given sample size and the task was performed on the Sinhala dataset since it contained more than 5000 samples. Chapter 6 summarizes these results.

Chapter 6

RESULTS AND DISCUSSION

This chapter presents the results and a discussion of the experiments presented in the Chapter 5. The first section provides the experiment results. The second section provides a detailed discussion based on the obtained outcomes.

6.1 Results

Table 6.1 presents the overall results for different methods presented in the Chapter 4. In the table "Features" column provides information about the different feature generation approaches. The "Classifier" column provides details about different classifier models used to identify intent from respective features.

Table 6.1: Summary of results

	Features	Classifier	Accuracy Sinhala %	Accuracy Tamil %
Previous	MFCC [38]	SVM	48.79	29.25
		6L FFN	63.23	26.98
	Unsupervised AUD	SVM	32.46	21.73
Proposed	DS Character Prob	SVM	70.04	23.77
		1D CNN	93.16	37.57
		2D CNN	92.09	76.30
	DS2 Character Prob	SVM	56.73	24.81
		1D CNN	88.49	38.02
		2D CNN	84.35	62.48
	Phoneme Prob	SVM	78.21	49.83
		1D CNN	**97.31**	**81.70**
		2D CNN	94.16	76.28

In the experiments, methods proposed by Buddhika et al. [38] and Unsupervised AUD approaches are used as a benchmark to measure the performance of new methods. When observing results presented in the Table 6.1, proposed methods give better intent detection accuracy compared to previously proposed methods.

For the character probability feature, two different ASR models were used to obtain features, DeepSpeech, and DeepSpeech 2 ASR models. Even though DeepSpeech 2 is the improved version of the DeepSpeech model, it produces lower accuracy for the low-resource speech intent detection. However, the best performing feature, the phoneme probability values outperform all other methods. Additionally, when we consider different classifier models, phoneme based feature produces the best results.

When it comes to feature classifier models and different proposed features, for the Sinhala dataset, 1D CNN gives the best intent identification accuracy with the character probability feature. In contrast to this, for Tamil speech data, 2D CNN gives the best results. However, with the Phoneme probability feature, it is difficult to observe such variance by observing results in the Table 6.1 and 1D CNN classifier models always produce the best results.

Figure 6.1 represents the overall accuracy change with the number of available data. Here, connected lines represent the overall accuracy change in the Sinhala data. Disconnected dots on the dashed-line represent the Tamil dataset experiment results for a 400 data sample. DeepSpeech model based character features and phoneme probability values have been used to generate these results.

For character-based features, it is possible to observe that when there is a limited number of training data, the 2C CNN classifier provides better intent identification accuracy. However, the difference between 1D and 2d CNN classifiers goes to zero when there are around 3000 samples and after that, 1D CNN can give better results for datasets with large samples. With phoneme probability features, it is not possible to observe this type of a classifier performance variance with number of available training data. However, 1D CNN classifier model always outperform the 2D CNN model.

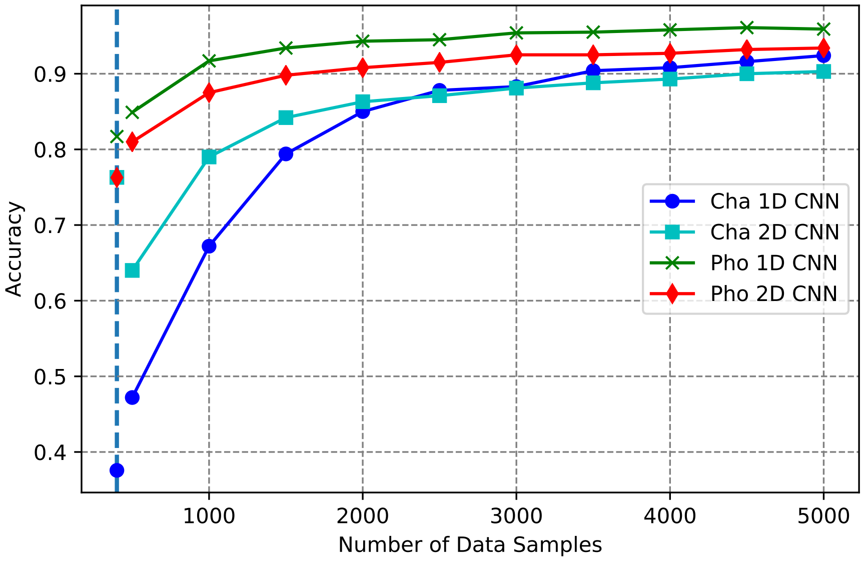

Figure 6.1: Overall accuracy change with the samples size

Connected dots - Sinhala, Dots on the dashed-line - Tamil

6.2 Discussion

Overall results presented in Table 6.1, and the graph in Figure 6.1 emphasize that phoneme probability features are more effective for speech intent identification compared to the rest of the features. For Sinhala and Tamil datasets, the proposed method achieves an overall accuracy of 97.38% and 81.70%, respectively. Further, these values indicate the usefulness of phoneme probability features despite the targeted low-resource language. According to the Figure 6.1, having 500 is enough to reach up to more than 80% accuracy. It needs more than 1000 data samples to achieve similar results using character probability features.

Unsupervised acoustic unit discovery feature performance is low, and accuracy values are behind the baseline method. This must be due to limited speech data. Character probability features from both ASR models perform better compared to baseline. However, the DS feature results are superior compared to DS2. Here, the obtained pre-trained ASR models are trained on different datasets and the dataset used on DS is larger than the DS2 training dataset. Generally, character probability features are better than both baseline and unsupervised features.

In the Figure 6.1 it can be seen that the accuracy for the Tamil dataset also lies close to the Sinhala dataset trend line. However, when examined closely, we can find one exception, 2D CNN shows a similar accuracy for both character and phoneme features in Tamil data. In the Tamil dataset, 61% of the sentences contain code-mixed speech queries with at least 1 English word. This can be a reason for such a higher result since feature generating ASRs are trained in English. In contrast to this, we cannot identify such anomaly results with phoneme features or 1D CNN character feature results. Further, this can happen because of having proper hyper-parameters for the 2D CNN character model. Because of the limited data, it is difficult to identify the effect of code-mixing. However, in general, phoneme probability features give better results compared to character probability features regardless of the language.

Table 6.2 and 6.3 present the most probable character/phoneme sequences for few selected sentences in Sinhala and Tamil languages, respectively. For better understanding, we present 39 phonemes with their IPA (International Phonetic Alphabet) notation.

Table 6.2: Most probable characters and phonemes for Sinhala utterances

Spoken Utterance	Character Sequence	Phoneme Sequence
ශේෂය කීයද (śēṣaya kīyada)	she is hegeal she he khe o shhe shegil shhe as shhakin ane she che a kidedd	ʃʃʃiiiɪʃʃiiʌʌkkiiiiiilllɝɹɹɝ ʃʃʃʃʃeɪeɪeɪʃʃʃiʌʌkkiiiiiɪðð ʌʌ ʃʃʃʃʃeɪeɪeɪʃʃʃeɪeɪikkkjiiuuunlʌn ʃʃʃʃʃieɪeɪeɪʃʃʃiiiɪkkkiiiiiʌnðʌʌllllɹ aɪ kkkʃʃʃʃʃiiiɪʃʃɪihæækkkhhhhɪɪɪdɪɪɪtθfff
ගිණුමේ ඉතිරිය කීයද (giṇumē itiriya kīyada)	give te matyy aga f giv the madiae tit go know may did he acey athe gien_me didd oge in did me mak ti takeo tit	dn ɪɪɪinʌʌmmeɪeɪeɪeɪdddɪdiiieɪggiiiiɪʌððʌtʌʌ huuuu gggɪɪnnʌnnneɪeɪeɪeɪdddiiiiiikkiiiiiiððʌʌʌθ ngggɪɪnnoʊmmmeɪeɪeɪeɪeɪeɪddɪɪttiiɪɪkkkiiiiɪɪɪɹððɛɛɛɹt nnn nniɪɪnʌnnnieɪeɪeɪkkkɪdiiiighiiiiiinnnn ððɪɪnnnnnnnnneɪeɪeɪeɪnndtttɪttiiiitttkuuuuuuuuvvɪɪɪtt
මුදල් තැන්පත් කරන්න ඕනේ (mudal tænpat karanna ōnē)	what a tempartan nnowi wo a themberber corono wilthose tem but coulo knnow onn would a dem but grannoi who adempat carannonin	wwl bwʌddoʊoʊætæææmnɑɑʌkkʌʌnnnoʊoʊɔɔnnii wwwwwððoʊoʊoʊoʊððææmmblʌʌkkkʊɝʌnnnoʊɔɔɔnnnn wwwwððɝɝɝɝððɛɛmmbʌʌttkkkʌtɑʌnnnaʊaʊɑɑɑɑɑnʌʌɜ wwwððaʊaʊaʊtðɛɛmmbʌʌtttʌtʌʌnnnoʊoʊoʊoʊnniii hhwwɹdʌoʊlddʌnmpʌʌpkggtɝʌnnnɔɔɔɹniiŋ
බිල් ගෙවීමක් (bil gevīmak)	bettte gave ye uckk belli give emac benn geni muck be give emma begive uckk	bbihʌɪɪɪɪmiiimɑɑʌkkkkkk ɪs bbiiieɪvgjɪɪŋŋiiiimmɑʌkkkkkk bbɪɪnnngɪɪðiiimmʌʌkkkkkkk bwbiiiiggɪɪɪðiiiimmɑʌʌkkkkkkk bbbbbiitʌʌʌdggɪɪɪdiiiiimmɑɑɑkkkkkk

When examined closely, it is possible to identify some common patterns in the

Table 6.3: Most probable characters and phonemes for Tamil utterances

Spoken Utterance	Character Sequence	Phoneme Sequence
Account balance என்ன? (Account balance eṉṉa?)	a gont belancint a goiinman in an a con bbalin sen mat a gonball intin now a gont milonss in e	æææţkgɔɔnnntmmmɛnʌʌnsssɪɪnnnnɪɛʌtt t ʌʌʌdgggoʊoʊoʊoʊnnnnʌʌʌʌʌnndzz_ɛɛnnnɹɝɝɝɝɝ æædkgɔɑɑnnbʌʌtʌnnzzɪɪnnnmmæææn ʌʌʌʌgggɑɑʌnmʌʌʌʌnnttɪɪnnnæætɝɝt ɹɹ ɔɔɔllll ɪɪɪtggɔnnnttɹɹllʌnnssssɪɪɪŋnnðʌʌ aɪaɪ
காசு போட வேணும் (Kācu pōṭa vēṇum)	canssiporewo as case olover o caausiborover carreolewar non cassof ordonum	æ kgɑɑɑɑɑssʌbbɔɔɔɹnnoʊoʊoʊoʊʌunnnnn ææææ hææ æʌʌsssʌn kkɑɑɑɑæssʌʌd ɔɔɔɔɹɹɛʌllɝɝʌʌnnʌʌvʌ kkɑɑɑsʌvvɔɔɹdoʊwɝɝɝɝɝ hlul kkɑɑɑɑsssʌʌɝtɹɔɔɔɹɹdɛɝɝɹɹɛeɪeɪeɪeɪnnoʊoʊoʊoʊnnn ææææææsssʌʌvpbɔɔɔɹɹddoʊɔɔntɝɝmmm
காச மீளெடுக்க வேணும் (Kāca mīḷeṭukka vēṇum)	casim mo a acoonno casston wane y covern cass and miler the covene cason moded gorl casaler coer	aɪaɪaɪaɪaɪzzzʌoʊʌmmllʌnaɪaɪdʌʌggoʊoʊoʊoʊnnoʊoʊnnn kkɑɑɔɹssʌnwwlhɹɝddʌkkkoʊoʊoʊoʊɝnnnnn gææɑɑɑsssʌnmwʌlllloʊoʊoʊddʌkkkʌɝɝɝɝoʊaɪnʌʌnv kkɑɑɑɔɑsssʌnnwwllidɪɝðʌdgggʊʊllɝɝɝnʌʌnv ɪ kkææææsssʌmmwʌlllloʊoʊnʌʌkggʌlllʌnnnaɪ
Bill காசு கட்டவேணும் (Bill kācu kaṭṭavēṇum)	belka e go teo dilcos i arton biass with at teronnar biis castsi cut teverm belicasiva torn	bbbiiiiiggææææssʌggɑʌdddeɪeɪeɪeɪeɪeɪnʌoʊoʊ baɪaɪi bbbɪɪdʌʌkkɑɑɑssðikɑɹttbɛɹɹɝɝnnnnnn bbðɪɪʌɹkkkæææssʌʌððætttðɛʌɹɝɝɝnnʌʌnn bbbiiiɪɪɪnkkkɑɑɑɑssʌʌnkkʌntnɛɛɛvvɝɝɝɝnnʌnmm bbbbɪilliiggɑɑɑsssʌðʌaɪʌddoʊoʊoʊɑʌnʌnn

sequences presented in the Tables 6.2 and 6.3. Most of the time these patterns do not occur sequentially, there are some other symbols in between them. The intent identification model trained on probability features tries to identify those hidden patterns. These sequential patterns can be affected by the language model of the high resource training language. Here, the ASR models used to generate features are based on the Recurrent Neural Networks (RNNs) and they are capable of language modeling [46, 47]. Therefore the ASR models try to predict character or phoneme sequences as observed in the training language (English in this case).

This effect is more visible in character probability outputs in the Table 6.2 and 6.3. Here, the first few characters with corresponding sounds have been detected. Sometimes it has predicted the English words with a similar sound. However, when it comes to the middle and end of a sequence, it is difficult to find any patterns and all look random. This is quite different for the phoneme sequence. If we inspect the generated symbol sequence, even in the middle we can find some patterns. Hence, phoneme-based features give a better representation compared to character-based features for a given audio query. Further, there is higher classification accuracy for phoneme features. It is possible to assume that these patterns affect this.

The next visible effect on results is the performance difference of the 1D and 2D CNN models. When there is limited data, 2D CNN outperforms 1D CNN in character probability features. This is changed when the training data size increases. With phoneme features, 1D CNN always outperforms 2D CNN. When the feature visualization for intensity points (most probable symbol) is examined, it is visible that these values change quite rapidly in the character feature map compared to phonemes. Additionally, it is possible to identify visible patterns inside the generated phoneme sequences (in Table 6.2 and 6.2). Hence 1D CNN can perform better with phoneme features. There are more rapid distortions in character features. Hence, 2D CNN may be more useful for character probability features.

Chapter 7

CONCLUSION

This study investigated the possibility of direct speech intent identification for low-resource languages. The existing processing pipeline depends on ASR to transform speech into text. However, creating a robust ASR requires a large speech dataset, which makes it difficult to train for low-resource languages.

This thesis provided a solution for direct speech intent identification for low-resource languages by incorporating transfer learning. The recommended solution consists of two-steps. The first step involves the generation of features that can represent semantic information in a given speech query. The second step involves the classification of the obtained features. Feature generation form pre-trained ASR showed better representation ability and two such different features, character and phoneme probability features are obtained. Phoneme feature combined with the 1D CNN classifier model showed the best results in intent detection and showed more than 80% accuracy using only 500 data samples for both Sinhala and Tamil languages.

Compared to the character and phoneme features, intent classification from the unsupervised feature (AUD), is not effective for the chosen limited datasets. Additionally, proposed approaches showed more than 30% increment of intent identification accuracy compared to benchmark methods. Even though the phoneme-based feature is the best, the character-based feature also shows competitive results for a slightly larger data amount.

It is possible to observe that intent classification performance is varying with different CNN models. The character-based feature shows better results with the 2D CNN models for limited data. However, when the size of the training data increases 1D CNN classifier model produces better results. For the phoneme feature, the 1D CNN classifier is the optimal one. It is possible to assume continuous patterns appearing in phoneme features make it more effective to use 1D CNN as the classifier. Further, these patterns in phoneme-based features, help to achieve better overall intent identification results.

In conclusion, the phoneme feature generated from pre-trained DNN-based ASR models combined with the 1D CNN model is much more effective for low-resource speech intent identification.

Chapter 8

FUTURE WORK

This study can be extended in two different directions in the future. One path is to experiment performance variance with the used high-resource languages. In the experiments, only English is used. English, Sinhala, and Tamil belong to three different language families. However, languages in the same family share similar phenoms and sounds largely. Hence there is a possibility of improving the final outcome.

Another path is the use of untranscribed speech data in the automatic acoustic unit discovery process. With the availability of the internet and digital media, it is easy to collect unannotated speech data. Having a larger amount of data can improve the learning of better sound representations in used unsupervised algorithm.

References

[1] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

[2] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, pages 173–182, 2016.

[3] Jui-Ting Huang, Jinyu Li, Dong Yu, Li Deng, and Yifan Gong. Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7304–7308. IEEE, 2013.

[4] Loren Lugosch, Mirco Ravanelli, Patrick Ignoto, Vikrant Singh Tomar, and Yoshua Bengio. Speech model pre-training for end-to-end spoken language understanding. *arXiv preprint arXiv:1904.03670*, 2019.

[5] Rakesh Gupta. Free-speech command classification for car navigation system, January 22 2013. US Patent 8,359,204.

[6] Ashwin Ram, Rohit Prasad, Chandra Khatri, Anu Venkatesh, Raefer Gabriel, Qing Liu, Jeff Nunn, Behnam Hedayatnia, Ming Cheng, Ashish Nagar, et al. Conversational ai: The science behind the alexa prize. *arXiv preprint arXiv:1801.03604*, 2018.

[7] Laurent Besacier, Etienne Barnard, Alexey Karpov, and Tanja Schultz. Automatic speech recognition for under-resourced languages: A survey. *Speech Communication*, 56:85–100, 2014.

[8] Ye-Yi Wang, Li Deng, and Alex Acero. Spoken language understanding. *IEEE Signal Processing Magazine*, 22(5):16–31, 2005.

[9] Mirco Ravanelli and Yoshua Bengio. Interpretable convolutional filters with sincnet. *arXiv preprint arXiv:1811.09725*, 2018.

[10] William Song and Jim Cai. End-to-end deep neural network for automatic speech recognition. *Standford CS224D Reports*, 2015.

[11] Ronan Collobert, Christian Puhrsch, and Gabriel Synnaeve. Wav2letter: an end-to-end convnet-based speech recognition system. *arXiv preprint arXiv:1609.03193*, 2016.

[12] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[13] Paul Lamere, Philip Kwok, Evandro Gouvea, Bhiksha Raj, Rita Singh, William Walker, Manfred Warmuth, and Peter Wolf. The cmu sphinx-4 speech recognition system. In *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP 2003), Hong Kong*, volume 1, pages 2–5, 2003.

[14] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011.

[15] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210. IEEE, 2015.

[16] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4960–4964. IEEE, 2016.

[17] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM, 2006.

[18] Andrew Rosenberg, Kartik Audhkhasi, Abhinav Sethy, Bhuvana Ramabhadran, and Michael Picheny. End-to-end speech recognition and keyword search on low-resource languages. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5280–5284. IEEE, 2017.

[19] Özgür Cetin, Madelaine Plauché, and Udhaykumar Nallasamy. Unsupervised adaptive speech technology for limited resource languages: A case study for tamil. In *Spoken Languages Technologies for Under-Resourced Languages*, 2008.

[20] Jonas Lööf, Christian Gollan, and Hermann Ney. Cross-language bootstrapping for unsupervised acoustic model training: Rapid development of a polish speech recognition system. In *Tenth Annual Conference of the International Speech Communication Association*, 2009.

[21] Ngoc Thang Vu, Franziska Kraus, and Tanja Schultz. Rapid building of an asr system for under-resourced languages based on multilingual unsupervised training. In *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[22] Julius Kunze, Louis Kirsch, Ilia Kurenkov, Andreas Krug, Jens Johannsmeier, and Sebastian Stober. Transfer learning for speech recognition on a budget. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 168–177, 2017.

[23] Ngoc Thang Vu, Florian Metze, and Tanja Schultz. Multilingual bottle-neck features and its application for under-resourced languages. In *SLTU*, 2012.

[24] Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*, 2018.

[25] Sibel Yaman, Li Deng, Dong Yu, Ye-Yi Wang, and Alex Acero. An integrative and discriminative technique for spoken utterance classification. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(6):1207–1214, 2008.

[26] Jinfeng Rao, Ferhan Ture, and Jimmy Lin. Multi-task learning with neural networks for voice query understanding on an entertainment platform. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 636–645. ACM, 2018.

[27] Xiaodong He and Li Deng. Speech-centric information processing: An optimization-oriented approach. *Proceedings of the IEEE*, 101(5):1116–1135, 2013.

[28] Dmitriy Serdyuk, Yongqiang Wang, Christian Fuegen, Anuj Kumar, Baiyang Liu, and Yoshua Bengio. Towards end-to-end spoken language understanding. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5754–5758. IEEE, 2018.

[29] Parisa Haghani, Arun Narayanan, Michiel Bacchiani, Galen Chuang, Neeraj Gaur, Pedro Moreno, Rohit Prabhavalkar, Zhongdi Qu, and Austin Waters. From audio to semantics: Approaches to end-to-end spoken language understanding. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 720–726. IEEE, 2018.

[30] Yuan-Ping Chen, Ryan Price, and Srinivas Bangalore. Spoken language understanding without speech recognition. In *2018 IEEE International Confer-*

ence on Acoustics, Speech and Signal Processing (ICASSP), pages 6189–6193. IEEE, 2018.

[31] Chunxi Liu, Jan Trmal, Matthew Wiesner, Craig Harman, and Sanjeev Khudanpur. Topic identification for speech without asr. *Proc. Interspeech 2017*, pages 2501–2505, 2017.

[32] Santosh Kesiraju, Raghavendra Pappagari, Lucas Ondel, Lukáš Burget, Najim Dehak, Sanjeev Khudanpur, Jan Černockỳ, and Suryakanth V Gangashetty. Topic identification of spoken documents using unsupervised acoustic unit discovery. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5745–5749. IEEE, 2017.

[33] Matthew Wiesner, Chunxi Liu, Lucas Ondel, Craig Harman, Vimal Manohar, Jan Trmal, Zhongqiang Huang, Najim Dehak, and Sanjeev Khudanpur. Automatic speech recognition and topic identification from speech for almost-zero-resource languages. *Proc. Interspeech 2018*, pages 2052–2056, 2018.

[34] Kate M Knill, Mark JF Gales, Anton Ragni, and Shakti P Rath. Language independent and unsupervised acoustic models for speech recognition and keyword spotting. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[35] Lucas Ondel, Lukáš Burget, and Jan Černocký. Variational inference for acoustic unit discovery. *Procedia Computer Science*, 81:80–86, 2016.

[36] Chunxi Liu, Jinyi Yang, Ming Sun, Santosh Kesiraju, Alena Rott, Lucas Ondel, Pegah Ghahremani, Najim Dehak, Lukaš Burget, and Sanjeev Khudanpur. An empirical evaluation of zero resource acoustic unit discovery. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5305–5309. IEEE, 2017.

[37] Aren Jansen and Benjamin Van Durme. Efficient spoken term discovery using randomized algorithms. In *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*, pages 401–406. IEEE, 2011.

[38] Darshana Buddhika, Ranula Liyadipita, Sudeepa Nadeeshan, Hasini Witharana, Sanath Javasena, and Uthayasanker Thayasivam. Domain specific intent classification of sinhala speech data. In *2018 International Conference on Asian Language Processing (IALP)*, pages 197–202. IEEE, 2018.

[39] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[40] Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Interspeech*, pages 498–502, 2017.

[41] Chia-ying Lee and James Glass. A nonparametric bayesian approach to acoustic model discovery. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 40–49. Association for Computational Linguistics, 2012.

[42] Yiren Wang and Fei Tian. Recurrent residual learning for sequence classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 938–943, 2016.

[43] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[44] Darshana Buddhika, Ranula Liyadipita, Sudeepa Nadeeshan, Hasini Witharana, Sanath Jayasena, and Uthayasanker Thayasivam. Voicer: A crowd sourcing tool for speech data collection. In *2018 18th International Conference on Advances in ICT for Emerging Regions (ICTer)*, pages 174–181. IEEE, 2018.

[45] James Bergstra, Brent Komer, Chris Eliasmith, Dan Yamins, and David D Cox. Hyperopt: a python library for model selection and hyperparameter optimization. *Computational Science & Discovery*, 8(1):014008, 2015.

[46] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*, 2010.

[47] Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *International conference on machine learning*, pages 1764–1772, 2014.