Exploring Model Level Transfer Learning For Improving Sinhala Speech Recognition

MCS3204

A.L. Nanayakkara

MASTERS OF SCIENCE IN COMPUTER SCIENCE
UNIVERSITY OF COLOMBO SCHOOL OF COMPUTING

2022

Declaration

I hereby declare that the thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis. This thesis has also not been submitted for any degree in any university previously.

Student Name: A.L Nanayakkara
Registration Number: 2019/MCS/062
Index Number: 19440626

[illegible] 07/02/2023

Signature of the Student & Date

This is to certify that this thesis is based on the work of Ms. A.L Nanayakkara under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by,
Supervisor Name: Dr. Ruvan Weerasinghe

[illegible] 08/02/2023

Signature of the Supervisor & Date

Acknowledgments

Firstly, I would like to convey my sincere gratitude to my supervisor, Dr. Ruvan Weerasinghe for his remarkable guidance throughout the research project. I'm grateful for his patience in listening to me and his profound knowledge and immensely valuable input were key factors in successfully completing this project. For this project I had to study several research articles, papers, blogs, and publicly available data in some websites. I would like to acknowledge all the authors of these publications and online articles.

I would also like to express my sincere thanks to all the staff members in the Language Technology Research Laboratory at UCSC and for those who contributed to the voice recording. Foremost, I would like to thank Mr. B.C Gamage and Mr. Masoud Parpanchi, for their support and suggestions throughout this project. I would also like to extend my gratitude for my colleagues who helped me in many ways to complete this project. Finally, I'm immensely thankful to all my family members. Their encouragement and support was what kept me going when the times were tough.

Abstract

Automatic Speech Recognition (ASR) is the process which accurately translate spoken utterances into its corresponding textual format. However, in ASR, it will only translate the given speech data into text and not worry on the semantic aspect on it. Through accurate ASR we can easily build an interface for the both illiterate and literate users. Anyway, ASR gives better results for the most widely used data rich languages likes, English and German, but not in the data scarcity languages like Sinhala. During past few years many researchers conducted several studies on developing more accurate ASR for Sinhala, but they failed to succeeded on it due to low resource problem. This project represents a study to build ASR system for a low resource Sinhala language, which is known as morphologically rich complex language. To tackle the data scarcity issue, we have used new mechanism called transfer learning. It is capable to transfer knowledge from data rich model to data scarce model.

We carry out several experiments on Sinhala speech recognition on DeepSpeech by considering various aspects such as applications on language optimizations, external scorer and data augmentations. Initially we start our experiments on transfer learning from pre-trained English to Sinhala without considering any data augmentation and achieved 22.92% in WER and 8.84% in CER. Later, when we applied data augmentation on the transfer learned model then it showed drastically reduction on WER and CER with compared to the initial models. It showed 17.19% for the WER and 5.9% in CER for the model which consists of 10% of reverb together with 30% of overlay augmented type with others on 40% in each by considering its default values and it is explained in this document.

Experiments were conducted for the Sinhala speech dataset gathered from Language Technology Research Laboratory at UCSC. It consists of 40 hours of data coverage including both male and female speakers, which were recorded with the support from Praat and RedStart tools. All the experiments conducted by using it with the external support on 4-gram language model which build on KenLM toolkit. Finally, in the user evaluation it gives fairly good results for our model.

Keywords: Recurrent Neural Network, Transfer Learning, Data Augmentation, Language Optimization, Speech recognition

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The subfield of Artificial Intelligence (AI), called Natural Language Processing (NLP), focuses on the understanding and interpretation of the given input speech and text data via analysis of slang, photonics, syntax, sentiments, and complex language patterns. The main goal of NLP is to provide a platform for machines to communicate with us in a way that humans are used to. In practice NLP is used in search queries, social media analysis, text to speech conversion, language translation, and information extraction. Today, NLP is widely used in virtual assistants, who answer user questions via text or audible speech. So, speech recognition is one of the most important tasks in NLP, because most users prefer to interact with the devices by using their speech commands than by typing or textual commands.

1.1 Automatic Speech Recognition

Automatic Speech Recognition (ASR) has been an emerging research area in Natural Language Processing field for a few decades. Through the ASR system, it will accurately translate the spoken utterance into a corresponding textual form as it shown in the figure 1.1. So, textual form can be either in terms of words or sentences or characters or syllables or phones. The main intention of speech recognition is to reduce the gap in human-machine or man-machine interaction

Figure 1.1: Automatic Speech Recognition

(Zajechowski 2014). As an example, in YouTube via closed captioning, an ASR engine which is located inside YouTube, it will automatically produce the corresponding transcript for the speech on audio clip in the given input video file. In addition to that, there are several front-end ASR systems which are famous in these days such as, Siri, Alexa, Cortana and Google Voice which help humans to do their day-to-day activities in a much more interactive, easiest and intuitive manner.

For a human, if he or she is familiar with the language then speech recognition is an easy task. Yet for a machine, it is a complex task, because they have to concern in a wide area or various aspects to achieve the best result. Sometimes it will be difficult because there are several sources of variability in the speech context and among them speaker characteristics are the most prominent one. It includes several other sub characteristics, such as accent, speed, or style of speech, it can be semi-spontaneous or continuous speech and background noises (Nadungodage & Weerasinghe 2011). Therefore, it's necessary to pay attention to these varieties to achieve a better output through ASR and it is quite complex time-consuming task.

ASR system architecture can be classified into two types, statistical ASR architecture and end-to-end (e2e) Deep Neural architecture. Until 2014, statistical ASR is known as the latest architecture for speech recognition, but later they focus on end-to-end Deep Neural architecture due to their higher accuracy (Gamage, Pushpananda, Weerasinghe & Nadungodage 2020). The major difference between these two architectures was the total number of models that they use for a speech recognition. In statistical ASR architecture, it uses three separate models as acoustics model, which understands its speech waveform and convert it into some discrete features or phone sequences, pronunciation model, which create a link between phones to the word through maintaining a simple pronunciation dictionary, and language model, specify the structure of those word order. When it comes to the end-to-end (e2e) ASR architecture, instead of having three different models it will use one which is the compression of all three models(Karunanayake, Thayasivam & Ranathunga 2019) (Gamage et al. 2020).

1.1.1 Automatic Speech Recognition for Sinhala

It can be identified that Sinhala is a morphologically rich language which contains of 40 an alphabet of 26 consonants and 14 vowels. In Sri Lanka majority of citizens, approximately 16.2 million, have used Sinhala as their mother language. Anyway, ASR gives much more accurate results for the most widely used languages such as English, French, Japanese, and German, but not for the low-resource languages likes Sinhala, Swedish and Urdu. Due to that fact, it is necessary to build an accurate ASR model for Sinhala but it difficult to create because of its complex language patterns (Karunanayake et al. 2019).

The written and the spoken language of Sinhala are significantly different. The written Sinhala grammar depend upon on gender, number, person and tense wise. Also, in spoken Sinhala depends on the regional variations and it can be classified as the Uva Province variant, Up Country variant, Southern variant and Sabaragamuwa variant. So, in order to achieve the best output results, it is necessary to consider larger vocabulary which covered almost all the words in spoken language which is time consuming difficult task (Nadungodage & Weerasinghe 2011). During the past few years, researchers carried out several experiments on developing Sinhala ASR through end-to-end Deep Neural architecture. Due to the data scarcity on the Sinhala language, it failed to give a predicted result accurate. So, it is necessary to pay more attention to developing a new pathway for analyzing spoken data on Sinhala with fewer data.

Then it introduces a new concept called transfer learning into the speech and language processing domain. The main objective of this concept is the idea of ”learning to learn” or ”try to learn from experiences or from some other existing model rather than gain knowledge from the scratch.” Many researchers concluded that transfer learning can be applied for low resource language processing with the support from data-rich language (Sarkar 2018). To handle drastic changes of speech signal variation such as language, gender, speaker, channel, and the environment we can use the experience or knowledge that we have gathered and learned previously from a model (known as source) to discover new patterns in a different model (known as target) through applying transfer learning concepts (Joshi, Zhao, Mehta, Kumar & Li 2020). By the way, transfer learning can be applicable in language processing focusing on transfer across the speaker or language or models.

During past few years, there are lots of experiments conducted on low resources languages likes Sinhala, through that it concludes that the Deep Neural approach is better than the statistical approach, but it gives a considerable Word Error Rate (WER). So, it is necessary to fine-tuning the model to reduce the WER as much as possible. For that, it is necessary to trained the models using end-to-end Deep neural architecture and select the best ASR architecture for the low resource language, likes Sinhala (Stoian, Bansal & Goldwater 2020). Espresso, Kaldi, DeepSpeech and Wav2letter are some of the most common tools which design for the purpose of implementing an ASR. Among them Mozilla’s DeepSpeech is an open source most used tool for speech related applications and it gives quality results or predictions. It comes with huge collection of documentation to carryout training on DeepSpeech and the system core is implemented with the help from C++ and python.

Due to that fact, through this research, I am interested to find out the applicability of transfer learning mechanisms for Sinhala by considering Mozilla’s DeepSpeech platform. Then select the best target model which gives better results for the Sinhala prediction by referring the knowledge extracted from pre-trained English model. Therefore, it is necessary to find the suitable transfer learning DeepSpeech architecture for Sinhala with the relevant parameters together with data augmentation to tackle the data scarcities on the target language.

1.2 Project Aim, Objectives and Research Question

The main goal of this research is to identify the applicability of transfer learning on Sinhala speech data for obtaining better accuracy than the existing DNN approach. We intend to achieve this through the following objectives,

- Identify the issues of end-to-end Deep Neural (e2e DNN) approach in Sinhala ASR.
- Identifying the most appropriate source DeepSpeech model in English language.
- Building the language model for the Sinhala.
- Identifying the most appropriate transfer learning regime to use in the e2e DNN based approach on Speech data for model-level transfer.
- Identifying the most appropriate transfer learning regime to use in the e2e DNN based approach on Speech data for data-level transfer.
- Identification of the best transfer learning approach of the two for Sinhala ASR.
- Evaluating the proposed approach by using proper evaluation metrics.

The following questions are considering as guidelines to fulfill the above-mentioned aim of the proposed project.

1. How to apply model level transfer to improve the performance of Speech Recognition on low resource language?
   Like it expresses in the section 1.1, statistical and e2e deep neural network architectures are the two main architectures in Automatic Speech Recognition (Gamage et al. 2020). Before 2014, statistical model architecture is the most prominent mechanism for ASR, but later researchers pay more attention to deep learning concepts due to their higher accuracy.
   It's a common fact that many ASR work fine with data rich languages such

as English, German, Russian, due to its availability of large speech corpus and the well-defined language model. When it comes to the low resource languages like Sinhala, we have to face lots of problems in the deep learning aspect due to its poor dataset and its complexity in the language structure. To improve the performances on Sinhala speech recognition, the best architecture is to use a deep learning-based technique, because it shows remarkable prediction on complex domains.
Accuracy of the deep leaning-based techniques highly depend on the richness in the given dataset. However, creating large dataset is impossible in most of the low resource languages due to its complexity. For that purpose, we are trying to introduce a mechanism to transfer knowledge from the well-defined DeepSpeech source model in English to the target model in Sinhala language via the suitable transfer learning technique.
It can be achieved via model level transfer (Asgarian 2020) with the additional support from the language model because size of the alphabet in source and target languages are not compatible with each other. Here, it's necessary to choose the appropriate technique for knowledge transfer from source to target in model level to avoid impact on negative transfer. At the model level, we can choose parameter-based or hybrid-based methods (Asgarian 2020). Then we can select inductive or unsupervised or transductive transfer learning to answer when to transfer knowledge from source to target.

2. How to select the most appropriate target model for Sinhala language in the transfer learning?
   By updating model parameters in the architecture of the desired DeepSpeech source model in English we can build different target models in Sinhala. Among them, it is necessary to identify the best suited model which gives the highest accuracy among other models. For that we can use data augmentation with the suitable amount on the selected sub augmented types with or without considering language model optimizations. The accuracy of the predicted transcription can be identified via the analysis of Word

Error Rate (WER), which is denoted as a total number of errors and it can be calculated as the summation of insertions, substitutions and deletions in each spoken words and Character Error Rate (CER), which is focused on how accurate our character identification through the desired model.

1.3 Significance and Justification of the research

Due to the various medical or other issues, nearly 1 million Sri Lankan citizens suffer from blindness which ultimately leads them to visual impairments. So, in such a situation it will be useful to have a proper system that control its operations through voice commands such as Siri and Cortana. However, all these systems are operated via common languages such as English, Chinese, and Korean, etc. It is necessary to have voice control software to help with the day-to-day activities in the countries which has low resource languages as their mother tongue. The majority civilians live in the Sri Lanka has illiteracy with English language and computing knowledge. Therefore, it is essential to build such system which controls by Sinhala.

In the computer science community, there are few completed and ongoing researches on speech recognition focusing on low resource languages, but they did not receive better accuracy due to its language complexity. Due to the low resource with the spoken utterances, researchers failed to identify the corresponding sound to text mapping, because speech data contains lots of variations such as speaker, noises, accents, and speech style, and it will increase the complexity of recognition. So, it is necessary to focus on all aspects of the speech data in the training process, which is time consuming and complex task.

Anyway, most of the researchers focus on doing experiments on low resource speech recognizes through statistical-based methods while considering three models; acoustic, pronunciation, and language model (Nadungodage & Weerasinghe 2011). As a result of the low accuracy of that approach due to the data scarcity issue, researchers focus on DNN methods, which give better accuracy than the statistical method by using single model (Karunanayake et al. 2019). Also, in Sinhala ASR has an issue with the creation of good language model with minimum perplexity due to its language phonological and morphological complexity. For creating such good model, it is necessary to have a domain expert contribution, which known as linguistics, and it is hard to do because for that we need lots of linguistic data and time (Wang, Wang & Lv 2019). In e2e architecture it resolved that problem by automatically creating language model for the target language in the training process (Hannun, Case, Casper, Catanzaro, Diamos, Elsen, Prenger, Satheesh, Sengupta, Coates et al. 2014). In addition to that, current e2e ASR system shows better results of using additional language model as a plugging for the model which trained with the e2e architecture including important language specific features (Hannun et al. 2014). So, it is a crucial task to create the most appropriate language model for our target model in the transfer learning.

Still, it has a considerable WER concerning the output of the e2e Deep Neural approach. Then it is necessary to reduce that error rate to achieve better performance, but the issue is size of the available dataset. Transfer learning is a solution for that problem. Because through that we can use the existing models or source model knowledge for building new knowledge to another model which is known as the target model. Through that, we can transfer the knowledge that we learned from the already build e2e Deep Neural model to sub-models.

It is necessary to create a proper Speech model for the Sinhala language to help the visually impaired community. Not only that, through a good Speech Recognizer linguistics can carry out experiments to create a much more realistic model, and then we can use it as a voice input for existing software. Overall, the main significance of this research is to provide a way of improving the performance on Speech Recognition by reducing the WER and CER in the Sinhala language by transferring the feature extracted from the source model to the desired target model via proper transfer learning technique.

1.4 Research Methodology

The overall performance of ASR will be relying on the richness of our collected dataset. Therefore, it is necessary to have a proper dataset. There are already available speech datasets at Language Technology Research Lab (LTRL) at UCSC which includes 40 hours of Speech data on Sinhala languages. However, that data set was collected through two different software as Praat, which is known as a free software package that targeting on speech analysis at the phonetic level, and Redstart, which comes as a recording software in the MaryTTS framework. When considering those two datasets, 45 persons are involved with Praat recording and among them 31 of them are female. Not only that, 78 persons contribute to Redstart recording and among them 55 are female (Gamage et al. 2020). Therefore, our dataset contains both male and female voices to maintain the gender balance. However, these datasets are noise-free datasets, and according to the researchers' observation, they mentioned it is better to have noisy data as the training dataset in the DNN model. Due to that fact, we have included some noisy spoken dataset for the model training.

When it comes to the Sinhala language, it is considered a morphologically rich low resource language. Due to that fact, we do not have many datasets available on Sinhala. Then it is necessary to find out a proper mechanism to build a speech recognition system via deep learning without worrying about the corpus size of Sinhala speech data. To tackle that problem, the most suitable approach is to use transfer learning from one language (or data rich language) to another language (low resource language).

Proposed methodology can be represented in three stages.

1. Developing language model for Sinhala language
   In end-to-end deep neural ASR model does not contain three different sub models such as acoustic, pronunciation and language model. Instead of that, it uses single model which combined all those models in to a one bundle. When building a DeepSpeech ASR a language model helps to increase the accuracy of the result (Hannun et al. 2014). According to the literature there is no any pre-existing language model available for Sinhala built with the Mozilla's DeepSpeech, so we had to create it from the scratch by using our LTRL speech corpus. With the compatibility of the Mozilla's DeepSpeech we created our 4-grams language model or an external scorer for Sinhala with the support of KenLM toolkit, which comes in built with the core model and feed it as external input for the model training.

2. Developing e2e neural ASR model for Sinhala language by using DeepSpeech
   DeepSpeech is an open-source end-to-end ASR toolkit, which contains the Recurrent Neural Network (RNN) as a core engine to generate text results from the given input spoken data. Here it contains three major sub components in the DeepSpeech architecture including MFCC feature extraction, acoustic model and hypothesis search. The basic DeepSpeech architecture contains six layers and the given raw audio signals fed into the three feed-forward fully connected layers with the ReLU activation function. Then it passes through another three layers and generate the text transcription for the provided audio clip with or without considering the augmentation (*Welcome to Deepspeech's documentation* n.d.).

3. Evaluating the WER and CER for the developed model and calculate the quality of the output
   There is already Sinhala speech dataset available at the LTRL and we are using it for the training and testing purpose of our ASR. Here, it uses word error rate (WER) and the character error rate (CER) to analyze the overall quality of the text transcription of given audio data in our desired language. Through that we can clearly identify the best DeepSpeech architecture for the Sinhala ASR. In addition to that, we can use comparison between the e2e Sinhala ASR model together with the transfer learning model to compare the results.

1.5 Scope of the Research

In Scope

The proposed approach intends to cover the following topics throughout the research timeline,

- Collecting the appropriate source and target dataset with its labels.

- Exploring the different approaches of homogeneous transfer learning techniques used in the Automatic Speech Recognition systems.
- Exploring the different techniques used in transfer learning in model level transfer.
- Identify the best DeepSpeech architecture for the source and target model.
- Enhancing the performances of the 4-gram word-level Sinhala language model by calculating the optimized values.
- Analyze and fine-tuning the target model by using data augmentation types.
- Compare the traditional e2e model (baseline) performances concerning the transfer learning method.
- Compare the transfer learning model performances concerning the data augmented transfer learning method.

Out Scope

Following things are not considered during the research period,

- Not focusing on selecting the best toolkit for creating the source model. Because here we are mainly interested in using DeepSpeech, due to its flexibility.
- We are not going to consider accents, gender-based characteristics on speech data.
- This research not going to consider Tamil language speech data, even though it's one of the languages in Sri Lanka.

1.6 Summary

The rest of the thesis is organized as follows; Chapter 2 describes the background knowledge on DeepSpeech model building process, Chapter 3 describes the existing studies related to our research area by considering their approaches and conclusions. In Chapter 4 explains our methodology which we have followed for the creation on Mozilla's DeepSpeech ASR model for Sinhala and in Chapter 5 shows the results which we obtained throughout the research under several experiments. Finally, in Chapter 6 provide the conclusion we arrived at the end of the research study including approaches for the future implementations.

Chapter 2

Background Study

2.1 Artificial Neural Network

Neural networks have been used with the predictions on many domains likes, speech recognition, face recognition, real-time translations, healthcare, and marketing. The Artificial Neural Network (ANN) can be identified as a graph based functional unit or the base of a Deep Learning. Deep learning is a sub field of machine learning where set of algorithms imitate the behavior of human brain in order to solve the complex problems effective. An ANN includes set of nodes, which are grouped together and represents them in layers in a directed acyclic graph (DAG). Those layers are starting from the input layer, then connect with the arbitrary number of hidden layers and finally joined with the output layers. Figure 2.1 shows a simple neural network which consists of an input layer, followed with a single hidden layer and end up with an output layer. A neural network starts its functioning when some input data values are fed into the input layer. Then those data processed via arbitrary number of hidden layers to produce the desired output in an output layer.

Feed Forward Neural Network, Recurrent Neural Network, Long-Short Term Memory (LSTM) network, and Boltzmann Machine are different types of neural networks. Among them, feed-forward neural network is the simplest and easily understandable one, and it will connect every node in one layer with every

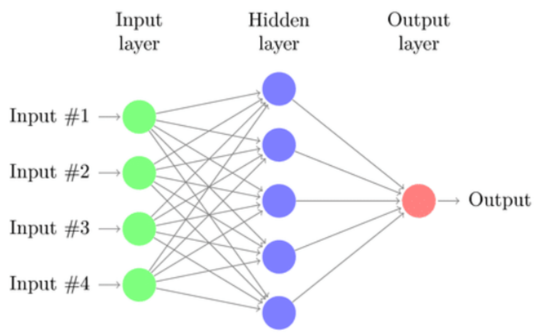


Figure 2.1: Simple ANN model

node in next layer like it shown in figure 2.1. Whatever the nodes in one layer relate to its next layer via the connection and each one has a weight value associated with them to determine the impact of the values passing through it. These weights are directly affected with the output, so during the training process it will optimize these weight values until it does the accurate predictions (Goodfellow, Bengio & Courville 2016).

Looking at an arbitrary node in the directed acyclic graph, it is input will be the weighted sum of the output in the preceding layer and then it will pass through the activation function before it send as an input to the next layer. Here, that activation function does operation to transform or change the weighted sum to some numerical value between some lower and upper value. There are several activation functions available such as Sigmoid, ReLU, ELU, tanh and Maxout. The Rectified Linear Unit (ReLU) is the most used activation function and it is used in DeepSpeech architecture as well and it represents in equation 2.1 (*Welcome to Deepspeech's documentation* n.d.).

$$F(x) = max(0, x) \tag{2.1}$$

In ReLU it will transform the inputs into either 0 or its maximum value. As it shown in the figure 2.2, it clearly depicts that if the given input is a negative value, then it will transform into 0 and if not then it will give a corresponding output for the given input value. Overall idea is if the neuron is more positive then

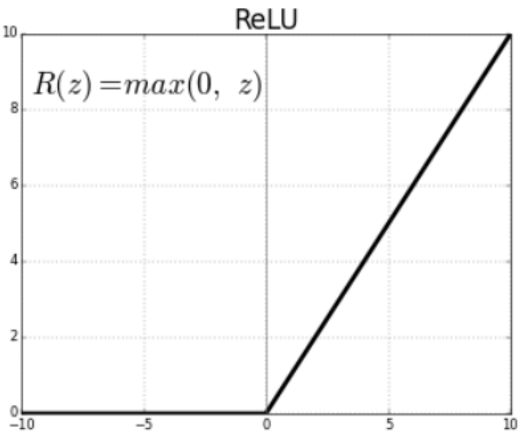


Figure 2.2: ReLU activation function

it will get a higher chance of getting activate than the negative ones (Sharma, Sharma & Athaiya 2017).

2.1.1 Training in ANN

However, in the output layer it is common that not to use activation function and instead of that it will just pass the raw values comes from the last hidden layer. Therefore, when we train an ANN model, we apply loss function at the output layer for the purpose of calculating a loss or a cost to measure the correctness of the desired output compared to the ground truth. Then, that calculated loss is used by an optimizer to update the weights of connections in the graph in order to improve the performances in the future. For that purpose, there are several optimizer available, among them the Adam optimizer is the most widely used one. However, in DeepSpeech it uses Connectionist Temporal Classification (CTC) loss function (Graves, Fernández, Gomez & Schmidhuber 2006). When working with the audio or text data, it is necessary to take sequence of input data to produce an output because current value rely on the preceding values and due to that reason in DeepSpeech it uses CTC loss function.

An ANN model training start with inputting data and it can be done in two ways, do it one data point at a time or do it as batches, called it as batch size. Once we go through all the data points, we called it as an epoch and depending on the application we can select the appropriate epoch number for the training. Depending on that number, it will carry out the training for multiple times because if we trained our model multiple times then only, we can observe good predictions at the end. However, here we must make sure that, our model is generalizing and not just only fitting with the training data points by using a separate dataset called development dataset. Here major issue was overfitting problem and through dropout layers we can handle it. In the training process with the dropout, for each node it will maintain a probability value, call as dropout rate, and depending on that value it will drop the node (Dahl, Sainath & Hinton 2013). With the selection of proper epoch number and dropout layers we can finetune the overall architecture of the model. Selected training parameters are represented in a script file and in DeepSpeech used bash script for that purpose and it explained under the section 4.4.

2.1.2 Recurrent Neural Network

Recurrent Neural Networks (RNN) are mainly used for the Natural Language Processing (NLP) tasks because it has an ability to easily adapt with the patterns which are changed with respect to the time. In NLP, we can split up audio file in to a sequence of sound waves and text can be splits into sequence of individual characters or words. With compared to the traditional neural network, in RNN output in the preceding step are fed into the current step as an input. In ASR, the preceding words affects to the meaning and words that coming in the future. By using RNN we can capture the context by considering the inputs which comes before the current one. For an example, if we consider the text classification, when the model considers a current word then it will encounter the preceding words and make the predictions for future words. RNN build upon on a component called units and there are two types, such as current units and previous units. In NLP it uses Long Short-Term Memory (LSTM) unit with the consideration on how much we will consider in the preceding units for the prediction on current one.

2.2 Transfer Learning

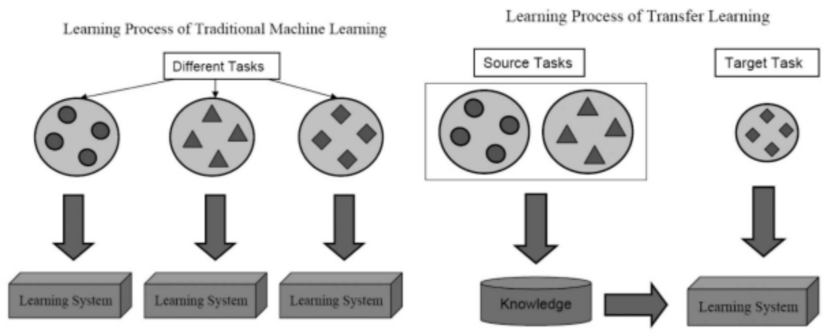


Figure 2.3: Comparison of ML and Transfer Learning

The main idea of transfer learning is to reuse the existing knowledge or experience to improve the learning of new things. In that scenario, the existing knowledge or discovered knowledge known as ”source domain” and the new things that we are interested in learning are known as ”target domain” (Torrey & Shavlik 2010). As shown in figure 2.3, in traditional Machine Learning (ML) each model is developed based on a single domain in an isolation manner, but in transfer learning, we used the knowledge extracted from the source domain to develop the prediction on target domain(Pan & Yang 2009). For an instance, in Speech Recognition, an acoustic model which we were trained for a particular language can be used with the prediction on another language which has poor training data. Through transfer learning, we can improve the overall performances and the model development time compared to the traditional ML and it works well with the low resource language applications.

In transfer learning, source to target knowledge transferring done by considering three aspects and the way of implementing that on DeepSpeech is presented in detail in Chapter 4. Firstly, it will consider ”what to transfer” or which part of the knowledge that we are trying to transfer from source domain to target, because in transfer learning we mainly focus on the common knowledge in both individual domains. Once it discovered the transferring knowledge which are common in both domains, then it considers on the way of transferring it or ”how to transfer.” Finally, it is necessary to identify the time or the situation to apply the transfer learning or ”when to transfer.” If source and target domains are not interrelated with each other, then we are not focusing on applying transfer learning, because it can eventually end up with the negative transfer effect (Pan & Yang 2009).

2.2.1 Transfer Learning for Speech Recognition

Building a speech model for a human language is a challenging task, because it is a dynamic field and new words includes into the vocabulary every day and it is impossible to have a model which last forever. To tackle that, we must have in depth understanding on the variation, imbalance and dynamic on languages and it can be extracted from the analysis on given speech data (Sharma et al. 2017). Insufficient annotated speech data is a major problem in the creation of ASR for low resource language like Sinhala. In order to deal with that issue, it is necessary to have a specialized method with smart algorithms to transcribe spoken data into its textual format. Then it comes transfer learning in the speech recognition task to predict the output results in target domain or data scarce domain by transferring pre-identified knowledge from the data rich domain(Pan & Yang 2009). It can be performed in two different ways.

Heterogeneous transfer learning is the most common approach in speech recognition. Here it will assume the feature space between source and the target domains or tasks are different from each other. For example, if we consider the Part of Speech (POS) tagging application. Here, if we are trying to use POS tagging in German & English languages, their feature spaces are different from each other. Then we pick English as a source and German as a target domain and perform transfer learning by considering their language features. Not only that, occurrences of taggers will vary concerning the domain. For an instance, ”complexity” is common in the technical domain but rarely appears in the restaurant domain. In such a scenario we can consider domain adaption to transfer knowledge from source to target (Asgarian 2020). It will train a source model using several languages and tasks simultaneously. To achieved best results, it is necessary to have a large amount of speech data in each language. Due to that this method cannot use for a low resource language.

In order to tackle that problem, it introduced another transfer learning type called model adaption or fine-tuning. Through this approach, we first train a model by using one or more languages and that trained model called as a source model. Then retrain entire model or the sub part of it by using another language, known as a low resource one. It will take the parameters that we discovered or learned during the training process of source model for the creation or building on the target model.

In this research is based on the transfer learning technique called model adaption or fine-tunning (Torrey & Shavlik 2010)(Pan & Yang 2009). When retraining the parameters, they can consider as static parameters such that they are not changing during the training, or they can be considered as a dynamic parameter such that they are adapted or changing during the training process. In the model adaption or fine-tunning on neural networks, the weights of the nodes in each layer of the model are transferring from the source to target in order to predict the results on the low resource target language will be based on the observation on features in data rich language. Simply we can consider weights reusing as fixed or adjustable. Instead of freezing or adjusting the layers, we can simply remove one or more layers when it moves from source model to target model as it explains in details in Chapter 4.

For an example, in each artificial neural network model last layer is responsible for predict the labels on the given task. When it comes to the identification of English characters then it will predict the letters which belongs to a-z, but when it comes to the Sinhala language (or target model) then it is prediction range will be change according to the number of characters in the Sinhala alphabet and is not compatible with the source language (Torrey & Shavlik 2010). To handle it, the last layer of the source model can be replaced with the new layer which contains extra spaces for the new labels which are common to the Sinhala.

2.3 Sound

Sound is the most important part of any speech related application and the way how we analyze it affects to the quality of output on such applications. Sound is generated through the vibrations which are travelled through air or water media and it can be represent as a transverse wave. There are common features comes with the sound wave, such as amplitude, pitch, and sample rate. The amplitude of the sound wave is mapped with the loudness of the sound wave. To determine the loudness, we refer pitch value of that corresponding sound file and it can be higher or lower. The sample rate of the wave file is known as the total number of samples that we are collecting per seconds and it will be defer from application to application. With respect to the source of audio file we can identify whether its mono or stereo. If we want to focus on a single channel then we can use mono instead of stereo. When we consider on the DeepSpeech we mainly consider on those audio features in the preprocessing and they are mentioned in table 4.1.

Most of the sound has many frequencies with different amplitudes. So, instead of taking time domain to a feature analysis which required larger space compared to other ways, we can go ahead with frequency domain. It uses individual amplitude's analysis and we referred as Discrete Fourier Transformation (DFT) (Håkansson & Hoogendijk 2020). The following sub sections describes the sound analysis method that are commonly used in the DeepSpeech.

2.3.1 Mel-Frequency Cepstral Coefficients (MFCC)

The Mel spectrogram, which considered as the building box of Mel-Frequency Cepstral Coefficients (MFCC), mainly used to transform frequencies into the Mel-scale. It is kind of approximated values which were done through the quantitative experiments and using that we can measure the difference between two pitch values in a different frequency level. Our speech sounds are produced as a result of vocal tract vibration with the support of tongue or teeth and if we can accurately determine its shape, then we can easily predict the character being said. So, in MFCC it gives short term sound shape illustration or the shape of the sound and it commonly used with most of the speech recognition systems. Here, it will capture the most important features or information of sounds and it requires less storage in space compared to others and because of that MFCC used in most of the machine learning tasks.

The transformation of sound waves into MFCC values can be represents in three stages. In the first stage, it will generate the DFT from our input wave file. In order to get accurate results on that we need to have consistent wave file. By defaults, we could not find consistency in our sound files, but if we only focus on a smaller section or part instead of the entire file then we can observe consistency on that. To handle that in DeepSpeech once we input our wave file in a desired language, it will divide into a smaller part called windows and normally here it considers window size as 20 milliseconds. Then we can apply DFT to each window individually and it will generate a list of amplitude for each frequency which mapped with that corresponding sound on that window in a vector format.

The second stage is to apply Mel scale into the values that we received from previous stage. It will map frequency values into Mel scaled values via Log-Amplitude spectrum. Here it will describe the pitch or frequency and loudness or amplitude in a logarithmic model. Because, otherwise amplitude change in lower frequency will have higher impact than the higher frequency. Using the technique called compression is applied for combine the Mel scale and log values. Here we will use the triangular filters around each sample point to avoid the misleading results in the prediction. Here it considered the summation of amplitudes in that specific area surrounded by the sample point. In the final stage, it applies Discrete Cosine Transformation (DCT) to our filtered values to generate MFCC of our specified windows. In DCT is more like the DFT, but here it replaces sinus function with cosine function.

Chapter 3

Literature Review

In the Automatic Speech Recognition, which gets acoustic wave form and trying to translate into textual format for the given language. There are several specialties of speech which make recognition task more difficult such as style, vocabulary, noise, and accent. Style of speech can be spontaneous or continuous. Earlier handling continuous speech is a challenging because of its continuous flow of the words. Handling the noise interference also one of the major challenges, because if we are speaking with the interference of background sounds or having an environment with lots of echoes, then we need to filter them out before process with it. Accent or the rate of speech of the speaker also affected to the accuracy of ASR. To tackle all of these issues we required large speech dataset and building a speech recognition for the low resource language is a challenging task.

3.1 History of ASR

The history of ASR begins from the 1920s onward with the invention of the prototype called "Rex" and it looks like a tiny dog. It is a frequency detector for a single word and not doing much in recognition. In the 1930s onward with the invention of Vocoder, which uses in the telecommunication field to encrypt voice messages before the transmission to ensure the security aspect. Later they introduce a modern version of it by applying modern technologies to enhance their performances (Nadungodage & Weerasinghe 2011).

In the 1952, Davis, Biddulph, and Balashek build a speaker-dependent system for recognizing the digit through a filter-based approach. After that, Oslan and Belar's at RCA laboratories introduced another speaker-dependent discrete speech system, which is capable only to handle 10 syllabi. Also, the team of researchers at MIT Lincoln Lab introduce another speech system to manage 10 speaker-independent vowels via measuring the spectral resonances (Nadungodage & Weerasinghe 2011). At that period almost all the speech recognition systems are capable to handle or recognize speeches which has single vowel or word.

There was another system called "ShoeBox" at IBM in 1962, which capable enough to recognize strings of digits together with the capabilities of doing simple arithmetic operations. However, it is limited up to 10 digits and 6 arithmetic operations which are in the isolated style. In the 1970s, the Carnegie Mellon University's developed an ASR system together with the support of AI and used to recognized 1011 words from a connected speech vocabulary. It was named as "Harphy" (Arora & Singh 2012). So, it is an improvement of speech recognition from isolated speech recognition to continuous speech recognition. After that, with the enhancement of the vocabulary sizes it introduced another system called the Large Vocabulary Continuous Speech Recognition (LVCSR), which initiates in the 1970s but it was hard at that time due to the existing technologies because it is targeting on speaker independent speech data.

Up to now all the researches were conducted template-based approaches to match the speech signals. Through that template-based approach, it took speech signal and get the frequency representation on that and map it with the best template and it will change from person to person, because for each has a different speech rate. Then in 1980s a paradigm shift was introduced and focus on statistical-based approach instead of template-based ones. At that time Hidden Markov Model (HMM) for continuous speech recognition arise and it used mathematical algorithm of the speech rather than just looking at the properties of the speech (Arora & Singh 2012). The SPHINX system which was invented by Kai-Fu Lee at Carnegie Meron University, uses HMM to map speech state with respect to the time and it uses Gaussian Markov Model (GMM) to identify observation probabilities in each HMM states (Lee 1988). During that period, HMM-GMM had been famous in the speaker independent continuous speech recognition on large vocabulary and it still used in current researchers works as well. Later it introduces the machine learning-based approaches starting from the decision tree mechanism and then introduces a statistical-based approach, which tells about the probabilistic decisions among words or sentences. Later it focused on a neural-based approach to overcome the poor accuracy issues in the statistical model (Nadungodage & Weerasinghe 2011).

Recently, deep leaning gives several benefits in many fields such as question and answering and speech assistant devices. With the new development of the deep learning, it introduced deep neural network (DNN). It replaces the previously calculated observation probabilities with the posterior probabilities of the HMM states and that model we called as HMM-DNN (Lee 1988). In 2011, Yu Dong and his team from Microsoft Research Institute introduced an HMM together with context based DNN and it called as context dependent (CD) DNN-HMM and it gave remarkable results of LVCSR compared to the HMM-GMM (Dahl, Yu, Deng & Acero 2011). Therefore, now almost all the real-world applications such as Cortana, Siri, Google Assistant and Alexa is built upon DNN.

Anyway, all these neural models have a common problem of short-term memory and LSTM is a specialized version of RNN which solved that problem. However, all those HMM and DNN models used stand-alone phone information dataset on their prediction and LSTM used temporal word and sentence information. When comparing the performances in the statistical approach and the DNN approach, it specifies that the neural approach achieves much better accuracy than the statistical one. Therefore, currently researchers' focus on building speech related systems using DNN approaches.

Whatever the HMM based model that we used, whether it is HMM-GMM or HMM-DNN, there are practical issues arise with LVCSR task. HMM based models used different training methods and dataset to train the different models and it ultimately increased the complexity of the training process. In order to make model construction simple, it uses conditional independence assumptions within HMM and other models, but it does not happen in real in LVCSR task. To tackle those problems, it is necessary to have a straight forward mapping from audio signal to sequence of words.

To achieve that the researchers were introduced new deep learning technology called end-to-end (e2e) speech recognition which includes 3 parts, such as firstly encoder; which align audio sequence to feature sequence, secondly aligner; which identify the alignment between feature sequence and the specific language, and finally decoder; which responsible for producing final word sequence. For Sinhala e2e speech recognition system that builds upon Lattice-Free Maximum Mutual Information (LF-MMI) on the Kaldi toolkit achieved 28.55% WER (Gamage et al. 2020). Still, it is much better than the traditional approach, by applying fine-tuning mechanism we can improve its performance by reducing the WER it is under the experimental stage with the recent research conducts by the Language Technology Research Lab (LTRL).

3.1.1 Speech Recognition for Low Resource Languages

The language that does not have enough linguistic resources to building a speech recognition system known as "low resource language." Those linguistic resources can be considered as lexicon or pronunciation dictionaries, paired speech corpus, which contains speech file together with its corresponding labels and unpaired speech corpus. There are approximately 7000 spoken languages around the world but only 100 languages support with the accurate ASR systems. So, majority of languages does not have proper ASR system, because they do not have sufficient speech resources to build the speech recognizer. When it comes to the Sinhala, it only contains 40 hours of speech data which is reordered using 2000 unique sentences from 123 speakers including male and female groups and it was created by LTRL group. Therefore, Sinhala falls into the low resource language group compared to the data rich languages like English, French and German. In 2011, a speaker dependent continuous speech recognizer was developed for the low resource non-Latin language called Sinhala and it has obtained 96.14% of accuracy in word prediction with the model which was trained by using 983 separate Sinhala speech files and tested with 106 wave files (Nadungodage & Weerasinghe 2011). The system consists two main sub models as HMM based acoustic model and bi-gram language model. However, the system gives inaccurate prediction for other speakers, because it was trained with the single female voice.

Collecting the speech dataset is the initial task in the statistical based speech recognizer, because system overall performance or the accuracy depend on the richness of speech corpus. So, in 2013, research has been done with the intention of creating speech corpus for the low resource language, Sinhala and it covered 78,667 sentences which lasts for 65 hours of recordings covering 12 different types including sentences (phonetically rich), keywords, currency, boolean data, digits, dates, times, music types, proper names, songs, numbers, and questions. Those data were recorded from the mobile phone calls and due to that it has some inferences with the noises(Nadungodage, Welgama & Weerasinghe 2013).

Sinhala can be considered as a language which has been spoken by small community, mainly in developing countries, is lack of obtaining or recording speech data due to its poor linguistic knowledge and the complexity on its language characteristics. To tackle that, in 2013, a group of researches were conducted a study to analysis the effective usage of training data for speech recognition by using the active learning with the intention on extracting the most informative speech data. In this study, they compared the results in between active learning data selection and random data selection using 2897 utterances. If we used the entire speech corpus then it has given 95.17% of accuracy and we can get the same accuracy by reducing roughly 42% of data in the active learning approach was their conclusion on the study (Nadungodage, Weerasinghe & Niranjan 2013).

There are several problems arise with the low resource ASR system. Interacting with unpaired speech corpus is tough task to achieve, because data labelling is a time-consuming task. This type of low resource language does not have enough researching background or analysis due to the insufficient linguistic specialists and it will ultimately result with building poor pronunciation dictionaries. In order to cooperate with these kinds of low resource languages issues, researchers have proposed new mechanism called transfer learning (Yi, Tao, Wen & Bai 2018).

3.2 Transfer Learning for Low Resource Languages

The transfer learning concept arise with the way of human obtained new knowledge in day-to-day life. Human has an ability to adapt with the new task or activity using the knowledge on some similar tasks which they familiar (Yi et al. 2018). In transfer learning we interact with two datasets, first dataset which known as large dataset or the data rich dataset and using that we learned the system for some tasks. Then by using the second dataset which is a real dataset that we are interested is quite small compared to the previous dataset but still we were focused on the same task. Here, we transfer knowledge from the first system to the second one for getting the better results. Likewise, in speech recognition we can use the knowledge which we gathered from the data rich model to predict the output prediction on the low resource model using transfer learning.

Transfer learning implementation methods can be identified as two types, such as transferring bottleneck features and transferring model parameters. In transferring bottleneck features we take the pre-trained model using well known large dataset and remove the output layer of that and replace with our own one which align with the target low resource speech data. In bottleneck feature transfer learning, it will remove original output layer weights while freezing all other preceding layers. When we start the training process again with the target speech data then only it will readjust the weights in output layer. Another type is transferring model parameters. Here it will share the hidden layer parameters, which we called as Bi-directional Long Short-Term Memory (BLSTM) parameters from the source model to target model (Yi et al. 2018).

In 2015, a group of researches proposed an approach to cross-lingual transfer using two ways, such as bottleneck features and deep neural network multilingual training done from a source model, which trained using Cantonese, Pashto, Tagalog, and Turkish, to the target model on Vietnamese and Tamil. It concluded that bottleneck feature systems performed well on the very limited data together with noisy environment (Xu, Van Hai Do, Xiao & Chng 2015). In 2017, a study has been conducted for testing the applicability of transfer learning on low resource language using model adaptation. It used Wav2Letter as a baseline model because of its short training time and fewer hardware requirements. They have been chosen rectified linear units as the model activation function, which worked well on most of the acoustic models, together with Adam optimizer. The source model trained for English using LibriSpeech corpus covering 1000 hours of read speech data and target model used German speech data gathered from Bavarian Achieve for Speech Signal (BAS) with the 383 hours of coverage. Initially it trained with English data and frozen layers up to kth level from the bottom and adapt rest with the German language until it reached and give the transcription for the input at output layer. They obtained the best performance of their transfer learning model with a WER of 42.49% and LER of 15.05% with the support of 4-grams language model. Through this study they concluded that while using transfer learning we can reduce the training time and adapting with the lower GPU memory requirements than other training mechanism for the low resource languages (Kunze, Kirsch, Kurenkov, Krug, Johannsmeier & Stober 2017).

In 2018, there was a competition called "Interspeech 2018" with the purpose of inventing innovative ideas on low resource speech recognition for 3 languages, Tamil, Telugu and Gujarati using 50 hours of data for each. Final evaluation is performed based on the WER on blind test. Initially they provided 3 baseline models as GMM-HMM concerning MFCC features, Karel's DNN concerning Feature-space Maximum Likelihood Linear Regression (fMLLR) and TDNN which includes the lowest WER. Among all other teams, Jilebi system performed well on all languages using multilingual TDNN model with transfer learning together with n-gram language model. Their proposed networked has been taken 40 high resolution MFCC features with 100 dimensional as an input for the speaker adaptation purpose. Then used 3-gram Kneser-Ney language model which was trained using SRILM for the output prediction and transfer weights of the hidden layers in source model to the target model for the above specified languages (Pulugundla, Baskar, Kesiraju, Egorova, Karafiát, Burget & Cernockỳ 2018). In addition to that, ISI-Billa system, which is a EESEN based e2e multilingual LSTM system trained with CTC, built as monolingual or multilingual speech recognizer performed better than the baseline models for each language (Srivastava, Sitaram, Mehta, Mohan, Matani, Satpal, Bali, Srikanth & Nayak 2018).

In 2019, research has been conducted to improve the performance on speech-to-text translation on low resource language by using model parameter transferring. Here, it created an ASR model using high resource speech data on English language covering 300 hours and then fine tune the parameters of that model to adapt with the Spanish to English speech to text translation which contained 20 hours of data coverage. Through this proposed approached they obtained BLEU score improvement from 10.8 to 20.2. They concluded that it is not necessary to pre-trained the ASR model using source or target language, instead of that it can do with different language and fine tune the model parameters. Still, it shows improvement on the output results and they experiment that with the French ASR to Spanish- English ST (Bansal, Kamper, Livescu, Lopez & Goldwater 2018).

End-to-end (e2e) speech recognition architecture gives us an ability to work on a speech data which includes different accent, multilingual and noisy environment perfectly than traditional statistic approach. In 2016, a group of researchers have been conducted a study on developing e2e speech recognition system for in English, which consists 11,940 hours coverage, and Mandarin language, which consists 9400 hours coverage. The proposed approach is trained end to end using RNN together with CTC loss function. The networked has been taken 20ms wave clips as input to predict the alphabet on each language. Through the proposed DeepSpeech model it showed better results with the lower WER for two read speech corpora generated from Wall Street Journal (WSJ) and LibriSpeech corpus from audio books (Amodei, Ananthanarayanan, Anubhai, Bai, Battenberg, Case, Casper, Catanzaro, Cheng, Chen et al. 2016). Using the finding on this research, another study has been conducted for creating transfer learning based e2e speech recognition system for low resource language called Tujia from Chinese language which was studied in 2019. The model was used CNN and bi-directional long short-term memory (BiLSTM) and it trained with 2 hours and 54 minutes spoken data coverage in Tujia by showing remarkable results on the recognition with 2.11% of WER reduction compared to the baseline model created on pure Tujia language (Yu, Chen, Li, Kang, Xu & Liu 2019).

In 2019, a group of researchers were conducted study on meta learning approach for low resource speech recognition using Model-Agnostic Meta Learning algorithm (MAML) and the model called as MetaASR. Then the results of

MetaASR model compared with the MultiASR model, which known as multilingual transfer learning or initializing the model with source language and then fine-tune it to adjust with the target language. For the conversational telephone speech dataset (CTS) from the IARPA BABEL project, showed improvement of performances in both MultiASR and MetaASR without considering pretrained parameters and on the other hand it showed better results for MetaASR. The study that has been presented at the InterSpeech 2019 and it showed applicability on transfer learning from knowledge extracted from one task to map them with another task, while taking ASR as source and semantic slot filling (SF) as target task. For both it used the same model architecture except text data inputs and output representations. For ASR training it used 207 hours of TED talks while considering English alphabet and additionally add blank to the output layer (Tomashenko, Caubrière & Estève 2019).

In 2020, a study has been conducted to evaluate the domain specific automatic speech recognition on Swedish language by using the transfer learning method from English to Swedish via DeepSpeech. In this study they mainly considered on three datasets such as NST acoustic dataset for Swedish covering 500 hours concerning 1000 speakers with 10 dialects, Lunchekot Swedish news corpus covering 5 hours and Sport dataset. Then they build the baseline model on Swedish using NST dataset and achieved WER:19.01% & CER: 7.05%. Then they applied data augmentation and 5-grams LM for the target model and improved it performances by 5.21% in WER and 2.27% in CER for the domain specific identification (Håkansson & Hoogendijk 2020).

In 2020, a study has been conducted to evaluate the performance on low resource language speech recognition such as Swiss German using an e2e DeepSpeech model. They built a baseline model using 70 hours of data gathered from shared task and its majority covered with Bernese dialect. That model trained with 3-grams LM and unfortunately it gave poor results achieving the 71.5% WER. Then they apply hyperparameter optimization, transfer learning techniques and data augmentation to increase the data coverage. In transfer learning they used English to German to Swiss knowledge transfer and then identify the optimized hyperparameter for the desired language. Finally, apply data augmentation considering frequency masking, time masking, speed scaling, and pitch scaling. When they applied data augmentation for the baseline model WER increased by 2.8% but with the optimization and transfer learning it gave much improved results compared to baseline model. In conclusion, with the parameter optimization and transfer learning without using data augmentation they achieved 56.6% of WER which less than 14.9% of baseline model (Agarwal & Zesch 2020).

A study has been conducted to identify the domain-specified speech command for Sinhala or Tamil languages by using the knowledge gathered from a high resource language called English via CNN model. They selected pre-trained DeepSpeech model on American English as a source model, which contains 5 bidirectional recurrent layers, due to its flexibility on training together with computational support. Then the target model trained with the audio clips gathered from banking domain in both Sinhala and Tamil languages and it takes MFCC coefficient features as an input to the network to predict the corresponding transcript by considering the extracted intermediate probability features on source model. It gave 93.16% and 76.30% accuracy for Sinhala and Tamil respectively and they concluded that adaptability on noisy environment and adapting with small feature space is the result for that performance (Karunanayake et al. 2019).

Until 2019, most of the speech recognition systems built upon supervised learning, but from that period onward researchers were focus on unsupervised learning and transfer learning due to the issue arise with the low resource languages. So, to adapt with the speech recognition on low resource languages, like Sinhala, we can apply the feature level transfer learning by keeping the e2e English ASR model as a source or the baseline model. It will open a door to the new era of speech recognition for Sinhala language and giving us change to do more analysis on other Hindu Aryan languages.

Chapter 4

Methodology

In this chapter the methodology that has been followed during the research is described and each main section represent how we developed this research design out of that methodology descriptive. Initially the appropriate Sinhala dataset is collected and then preprocessed as it described in Section 4.1. In section 4.2 explained the implementation of the development of language model based on KenLM toolkit for Sinhala, to identify and incorporate the features which are specific to the Sinhala language. Using the preprocessed speech data together with the help of language model, Mozilla's DeepSpeech is used to fine tune the existing English pre-trained model for building a Sinhala ASR via transfer learning. For the evaluation purpose it is necessary to build a baseline Sinhala ASR model and that are described in Section 4.3 including its experiments. Later application of transfer learning for the prediction on Sinhala data with or without considering data augmentations are described in Section 4.4.

4.1 Pre-processing the Dataset

To train our own ASR model, it is necessary to have speech data and its corresponding transcripts in a desired language. Data collection or data preparation is the most crucial task in the ASR pipeline and overall accuracy is depended on the richness of our collected dataset. Therefore, it is necessary to have a proper dataset which covered almost all the desired features on the target language. There are already available pre-collected speech datasets at Language Technology Research Lab (LTRL) from UCSC, which includes 40 hours of Speech data on Sinhala languages. However, that data set was collected through two different softwares as Praat, which is known as a free software package that targeting on speech analysis at the phonetic level, and the other one is Redstart, which comes as a default recording software in the MaryTTS framework.

When considering those two datasets, 45 persons are involved with Praat recording and among them 31 of them are female and rest 14 of them are male. In addition to that, 78 persons contribute to Redstart recording and among them, 55 are female and 23 of them are male. Therefore, our dataset contains both male and female voices. When the recording phase, each speaker gets 200 sentences and it contains news readings, number readings, time, and date readings. Among that first 50 sentences are common to all the speakers and it is used to handle variations in phones in Sinhala. Next 150 sentences are unique to the speaker and rarely their can be reaping ones as well. However, these datasets are noise-free datasets, and according to the researchers' observation, they mentioned it is better to have noisy data as the training dataset in the DNN model. Due to that fact, it includes another 80 utterances on Sinhala which were recorded in a noisy environment via iPhone and it used for the testing purpose on our model (Gamage et al. 2020).

In order to achieve better accuracy with any machine learning model it is necessary to follow preprocessing on the collected data in advanced. Spoken data in real world is not pure, because it can be noisy, inconsistent, or incomplete. Therefore, before it starts the training process it is necessary to filter out those impure or dirty data from our dataset.

All the speech files that we have collected for the model training must be converted into the suitable format which can be used with the Mozilla's DeepSpeech (*Welcome to Deepspeech's documentation* n.d.). Those required speech file properties are listed in table 4.1

Sample Rate	16000Hz
Resolution	16bits
Channels	1 (mono)
Encoding	Linear PCM (LPCM)

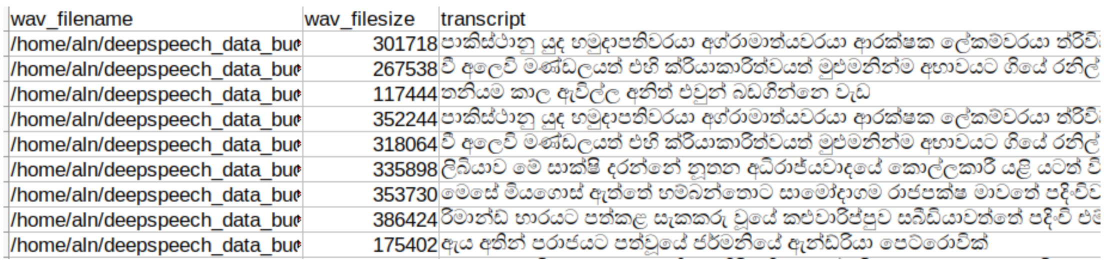
Table 4.1: Speech file properties in Mozilla's DeepSpeech

4.1.1 Filtering the undefined data

Then it is necessary to normalize the transcript according to the training requirements of DeepSpeech. First, we need to create a single text file which contains all the transcriptions on collected spoken data. Corresponding transcriptions are only allowing to have a special set of characters in our transcripts and it called as an alphabet. Initially it taken English alphabet as a default, so it is necessary to replace it with Sinhala alphabet because our focus is on detecting Sinhala speech utterances into textual format.

To the accurate Sinhala utterance identification purpose, it included zero-width non joiner, zero-width joiner and zero width space special characters in it as states in Gamage et al. study (Gamage et al. 2020). If any transcript contains characters which are not mentioned or undefined in the alphabet, then we must convert it into the characters which defined in it. Otherwise, during the training if DeepSpeech model detected some undefined character such as "/" or "." or "," or "@" or "&" etc., then our entire model will be crashed. Therefore, before preparing the csv files for the model training, we must remove such characters in the pre processing. Since alphabet is made up with all lower-case characters and before converting undefined characters it is necessary to convert all characters into lower case to make sure that it does not contain any capitalized letters within our transcripts. Since in DeepSpeech it will not identify the letters A and a as equal letters (Håkansson & Hoogendijk 2020).

Then we can transfer our training, testing, and validating dataset into the form of comma separated values (CSV), because in Mozilla's DeepSpeech it required the data represent in that format. Each file starts with the header by specifying the 3 parameters that we collected such as wav_filename, wav_filesize and transcript followed with set of data values correspond to each speech file as it shows in figure 4.1. A single record in this CSV file is representing desired parameters in one wav file. Here if we choose one wav file then path to the that specific file is represented in wav_filename and the size of that wav file in bytes mentioned as a 2nd parameter. Normalized transcript is represented under 3rd column in csv file.



wav_filename	wav_filesize	transcript
/home/aln/deepspeech_data_bu	301718	පාකිස්ථානු යුද හමුදාපතිවරයා අග්‍රාමාත්‍යවරයා ආරක්ෂක ලේකම්වරයා ත්‍රිවි
/home/aln/deepspeech_data_bu	267538	වී අලෙවි මණ්ඩලයත් එහි ක්‍රියාකාරිත්වයත් මුළුමනින්ම අභාවයට ගියේ රනිල්
/home/aln/deepspeech_data_bu	117444	තනියම කාල ඇවිල්ල අනිත් එවුන් බඩගින්නෙ වැඩ
/home/aln/deepspeech_data_bu	352244	පාකිස්ථානු යුද හමුදාපතිවරයා අග්‍රාමාත්‍යවරයා ආරක්ෂක ලේකම්වරයා ත්‍රිවි
/home/aln/deepspeech_data_bu	318064	වී අලෙවි මණ්ඩලයත් එහි ක්‍රියාකාරිත්වයත් මුළුමනින්ම අභාවයට ගියේ රනිල්
/home/aln/deepspeech_data_bu	335898	ලිබියාව මෙ සාක්ෂි දරන්නේ නූතන අධිරාජ්‍යවාදයේ කොල්ලකාරී යළි යටත් වි
/home/aln/deepspeech_data_bu	353730	මෙසේ මියගොස් ඇත්තේ හම්බන්තොට සාමෝදාගම රාජපක්ෂ මාවතේ පදිංචිව
/home/aln/deepspeech_data_bu	386424	රිමාන්ඩ් භාරයට පත්කළ සැකකරු වූයේ කළුවාරිප්පුව සබීඩියාවත්තේ පදිංචි එම
/home/aln/deepspeech_data_bu	175402	ඇය අතින් පරාජයට පත්වූයේ ජර්මනියේ ඇන්ඩ්‍රියා පෙට්රොවික්

Figure 4.1: Image of preprocessed train CSV data file

4.1.2 Splitting the speech dataset

In machine learning models, dataset need to be split into three parts as training set, testing set and validation set. In the training set it contains speech files including 67 females and 27 males covering approximately 25 hours of speech data. In the validation set it contains speech files including 8 females and 3 male speakers which uses to fine tune the model which we created. Finally, in the testing dataset it uses speech data collected from 4 females and 4 males in a noisy environment and it was gathered 80 sentences from iPhone (Gamage et al. 2020).

4.1.3 Activating Mozilla's DeepSpeech enviornment

However, building an own model from it scratch is bit expensive task, so instead of that in this research we used existing model, which we can easily find a clone of that in Mozilla's DeepSpeech repository on GitHub. Once we download and installed it, next it is necessary to build a virtual environment to conduct all our executions. It will create a directory which contains python3 binaries which helps to do speech to text conversion on an audio file including all other tools that needed to train a DeepSpeech model. As it stated in the document, we can use following command to create the virtual environment.

```
$ python3 -m venv ./deepspeech-train-venv/
```

Once it executed this command successfully, that created environment is ready to activate. Every time when we are using the Mozilla's DeepSpeech, it is necessary to activate the environment in advance and it can be easily done with the following command,

```
$ source ./deepspeech-train-venv/bin/activate
```

Once it successfully activated, we can see the deepspeech-train-venv at the beginning of our prompt line within the bracketsand it means that we are already within that created environment or we are active in the environment.

4.2 Building the Language Model

When developing ASR system by using Mozilla's DeepSpeech, a language model, which is optional in the architecture and normally it used to improve the accuracy of the result or the prediction on target language. In DeepSpeech, result is identified based on the sound of the input file and still there can be misclassifications happens due to the in adequateness with language features. It can be handled via the proper analysis on the language specific features and we comprised all together and build something called language model or external scorer. Still, there are no any freely available language model for the Sinhala which is compatible with the Mozilla's DeepSpeech covering required variations in Sinhala. So, we must create it from scratch by using our Sinhala speech dataset.

Before it starts the implementation on language model or external scorer using KenLM toolkit, we must apply some normalization to our text corpus. The text corpus that we have used for the building process on the language model is a combination of 3 different corpora such as UCSC Novel Corpora, which has 90000 unique sentences coverage, Chatbot corpora, which consists of 388 unique sentences coverage and 20000 unique sentence coverage collected from active learning method(Gamage et al. 2020). As it mentions in the documentation, we cannot use those corpora in its original format, and it required some pre-processing. Here we must translate all numbers and abbreviations into the Sinhala text and then copy all sentences into one file. Then we can feed it into our KenLM toolkit (Håkansson & Hoogendijk 2020).

So, external scorer building process can be divided into two parts, such as building KenLM language model and trie data structure which includes all words in the vocabulary based on our input file. The KenLM toolkit has inbuilt separate scripts and tools to build them. Among them, lmplz tool is used for creating the n-grams based language model for the provided normalized input text corpora. Second tool is known as build_binary tool which can convert the previously generated n-grams language model in to the binary representation. Through the lmplz tool it allows us to mention n, when we were developing n-grams language model and for Sinhala we used 4 grams model from the selected corpora as it states in the previous study (Gamage et al. 2020). Once we successfully build the language model for Sinhala, then we need to convert it into the binary trie-data structure with the support of build_binary tool. For that trie-data structure creation we must feed target language alphabet as an input to the execution. By using generate_lm.py script, it will generate a binary file called lm.binary and vocab-500000.txt and then it fed into the process of scorer package creation with the execution of generate_scorer_package. As it shown in the following code, first it will create 4-grams language model and it represents with the arpa_order value as 4. Then once it successfully executed the script file it will automatically generate the binary representation of the scorer file and it stored as lm.binary.

```
generate\_lm.py —input\_txt
/home/aln/corpus\_data\_buddhi/corpus\_20000+90000+
chatbot.txt  —output\_dir  .
—top\_k 500000 —kenlm\_bins
/home/aln/MCS/DeepSpeech/kenlm/build/bin/
—arpa\_order 4 —max\_arpa\_memory ”85\%”
```

```
--arpa\_prune "0|0|1" --binary\_a\_bits 255
--binary\_q\_bits 8 --binary\_type trie
```

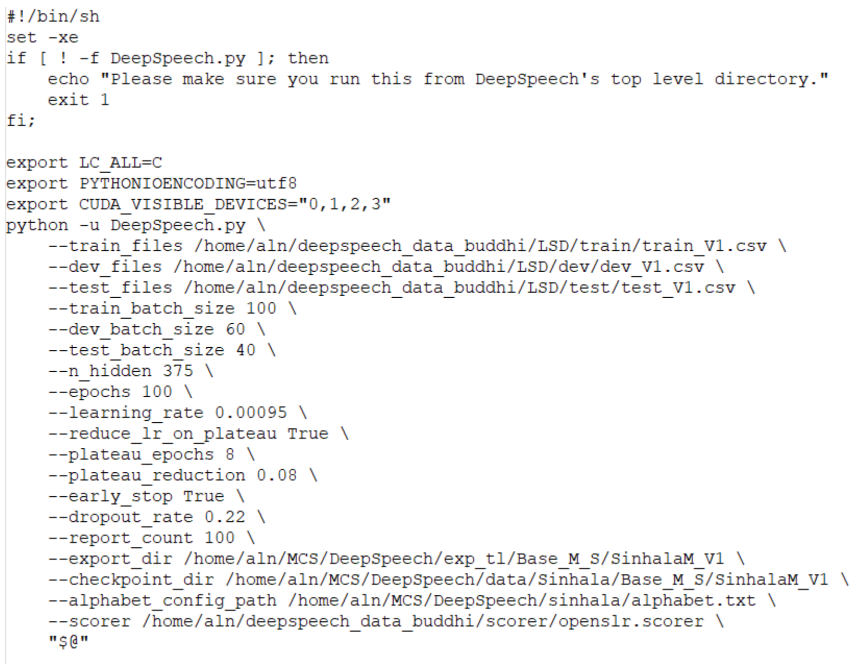
Later we need to create an external scorer as a package for future applications using the previously generated two output files, lm.binary and vocab-500000.txt with the default alpha and beta values as it shown in the code below (*Welcome to Deepspeech's documentation* n.d.).

```
./generate\_scorer\_package --alphabet /home/aln/MCS
/DeepSpeech/sinhala/alphabet.txt --lm /home/aln/MCS/
DeepSpeech/data/lm/lm.binary --vocab /home/aln/MCS/
DeepSpeech/data/lm/vocab-500000.txt --package /home/
aln/MCS/DeepSpeech/data/lm/openslr.scorer
--default\_alpha 0.931289039105002
--default\_beta 1.1834137581510284
```

4.3 Implementing baseline Sinhala ASR model

The Sinhala ASR model was implemented by using Mozilla's DeepSpeech and we used version 0.9.3 for our study. Here it used the preprocessed Sinhala spoken data files as it described in section 4.1 for the training purpose of the RNN model. In this research, we are mainly focusing on two approaches. Among them the first approach is to develop Sinhala ASR system from it scratch by considering the dataset explained in Section 4.1 and with the support of the language model which explained in the section 4.2. Next approach is to fine-tune the pre-trained English model, which was already pre-trained with the LibriSpeech speech data corpus covering 5500 hours, with our existing Sinhala dataset by using the transfer learning mechanism and in Section 4.4 it describes in details. Also, the model which were trained from the scratch used as a baseline model to evaluate the performances on the model trained by considering the transfer learning.

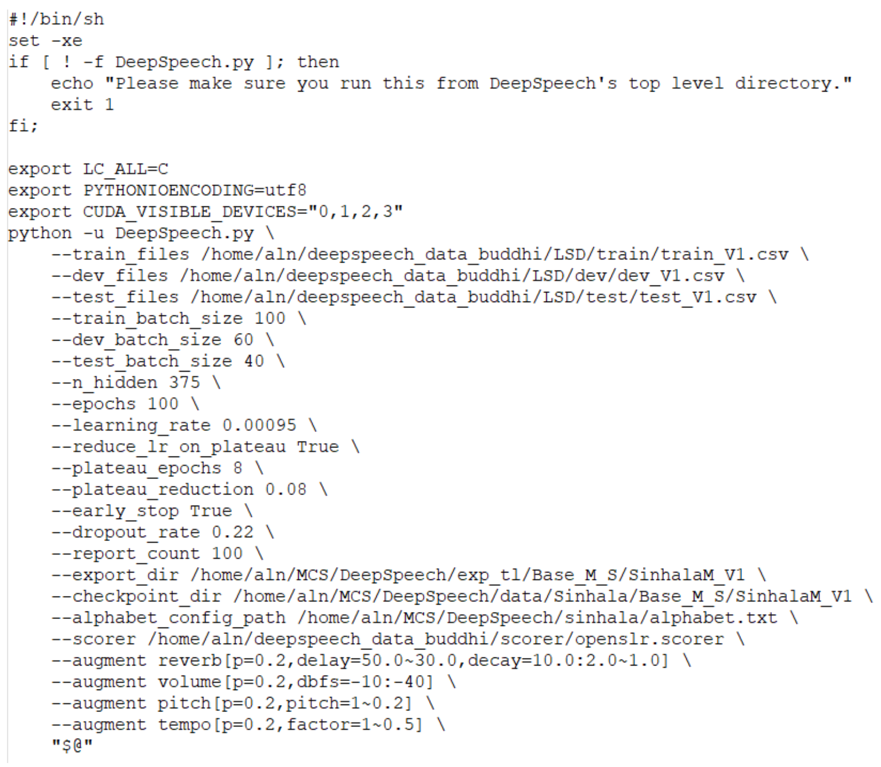
As a starting point of our first approach, it is necessary to find out the most appropriate hyperparameters for Sinhala model training with respect to the collected dataset. Based on the finding on the previous study (Gamage et al. 2020), we have selected the hyperparameter values for batch size, learning rate and dropout rate as it mentions in the study. Since learning rate is quite compared to the default value, predictions may get bit long time to align with the actual target, because in machine learning model learning rate is the criteria that we used to adjust on predictions. With respect to the previously selected parameters, we have noticed that lost is not decreasing with respect to the training iterations and it concludes that our trained model may have plateaued. To handle that issue, we have added learning rate adjustments to the model that we created and the selected parameters are represented as it shown in figure 4.2.



```
#!/bin/sh
set -xe
if [ ! -f DeepSpeech.py ]; then
    echo "Please make sure you run this from DeepSpeech's top level directory."
    exit 1
fi;

export LC_ALL=C
export PYTHONIOENCODING=utf8
export CUDA_VISIBLE_DEVICES="0,1,2,3"
python -u DeepSpeech.py \
    --train_files /home/aln/deepspeech_data_buddhi/LSD/train/train_V1.csv \
    --dev_files /home/aln/deepspeech_data_buddhi/LSD/dev/dev_V1.csv \
    --test_files /home/aln/deepspeech_data_buddhi/LSD/test/test_V1.csv \
    --train_batch_size 100 \
    --dev_batch_size 60 \
    --test_batch_size 40 \
    --n_hidden 375 \
    --epochs 100 \
    --learning_rate 0.00095 \
    --reduce_lr_on_plateau True \
    --plateau_epochs 8 \
    --plateau_reduction 0.08 \
    --early_stop True \
    --dropout_rate 0.22 \
    --report_count 100 \
    --export_dir /home/aln/MCS/DeepSpeech/exp_tl/Base_M_S/SinhalaM_V1 \
    --checkpoint_dir /home/aln/MCS/DeepSpeech/data/Sinhala/Base_M_S/SinhalaM_V1 \
    --alphabet_config_path /home/aln/MCS/DeepSpeech/sinhala/alphabet.txt \
    --scorer /home/aln/deepspeech_data_buddhi/scorer/openslr.scorer \
    "$@"
```

Figure 4.2: Training script used for the baseline Sinhala ASR model without augmentation

However, Sinhala is known as a low resource language and our dataset includes approximately 40 hours speech data coverage. It is a common fact that deep learning approach result overall quality highly depends on the size of the corpus coverage. In machine learning model data augmentation helps to build generalized model. Here it will transform our existing data without creating additional data and it was done according to the order given in the script. This is helpful if we have enough data and want to generalize our model bit according to the given consideration, which was given in the augmentation flags under 4 different domains as sample, signal, spectrum, and feature (Agarwal & Zesch 2020). We applied data augmentation for our previously defined training script by considering the given order as it shown in figure 4.3. It uses as a baseline model for the transfer learning on English to Sinhala ASR with augmentation. To identify the stopping point of the model training, we enable the early stopping feature. As the result of that at the end of each epoch, it will calculate the loss value and if it is remained as it is without changing for few epochs, then the model training was stopped.



```
#!/bin/sh
set -xe
if [ ! -f DeepSpeech.py ]; then
    echo "Please make sure you run this from DeepSpeech's top level directory."
    exit 1
fi;

export LC_ALL=C
export PYTHONIOENCODING=utf8
export CUDA_VISIBLE_DEVICES="0,1,2,3"
python -u DeepSpeech.py \
    --train_files /home/aln/deepspeech_data_buddhi/LSD/train/train_V1.csv \
    --dev_files /home/aln/deepspeech_data_buddhi/LSD/dev/dev_V1.csv \
    --test_files /home/aln/deepspeech_data_buddhi/LSD/test/test_V1.csv \
    --train_batch_size 100 \
    --dev_batch_size 60 \
    --test_batch_size 40 \
    --n_hidden 375 \
    --epochs 100 \
    --learning_rate 0.00095 \
    --reduce_lr_on_plateau True \
    --plateau_epochs 8 \
    --plateau_reduction 0.08 \
    --early_stop True \
    --dropout_rate 0.22 \
    --report_count 100 \
    --export_dir /home/aln/MCS/DeepSpeech/exp_tl/Base_M_S/SinhalaM_V1 \
    --checkpoint_dir /home/aln/MCS/DeepSpeech/data/Sinhala/Base_M_S/SinhalaM_V1 \
    --alphabet_config_path /home/aln/MCS/DeepSpeech/sinhala/alphabet.txt \
    --scorer /home/aln/deepspeech_data_buddhi/scorer/openslr.scorer \
    --augment reverb[p=0.2,delay=50.0~30.0,decay=10.0:2.0~1.0] \
    --augment volume[p=0.2,dbfs=-10:-40] \
    --augment pitch[p=0.2,pitch=1~0.2] \
    --augment tempo[p=0.2,factor=1~0.5] \
    "$@"
```

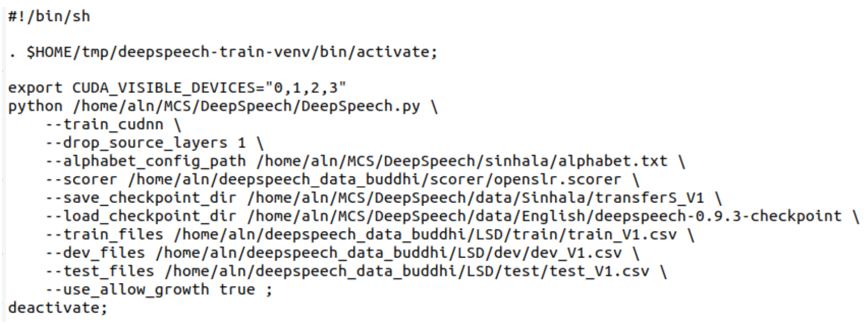
Figure 4.3: Training script used for the baseline Sinhala ASR model with augmentation

4.4 Implementing Sinhala ASR model using transfer learning

As it mentions in the above, our second approach was to implement Sinhala ASR system using transfer learning referring to the pre-defined English Mozilla's DeepSpeech model. From 2019 onwards transfer learning technique widely used to implement speech recognition system for low resource languages. It will do the prediction on low resource speech utterance depending on the extracted knowledge on high resource or data rich model. There are numerous studies has been conducted for several low resource language including Sinhala, but due to the data scarcity they only considered a small domain in their analysis. Due to that reason, we were interested with applying transfer learning for wide range analysis on Sinhala data. In transfer learning our source model has to be data rich model and it requires an efficient hardware and larger memory capacity in the training phase. Instead of training an English DeepSpeech model from it scratch, we used pre-trained English model which were trained on Librispeech data and it can be easily downloaded in the DeepSpeech resources.

The pre-defined English model was trained by using the English alphabet and it is capable enough to cover all the characters in English transcripts. When it comes to the Sinhala, we cannot recognize any Sinhala letter using default English alphabet. To tackle that it is necessary to feed our Sinhala alphabet in the training process. At the same time, we must drop the output layer of the pre-trained English model, because it builds to identify 26 different English letters and it could not identify Sinhala letters through that. Then replace it with the size of our Sinhala alphabet and it uses 89 nodes instead of 26 default nodes. It means whatever the weights stored on output layer was deleted and recalculated depending on total number of letters in Sinhala alphabet and keeping the lower layer weights as it is.

Once we finaliz all the changes which is needed to train the model using Sinhala speech data, it will start the training followed with the testing process using transfer learning from English to Sinhala. It used almost same way as we did in the training from it scratch. However, here in transfer learning initially load the weights from the source model, which known as English model, and then drop the top layer and readjust the weights only on that layer depending on the target model Sinhala speech data as it shown in figure 4.4 together with the support of alphabet and scorer.



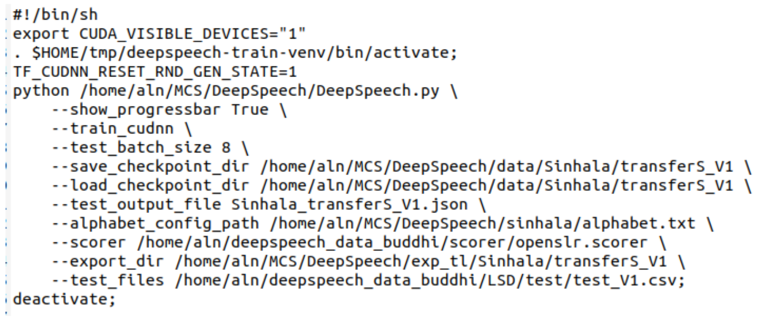
```
#!/bin/sh

. $HOME/tmp/deepspeech-train-venv/bin/activate;

export CUDA_VISIBLE_DEVICES="0,1,2,3"
python /home/aln/MCS/DeepSpeech/DeepSpeech.py \
    --train_cudnn \
    --drop_source_layers 1 \
    --alphabet_config_path /home/aln/MCS/DeepSpeech/sinhala/alphabet.txt \
    --scorer /home/aln/deepspeech_data_buddhi/scorer/openslr.scorer \
    --save_checkpoint_dir /home/aln/MCS/DeepSpeech/data/Sinhala/transferS_V1 \
    --load_checkpoint_dir /home/aln/MCS/DeepSpeech/data/English/deepspeech-0.9.3-checkpoint \
    --train_files /home/aln/deepspeech_data_buddhi/LSD/train/train_V1.csv \
    --dev_files /home/aln/deepspeech_data_buddhi/LSD/dev/dev_V1.csv \
    --test_files /home/aln/deepspeech_data_buddhi/LSD/test/test_V1.csv \
    --use_allow_growth true ;
deactivate;
```

Figure 4.4: Sinhala DeepSpeech model training without augmentation and optimization using transfer learning

Once it completed the training after approximately 48 hours, it will create checkpoint directory as it shown in the path in the script to save trained model. Then we will feed it into our testing script, as it showed in the figure 4.5 and evaluate the model accuracy in terms of WER and CER.

As it mentions in the DeepSpeech documentation, we must feed Sinhala alphabet and the language model or an external scorer including the training and testing data files which we have created in csv format as it describes in section 4.1 Once it has completed the testing on the model it will create the trained model as .pb file, which known as protocol buffer file, and it saved into the place which we specified under the export_dir flag. Later we can use it for the creation on the mmap-able or memory mappable model on that. All the implementation must be done in the GPU environment, because our pre-trained English model was built on that same environment and it is more flexible to train quite large dataset without any memory issue.

In DeepSpeech, uses an algorithm called as CTC or connectionist temporal classification to align input audio sequences with output character sequences and it is not one-to-one mapping. Due that for each input there can be many output characters are mapped and it is necessary to select the most appropriate one



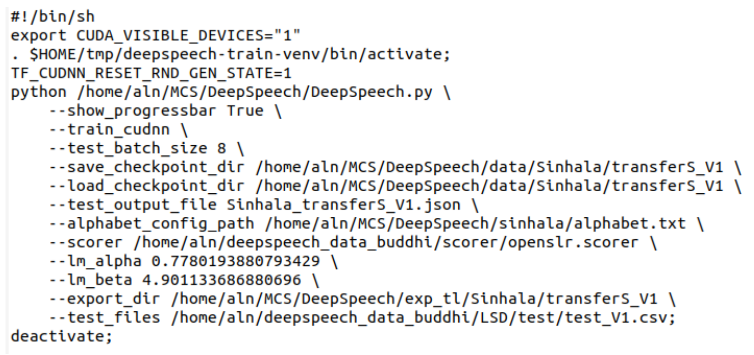
```
#!/bin/sh
export CUDA_VISIBLE_DEVICES="1"
. $HOME/tmp/deepspeech-train-venv/bin/activate;
TF_CUDNN_RESET_RND_GEN_STATE=1
python /home/aln/MCS/DeepSpeech/DeepSpeech.py \
    --show_progressbar True \
    --train_cudnn \
    --test_batch_size 8 \
    --save_checkpoint_dir /home/aln/MCS/DeepSpeech/data/Sinhala/transferS_V1 \
    --load_checkpoint_dir /home/aln/MCS/DeepSpeech/data/Sinhala/transferS_V1 \
    --test_output_file Sinhala_transferS_V1.json \
    --alphabet_config_path /home/aln/MCS/DeepSpeech/sinhala/alphabet.txt \
    --scorer /home/aln/deepspeech_data_buddhi/scorer/openslr.scorer \
    --export_dir /home/aln/MCS/DeepSpeech/exp_tl/Sinhala/transferS_V1 \
    --test_files /home/aln/deepspeech_data_buddhi/LSD/test/test_V1.csv;
deactivate;
```

Figure 4.5: Sinhala DeepSpeech model testing without augmentation and optimization using transfer learning

based on the probability value. Language model plays the major role for that optimal value or beam selection. However, when we build the language model initially, we were considered the default alpha and beta values and it was designed targeting on English. Anyway, by executing the lm_optimizer.py python script by feeding our test data files and checkpoint directory folder, which save all the necessary information on the model training to generate the default_alpha and default_beta values for Sinhala. Once we identify that specific value then we can easily replace the default alpha and beta with our identified values new values 0.7780193880793429 and 4.901133686880696 respectively in the testing script as it shown in figure 4.6. With compared to the performances on transfer learning-based Sinhala ASR and the model build just only considering the Sinhala speech data from it scratch, we can see the improvement on prediction in transfer learning based one. More explanations on result analysis are explained in next chapter.

4.4.1 Applying Data Augmentation

Data augmentation is the technique that we used to generalized our machine learning model and helps to cover problem space as much as possible for better performance. In this study main aim is to build an ASR system for low resource Sinhala language. Through data augmentation we can easily handle the data scarcity issue. In Mozilla's DeepSpeech, it has 4 domain areas to concern on data augmentation as sample, signal, spectrogram, and feature. In each domain



```
#!/bin/sh
export CUDA_VISIBLE_DEVICES="1"
. $HOME/tmp/deepspeech-train-venv/bin/activate;
TF_CUDNN_RESET_RND_GEN_STATE=1
python /home/aln/MCS/DeepSpeech/DeepSpeech.py \
    --show_progressbar True \
    --train_cudnn \
    --test_batch_size 8 \
    --save_checkpoint_dir /home/aln/MCS/DeepSpeech/data/Sinhala/transferS_V1 \
    --load_checkpoint_dir /home/aln/MCS/DeepSpeech/data/Sinhala/transferS_V1 \
    --test_output_file Sinhala_transferS_V1.json \
    --alphabet_config_path /home/aln/MCS/DeepSpeech/sinhala/alphabet.txt \
    --scorer /home/aln/deepspeech_data_buddhi/scorer/openslr.scorer \
    --lm_alpha 0.7780193880793429 \
    --lm_beta 4.901133686880696 \
    --export_dir /home/aln/MCS/DeepSpeech/exp_tl/Sinhala/transferS_V1 \
    --test_files /home/aln/deepspeech_data_buddhi/LSD/test/test_V1.csv;
deactivate;
```

Figure 4.6: Sinhala DeepSpeech model testing without augmentation using transfer learning considering optimization

there are various sub techniques are mentioned and depending on the probability value we can command the which data proportion gets augmented. However, when considering on the signal domain, with respect to the order we specify the augmentation technique will matter for the result, but in others it does not matter.

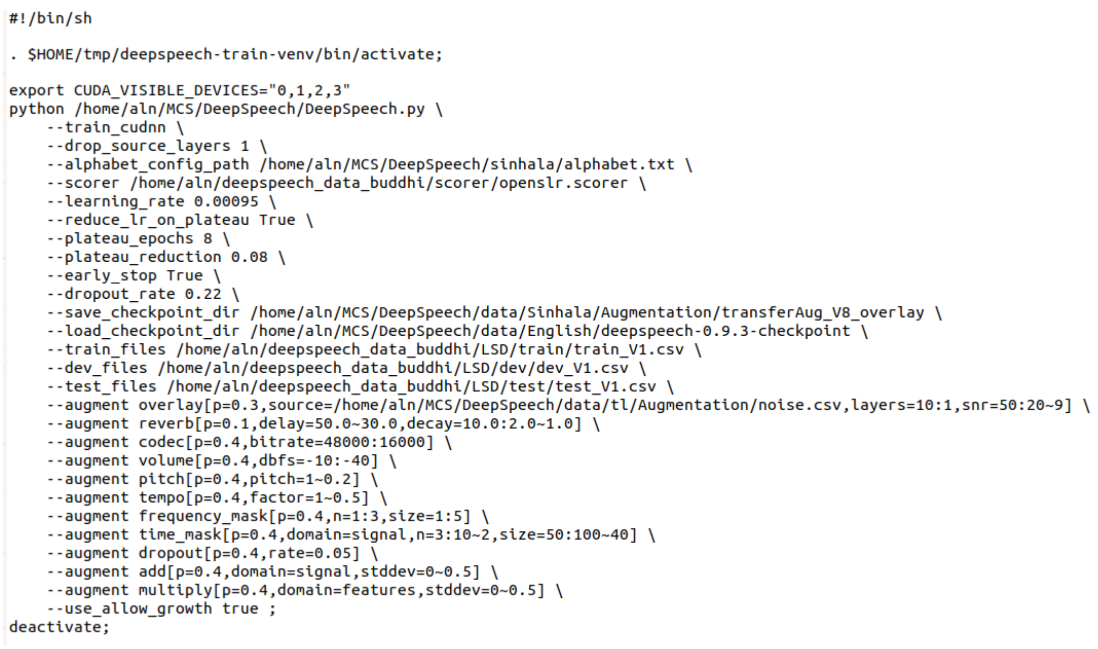
Normally in machine learning model data augmentation applied only to the training dataset and it can be applied as offline or online augmentation. In Mozilla's DeepSpeech, augmentation is applied during the training time with the help from set of libraries. Due to that we can easily develop the model using GPU machine and the augmentation code sample is also mentioned within the main training python script. However, it is important that we apply appropriate amount of augmentation, which can be denoted within the 1st parameter in the augmentation syntax, by considering various aspects, such as changing pitch, noise insertion, changing volume, changing tempo, shifting time etc.

Here we did few experiments with the augmentation technique to select the most appropriate set of augmentations in each domain by considering the default parameter values as it explains in the DeepSpeech documentation (*Welcome to Deepspeech's documentation* n.d.). Initially we start with the probability values as 10% and then gradually increased it and check the model accuracy to pick the most suitable architecture for the data augmented transfer learning model from English to Sinhala. To identify the best data augmented model we did 6 experiments by changing the augmentation techniques with its probabilities. Initially start with the few techniques such as reverb, volume, pitch, and tempo to check whether it will improve the overall accuracy or reduce the WER in the training model. Then we have noticed that it improved bit and secondly, we additionally applied codec, frequency mask, time mask, dropout, add, and multiply and remove reverb in the augmentation list and it also improved bit on the accuracy compared to the initial one.

Then we noticed that our model may be overfitting and to tackle that we introduced dropout rate as 0.22 within our training script and then it gives much better results so we thought to use it for other experiments as well. When we applied reverb augmentation for that previous experiment it did not give us much improvement in the results so we change the number of samples that we applied it. At that time, it gave us much better accuracy with the model and when we apply remaining augmentation technique called overlay it improved that previous experiment accuracy and gave us the best performing parameters and the probabilities on each augmentation techniques which adapt with our Sinhala speech dataset. Considering all the experiment our last model gave us the best results with the prediction and it is showed that via using transfer learning and the data augmentation we can improve the performances of ASR system compared to the baseline ASR model and the training script that we used for that is shown in the figure 4.7.

4.5 Exporting a model for inference

Once we have completed the model training and validation, a created .pb model for that particular training is exported into the export directory. It can be mentioned within the testing script file under the export_dir flag as it is given in the figure 4.5. In Mozilla's DeepSpeech it will generate it as .pb file which known as protocol buffer file. Most of the training model they will create this .pb file at the end, but still it has negative impacts. However, these files are not memory mappable one and due to that reason, it consumes lots of memory capacity and in DeepSpeech it took nearly 700MB in size(Talasila 2020). If we want to refer this created model file by using its file descriptor, then it is necessary to convert



```
#!/bin/sh

. $HOME/tmp/deepspeech-train-venv/bin/activate;

export CUDA_VISIBLE_DEVICES="0,1,2,3"
python /home/aln/MCS/DeepSpeech/DeepSpeech.py \
    --train_cudnn \
    --drop_source_layers 1 \
    --alphabet_config_path /home/aln/MCS/DeepSpeech/sinhala/alphabet.txt \
    --scorer /home/aln/deepspeech_data_buddhi/scorer/openslr.scorer \
    --learning_rate 0.00095 \
    --reduce_lr_on_plateau True \
    --plateau_epochs 8 \
    --plateau_reduction 0.08 \
    --early_stop True \
    --dropout_rate 0.22 \
    --save_checkpoint_dir /home/aln/MCS/DeepSpeech/data/Sinhala/Augmentation/transferAug_V8_overlay \
    --load_checkpoint_dir /home/aln/MCS/DeepSpeech/data/English/deepspeech-0.9.3-checkpoint \
    --train_files /home/aln/deepspeech_data_buddhi/LSD/train/train_V1.csv \
    --dev_files /home/aln/deepspeech_data_buddhi/LSD/dev/dev_V1.csv \
    --test_files /home/aln/deepspeech_data_buddhi/LSD/test/test_V1.csv \
    --augment overlay[p=0.3,source=/home/aln/MCS/DeepSpeech/data/tl/Augmentation/noise.csv,layers=10:1,snr=50:20~9] \
    --augment reverb[p=0.1,delay=50.0~30.0,decay=10.0:2.0~1.0] \
    --augment codec[p=0.4,bitrate=48000:16000] \
    --augment volume[p=0.4,dbfs=-10:-40] \
    --augment pitch[p=0.4,pitch=1~0.2] \
    --augment tempo[p=0.4,factor=1~0.5] \
    --augment frequency_mask[p=0.4,n=1:3,size=1:5] \
    --augment time_mask[p=0.4,domain=signal,n=3:10~2,size=50:100~40] \
    --augment dropout[p=0.4,rate=0.05] \
    --augment add[p=0.4,domain=signal,stddev=0~0.5] \
    --augment multiply[p=0.4,domain=features,stddev=0~0.5] \
    --use_allow_growth true ;
deactivate;
```

Figure 4.7: Sinhala DeepSpeech model training with augmentations using transfer learning

this .pb file into the memory mappable version of it and it is in .pbmm format. Therefore, to convert it we used graphdef tool which is come in default in TensorFlow (Håkansson & Hoogendijk 2020). By executing the following command we can easily export memory mappable version of our .pb file then we will use it for the evaluation purpose which is described in details in the next chapter.

```
./convert\_graphdef\_memmapped\_format
—in\_graph =
/home/aln/MCS/DeepSpeech/exp\_tl/Sinhala/tra
nsferAug\_V5\_A40/output\_graph.pb
—out\_graph =
/home/aln/MCS/DeepSpeech/exp\_tl/Sinhala/tra
nsferAug\_V5\_A40/output\_graph.pbmm
```

Chapter 5

Results

In this chapter the results that has been obtained during the research through the several experiments is described under several sections. In the first section it will discuss the evaluation matrices that we have selected to evaluate the performances on Sinhala ASR system. In the second section it represents the results obtained from the Mozilla's DeepSpeech baseline model and the transfer learning approach.

5.1 Evaluation Matrices

When we are creating any sort of machine learning model, it is important to evaluate how good it worked within the given environment by considering the prediction. Or else here we will evaluate how well the trained model do the predictions. When it comes to the ASR systems, the Word Error Rate (WER) and the Character Error Rate (CER) are the most important and commonly used evaluation matrices. In here it will compare the predicted result text from our trained model for the given speech file on desired language, with the actual transcription on that corresponding speech file and then it will do the calculations on the correlation between those two results (Agarwal & Zesch 2020). Most of the speech recognition researches evaluation is conducted by analyzing the WER and

CER changes among each model. Since we are comparing the results in several version of the trained model such as baseline model which is built from it scratch, transfer learning model from English to Sinhala and transfer learning model with augmentations, we will consider the same test dataset for the evaluation.

- **Word Error Rate**
  To determine the accuracy in most of the speech-based models we have used WER, which is known as a common method of performance of speech recognition systems. In order to calculate the WER, we will consider several parameters such as total number of words within the given transcript, substitutions, deletions, and insertions as it represented in the equation 5.1. To calculate the substitutions (S), deletions (D) and insertions (I), first it is necessary to have an alignment between the output of the desired Mozilla's DeepSpeech model with the given reference transcript on the speech file. After that, we can easily calculate the required values for each parameter and then applied it to the WER equation given below by calculation the total number of words (N) in the given reference transcript of speech file. In DeepSpeech model testing phase it will calculate the WER automatically and we can further analyse the alignment between output and the actual transcripts.

$$WER = \frac{S + E + I}{N} \tag{5.1}$$

- **Character Error Rate**
  During the speech recognition model training we can easily visualized that the training and validation evaluation matrices gradually dropped with respect to the time after we applied dropout rate as 0.22. At the end of the model training, it will automatically print the WER and CER in the console and same thing happened in the testing as well. However, through WER we can only get some idea on the accuracy of our language model which we have created under the section 4.4. In speech recognition we mainly

focusing on identifying characters with respect to the given language and it is necessary for translate spoken utterance into text. The character error rate or CER is basically concerned on how accurately Mozilla's DeepSpeech is capable enough to identify the character with the support from the alphabet in the desired language. In our study it will make sure that how accurately our acoustic model for Sinhala is worked. Considering the CER value, we can consider that whether our acoustic model requires fine tuning or not. If we received high CER and a low WER then it will give us a hint that we must perform fine tuning to increase the performances on acoustic model (*Welcome to Deepspeech's documentation* n.d.).

5.2 Experiments and Results

Before we evaluate the performances on each model it is necessary to trained the model by using given speech dataset as it explained in the chapter 4. As it discussed we built the ASR model by using the external support from the language model and adjusting some default parameters to get high accuracy with the final predictions. For the training purpose we have been used Sinhala speech data coverage with 94 male and female speakers and in validation we have been used 11 speakers. Finally for the testing purpose we have used 8 speakers and among them half of male and others are female. We used that testing dataset to measure the evaluation on our developed model by comparing the WER changes.

5.2.1 Results from Baseline model

Based on the previous study conducted for the Sinhala speech data we have replaced few hyperparameters and keep few of them same as the default values in the pre-trained Mozilla's DeepSpeech English model. Here we changed the learning rate as 0.00095 which is previously was 0.00001 and applied dropout rate as 0.22 together with 0.08 in plateau reduction. However, developing the ASR model from it scratch can be done easily by referring the DeepSpeech documentation. Anyway, language specific characteristics are different from English and Sinhala it is necessary to feed language model as an external input to the model and for that we have used 4-grams LM. The results obtained from this baseline model used for a comparison on the prediction experimental models trained with transfer learning.

The results that we gained from the baseline Sinhala ASR model have showed in the table 5.1 by considering the dataset that explained in the section 4.1. In this table it represents the result we achieved with several hyperparameter changes in terms of WER and CER and the best one is highlighted in bold. According to the results from the testing on baseline model without considering any data augmentation, when we changed the learning rate as 0.0095 and applying the 0.08 in plateau handling together with 0.22 dropout rate, we received the best performing baseline model.

Model	WER (%)	CER (%)
LR= 0.0001	51.28	25.9
LR= 0.0001 + Plateau handling	66.51	42.0
LR= 0.0095	47.81	25.02
LR= 0.0095 + Plateau handling	**41.62**	**20.60**

Table 5.1: Obtained results for baseline ASR model which trained from it scratch without considering data augmentation

After that, we have taken above highlighted model and apply data augmentation by considering 4 techniques such as reverb, volume, pitch, and tempo with defaults parameters which mentioned in the documentation. Through that we achieved 23.52% on CER and 43.58% on WER.

5.2.2 Results from transfer learning model

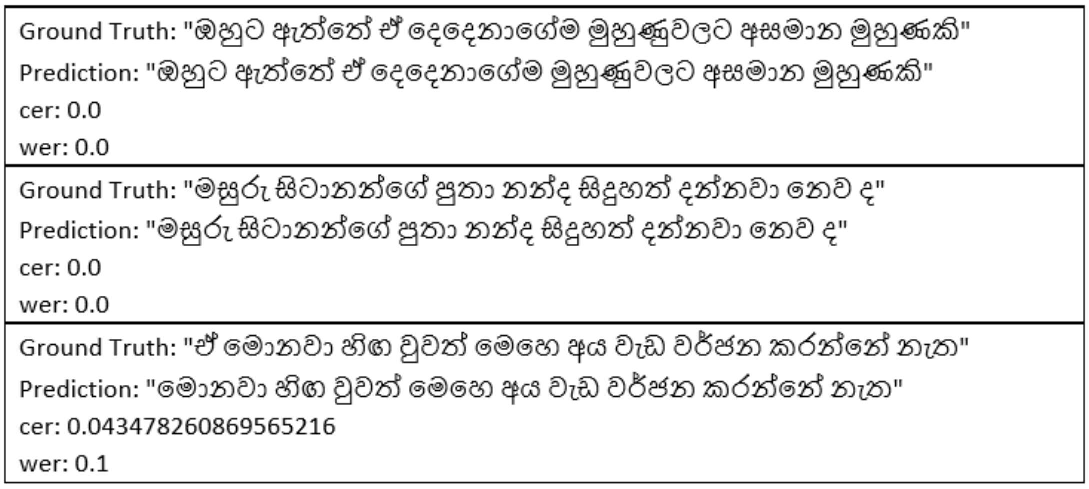
From 2019 onwards researchers were noticed that the e2e speech models must have higher data coverage to get the good accuracy in the prediction. However, Sinhala is known as low resource language which contained complex language features, therefore it is necessary to have some mechanism to handle that data scarcity issue. For that in most of the deep learning-based model used transfer learning approach and in DeepSpeech documentation also mention how to apply it from source to target model.

As explained in the previous chapter, we have used pre-trained English model as our source model and Sinhala one as target. Here we have selected learning rate as 0.0001 and feed alphabet and the scorer or language model as an external input to the model. Once we executed the script it will rearrange and recalculate the output layer weights according to the Sinhala alphabet by keeping preceding once as it is. Later we applied optimization on our created language model by assigning alpha and beta values which are specific to the Sinhala into the testing script. Then observe the changes on WER and CER and the best results on prediction comes with that model and it is highlighted in bold in the table 5.2. With compared to the best baseline ASR model prediction without considering data augmentation, evaluation matrices improvement in transfer-learning based model is represented within the bracket in an italic. In figure 5.1, the random results or transcription prediction on the below highlighted model is shown in term of WER and CER on the test Sinhala dataset.

Model	WER (%)	CER (%)
Without LM	84.31 *(+42.69)*	25.93 *(+5.33)*
Without scorer optimization	29.26 *(-12.36)*	10.80 *(-10.52)*
With scorer optimization	**22.93 *(-18.69)***	**8.84 *(-11.76)***

Table 5.2: Obtained results for transfer-learning based ASR model without considering data augmentation

According to the results when we consider transfer learning application from English model to Sinhala model then we will get better prediction. In the other hand, here our CER is little bit higher and WER is lower, so it means we need to apply fine tuning on our acoustic model. Therefore, we are considering on applying data augmentation for the best performing transfer learning model.


Ground Truth: "ඔහුට ඇත්තේ ඒ දෙදෙනාගේම මුහුණුවලට අසමාන මුහුණකි"
Prediction: "ඔහුට ඇත්තේ ඒ දෙදෙනාගේම මුහුණුවලට අසමාන මුහුණකි"
cer: 0.0
wer: 0.0

Ground Truth: "මසුරු සිටානන්ගේ පුතා නන්ද සිදුහත් දන්නවා නෙව ද"
Prediction: "මසුරු සිටානන්ගේ පුතා නන්ද සිදුහත් දන්නවා නෙව ද"
cer: 0.0
wer: 0.0

Ground Truth: "ඒ මොනවා හිඟ වුවත් මෙහෙ අය වැඩ වර්ජන කරන්නේ නැත"
Prediction: "මොනවා හිඟ වුවත් මෙහෙ අය වැඩ වර්ජන කරන්නේ නැත"
cer: 0.043478260869565216
wer: 0.1

Figure 5.1: Transcriptions generated with transfer-learning based ASR model without considering data augmentation

5.2.3 Results from transfer learning model using data augmentation

Previously it was noticed that our transfer learning based acoustic model need fine tuning and for that here we have used data augmentation. In Mozilla's DeepSpeech data augmentation can be classified as 4 domain such as sample, signal, spectrogram, and feature. It will perform the augmentation with respect to the order that we have stated in the training script. Moreover, it is necessary to select the appropriate amount of sample which we need to augmented during the model training and it can easily represent as a probability value in the augmentation syntax. In order to identify the most appropriate augmentation types together with appropriate probabilities in each according to the Sinhala language have been identified using several experiments.

Initially we start our first experiment by considering the previously obtained best transfer learning model and apply 4 types of augmentation techniques such as reverb, volume, pitch, and tempo, which is denoted as Augmented Model 1 in 5.3. According to the literature on data augmentation application in Mozilla's DeepSpeech almost all the previous approach has used default values which it describes in the documentation(Talasila 2020). Therefore, we also used it and initially apply it for 10% of the training samples and then increase it up to 20%. Then we have observed the result on WER and CER with and without using language optimization as it shown in table 5.3. According to the result when we increase the probability then it will reduce the performance. Then we

Model	Language Optimization	Amount (%)	WER (%)	CER (%)
Augmented Model 1	No	10	26.24	9.44
	Yes	10	24.74	8.46
	No	20	27.45	10.62
	Yes	20	26.24	10.48

Table 5.3: Obtained results for 1st experiment in data augmented transfer-learning model

were considered other data augmentation types such as code, frequency mask, time mask, dropout, add, and multiply and it represents as Augmented Model 2. First apply 10% for each and then apply 20% and observed the performances as it shown in the table 5.4. It also behaves same as the previous one and reduce the performances when we are increasing the amount we are applying during the training. In 3rd experiment we have additionally added dropout rate as 0.22

Model	Language Optimization	Amount (%)	WER (%)	CER (%)
Augmented Model 2	No	10	24.59	8.92
	Yes	10	23.98	8.82
	No	20	25.94	8.89
	Yes	20	30.47	11.44

Table 5.4: Obtained results for 2nd experiment in data augmented transfer-learning model

and observed the WER and CER changes for the training data sample application on 10%, and 20%. When comparing the results as it shown in the table 5.5, we achieved best performances for the training model (known as Augmented Model 3) compared to the previous experimental approaches without considering the LM optimization in 20% of data samples augmentation. In 10% of data samples augmentation, we achieved best compared to the previous two experiments with

Model	Language Optimization	Amount (%)	WER (%)	CER (%)
Augmented Model 3	No	10	23.07	8.73
	Yes	**10**	**21.41**	**7.83**
	No	**20**	**21.72**	**8.62**
	Yes	20	24.59	8.59

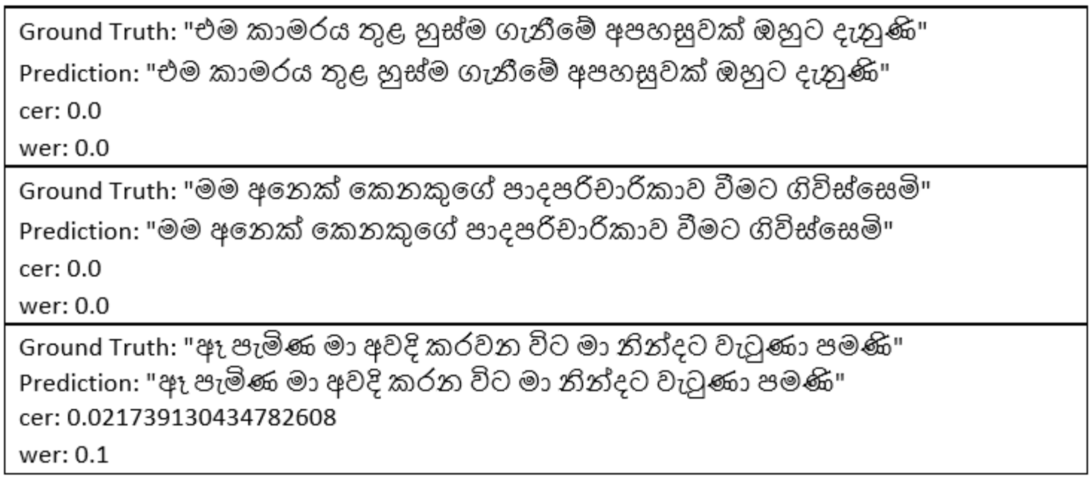
Table 5.5: Obtained results for 3rd experiment in data augmented transfer-learning model

LM optimization. In the 4th experiment we have additionally added plateau handling of 0.08 and changed the default learning rate as 0.00095 and observed the performances for the augmented sample application for 10%, 20% and 40% as it shown in the table 5.6. Here we have kept reverb data augmentation type application for the data samples as 10% fixed and change other types according to the given order. Because in previous experiment we noticed that if we increase reverb probability more than 10% then it will reduce the overall performances. According to the results in the table 5.6 it gave much better results for the model (known as Augmented Model 4) compared to all other experimental models. In figure 5.2, the random results or transcription prediction on the below highlighted model is shown in term of WER and CER on the test Sinhala dataset.

Model	Language Optimization	Amount (%)	WER (%)	CER (%)
Augmented Model 4	No	10	23.07	8.73
	Yes	10	21.41	7.83
	No	20	20.81	8.11
	Yes	20	22.32	8.16
	No	40	20.96	8.40
	Yes	**40**	**17.34**	**6.05**

Table 5.6: Obtained results for 4th experiment in data augmented transfer-learning model

In the 5th experiment we have added additionally overlay augmentation type by considering the default parameters which mentioned in the documentation for the 4th experiment training script and observed the changes in WER and CER.

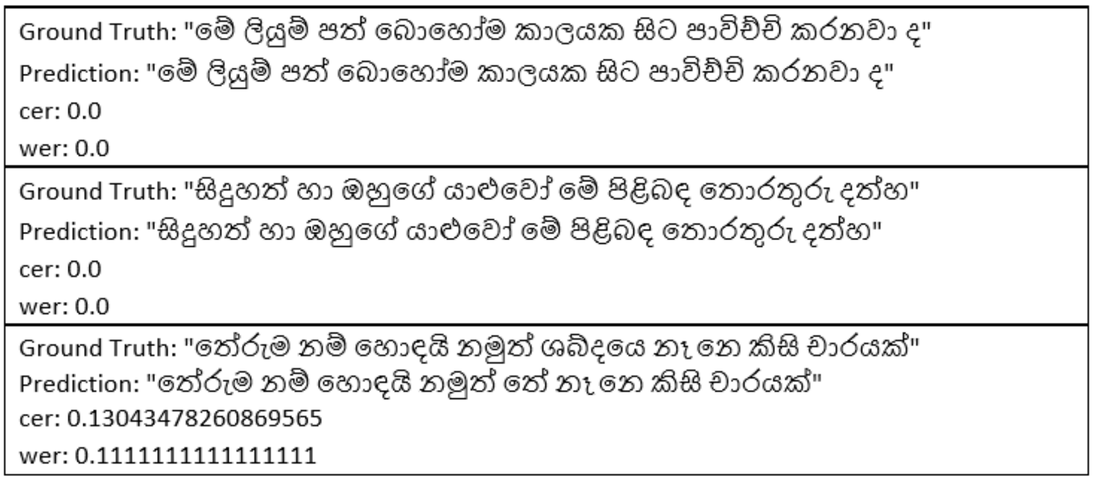


Ground Truth: "එම කාමරය තුළ හුස්ම ගැනීමේ අපහසුවක් ඔහුට දැනුණි"
Prediction: "එම කාමරය තුළ හුස්ම ගැනීමේ අපහසුවක් ඔහුට දැනුණි"
cer: 0.0
wer: 0.0

Ground Truth: "මම අනෙක් කෙනකුගේ පාදපරිචාරිකාව වීමට ගිවිස්සෙමි"
Prediction: "මම අනෙක් කෙනකුගේ පාදපරිචාරිකාව වීමට ගිවිස්සෙමි"
cer: 0.0
wer: 0.0

Ground Truth: "ඈ පැමිණ මා අවදි කරවන විට මා නින්දට වැටුණා පමණි"
Prediction: "ඈ පැමිණ මා අවදි කරන විට මා නින්දට වැටුණා පමණි"
cer: 0.021739130434782608
wer: 0.1

Figure 5.2: Transcriptions generated with data augmented transfer-learning based model without overlay type



Ground Truth: "මේ ලියුම් පත් බොහෝම කාලයක සිට පාවිච්චි කරනවා ද"
Prediction: "මේ ලියුම් පත් බොහෝම කාලයක සිට පාවිච්චි කරනවා ද"
cer: 0.0
wer: 0.0

Ground Truth: "සිදුහත් හා ඔහුගේ යාළුවෝ මේ පිළිබඳ තොරතුරු දත්හ"
Prediction: "සිදුහත් හා ඔහුගේ යාළුවෝ මේ පිළිබඳ තොරතුරු දත්හ"
cer: 0.0
wer: 0.0

Ground Truth: "තේරුම නම් හොඳයි නමුත් ශබ්දයෙ නෑ නෙ කිසි චාරයක්"
Prediction: "තේරුම නම් හොඳයි නමුත් තේ නෑ නෙ කිසි චාරයක්"
cer: 0.13043478260869565
wer: 0.1111111111111111

Figure 5.3: Transcriptions generated with data augmented transfer-learning based model without overlay type

Model	Language Optimization	Amount (%)	WER (%)	CER (%)
Augmented Model 4	No	10	20.36	8.10
	Yes	10	19.15	6.52
	No	20	21.42	8.24
	Yes	20	19.00	6.35
	No	30	21.87	8.62
	Yes	**30**	**17.19**	**5.97**
	No	40	22.62	8.76
	Yes	40	19.00	6.79

Table 5.7: Obtained results for 5^{th} experiment in data augmented transfer-learning model

According to the results we achieved from that experiment as it shown in the table 5.7, we cannot apply more than 30% of overlay augmentation to the data samples and if we applied 30% of overlay, 10% of reverb together with other augmentation type considering the default parameters we can achieved the best performance on data augmented transfer-learning based ASR. If we compared it with the data augmented baseline ASR model, it showed significant reduction in WER and CER and it is represented within the bracket in table 5.7. In figure 5.3, the random results or transcription prediction on the above highlighted model is shown in term of WER and CER on the test Sinhala dataset.

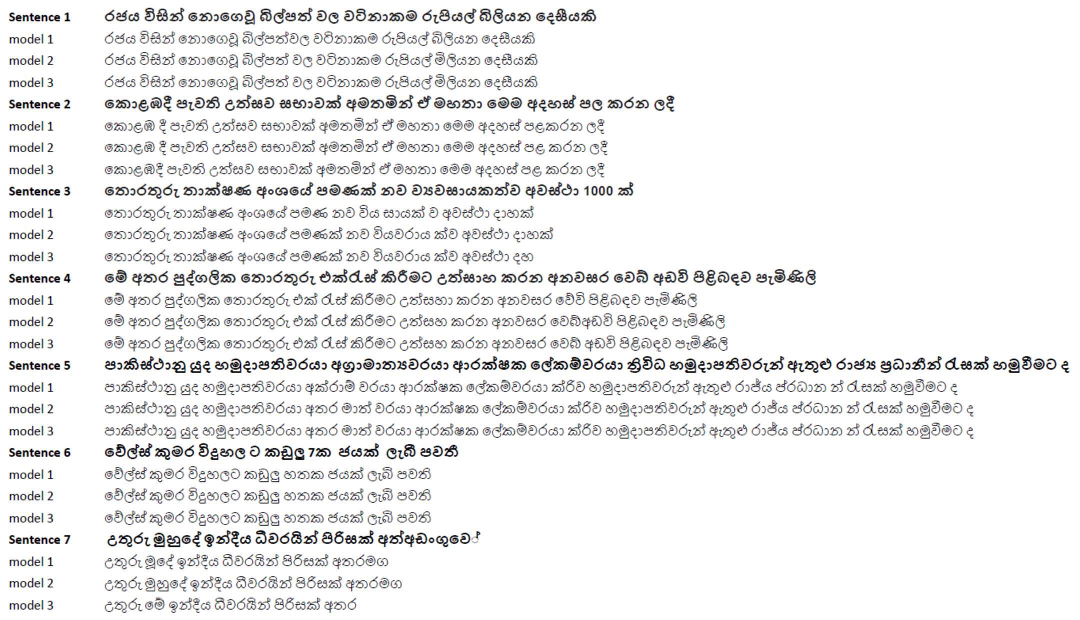
Overall, we can conclude if we apply 10% of reverb and 30% of overlay with other types keep it as 10% will give the outstanding performance in the transfer learning as it shown in Augmented Model 5. Here with respect to the WER it reduced by 26.39% and in CER it reduced by 17.55% compared to the baseline augmented model.

5.3 User evaluation

In order to evaluate our trained Sinhala ASR models with or without using data augmentations, we have used pre recorded 7 sentences with different lengths considering both males and females. Those sentences were named as test_1, test_2, test_3 etc. Among those sentences one sentence was gathered from the model training dataset and remaining ones taken from the random males and females. Spoken utterances were recorded within their preferable environment and once it successfully recorded we fed it into the generated .pbmm model and evaluate the performances comparing predicted one with its original using the below command.

```
deepspeech --model
/home/aln/MCS/DeepSpeech/exp\_tl/Sinhala/transferS\_V1/
output\_graph.pbmm --scorer
/home/aln/deepspeech\_data\_buddhi/scorer/openslr.scorer
 --audio /home/aln/deepspeech\_data\_buddhi/Test/test\_1.wav
```

For the evaluation we have used three different models. Firstly, starts with transfer learned model without considering any data augmentation and it mention as model 1. Then for the 2nd model we have chosen data augmented transfer learned model which we created under the 4th experiment and it mention as model 2. Finally, we have considered data augmented transfer learned model which we created under the 5th experiments and it mention as model 3. Model predictions on each test sentences are illustrated in figure 5.4.



Sentence 1	**රජය විසින් නොගෙවූ බිල්පත් වල වටිනාකම රුපියල් බිලියන දෙසීයකි**
model 1	රජය විසින් නොගෙවූ බිල්පත්වල වටිනාකම රුපියල් බිලියන දෙසීයකි
model 2	රජය විසින් නොගෙවූ බිල්පත් වල වටිනාකම රුපියල් මිලියන දෙසීයකි
model 3	රජය විසින් නොගෙවූ බිල්පත් වල වටිනාකම රුපියල් මිලියන දෙසීයකි
Sentence 2	**කොළඹදී පැවති උත්සව සභාවක් අමතමින් ඒ මහතා මෙම අදහස් පල කරන ලදී**
model 1	කොළඹ දී පැවති උත්සව සභාවක් අමතමින් ඒ මහතා මෙම අදහස් පළකරන ලදී
model 2	කොළඹ දී පැවති උත්සව සභාවක් අමතමින් ඒ මහතා මෙම අදහස් පළ කරන ලදී
model 3	කොළඹදී පැවති උත්සව සභාවක් අමතමින් ඒ මහතා මෙම අදහස් පළ කරන ලදී
Sentence 3	**තොරතුරු තාක්ෂණ අංශයේ පමණක් නව ව්‍යවසායකත්ව අවස්ථා 1000 ක්**
model 1	තොරතුරු තාක්ෂණ අංශයේ පමණ නව විය සායක් ව අවස්ථා දාහක්
model 2	තොරතුරු තාක්ෂණ අංශයේ පමණක් නව වියවරාය ක්ව අවස්ථා දාහක්
model 3	තොරතුරු තාක්ෂණ අංශයේ පමණක් නව වියවරාය ක්ව අවස්ථා දහ
Sentence 4	**මේ අතර පුද්ගලික තොරතුරු එක්රැස් කිරීමට උත්සාහ කරන අනවසර වෙබ් අඩවි පිළිබඳව පැමිණිලි**
model 1	මේ අතර පුද්ගලික තොරතුරු එක් රැස් කිරීමට උත්සහා කරන අනවසර වේවි පිළිබඳව පැමිණිලි
model 2	මේ අතර පුද්ගලික තොරතුරු එක් රැස් කිරීමට උත්සහ කරන අනවසර වෙබ්අඩවි පිළිබඳව පැමිණිලි
model 3	මේ අතර පුද්ගලික තොරතුරු එක් රැස් කිරීමට උත්සහ කරන අනවසර වෙබ් අඩවි පිළිබඳව පැමිණිලි
Sentence 5	**පාකිස්ථානු යුද හමුදාපතිවරයා අග්‍රාමාත්‍යවරයා ආරක්ෂක ලේකම්වරයා ත්‍රිවිධ හමුදාපතිවරුන් ඇතුළු රාජ්‍ය ප්‍රධානීන් රැසක් හමුවීමට ද**
model 1	පාකිස්ථානු යුද හමුදාපතිවරයා අක්රාම් වරයා ආරක්ෂක ලේකම්වරයා ක්රිව හමුදාපතිවරුන් ඇතුළු රාජ්ය ප්රධාන න් රැසක් හමුවීමට ද
model 2	පාකිස්ථානු යුද හමුදාපතිවරයා අතර මාත් වරයා ආරක්ෂක ලේකම්වරයා ක්රිව හමුදාපතිවරුන් ඇතුළු රාජ්ය ප්රධාන න් රැසක් හමුවීමට ද
model 3	පාකිස්ථානු යුද හමුදාපතිවරයා අතර මාත් වරයා ආරක්ෂක ලේකම්වරයා ක්රිව හමුදාපතිවරුන් ඇතුළු රාජ්ය ප්රධාන න් රැසක් හමුවීමට ද
Sentence 6	**වේල්ස් කුමර විදුහල ට කඩුලු 7ක ජයක් ලැබී පවතී**
model 1	වේල්ස් කුමර විදුහලට කඩුලු හතක ජයක් ලැබි පවති
model 2	වේල්ස් කුමර විදුහලට කඩුලු හතක ජයක් ලැබි පවති
model 3	වේල්ස් කුමර විදුහලට කඩුලු හතක ජයක් ලැබි පවති
Sentence 7	**උතුරු මුහුදේ ඉන්දීය ධීවරයින් පිරිසක් අත්අඩංගුවේ**
model 1	උතුරු මූදේ ඉන්දීය ධීවරයින් පිරිසක් අතරමග
model 2	උතුරු මුහුදේ ඉන්දීය ධීවරයින් පිරිසක් අතරමග
model 3	උතුරු මේ ඉන්දීය ධීවරයින් පිරිසක් අතර

Figure 5.4: Model Predictions

Chapter 6

Conclusion

In conclusion, this thesis has discussed the applicability of Mozilla's DeepSpeech for speech recognition on low resource Sinhala language. Initially, it starts with the brief introduction on the automatic speech recognition and then focused on the speech recognition studies conducted for Sinhala. In chapter 2 it described background information which align with the DeepSpeech architecture such as recurrent neural network, transfer learning concept in speech recognition and sound related feature analysis. In chapter 3 some of the key observation from speech domain analysis together with their approach are highlighted and discussed further in details by referring techniques which they have used, the data set and the conclusion on their studies. based on the information gain from previous studies our methodology and achieved experimental results are mentioned briefly in chapter 4 and chapter 5 respectively. In this chapter, some of the key observations which we encountered during the study are highlighted and represented in an appropriate manner.

6.1 The importance of language models

The automatic speech recognition is the system which accurately translate spoken utterances into the textual format. The most traditional ASR system architecture is known as statistical ASR, which consists of three different models such as acoustic, pronunciation and language model. The next architecture is e2e approach which comprised previously mentioned all three model into one. Language model or an external scorer which we created has been used in every training and testing scripts, because it will improve the model predictions significantly as it shown in the table 5.2. With that we can concluded that any ASR model is performed well on the prediction when we are using language model as a parameter and it will give impressive results while we are using language model together with its optimized values. In the table 5.3, table 5.4, table 5.5, table 5.6 and table 5.7 show that using LM and its optimization will increase the model performances in terms of WER and CER.

Another observation on language model is that increasing the data coverage or the size of the corpora used to create the 4-grams language model using KenLM toolkit has an impact to the overall performances. Sinhala is known as morphologically rich language which consists of complex grammatical features. Therefore, instead of using the dataset which gathered from Paraat and RedStart tools, we have used the dataset gathered from three different corpuses such as UCSC Novel corpora, chatbot and active learning coverage. Through that it will extract most of the features in Sinhala and incorporate it with the training model for better predictions.

6.2 Conclusion about the research questions

In this study we have concerned with two research questions such as "How to apply model level transfer to improve the performance of Speech Recognition on low resource language?" and "How to select the most appropriate target model for Sinhala language in the transfer learning?" Mozilla's DeepSpeech is an e2e speech recognition system which consists of the most informative documentation in it. It clearly explained how to apply transfer learning and the way of doing it discussed under the chapter 4.

Only few studies have been conducted with regarding Sinhala speech processing, due to that reason there are very few speech resources are available in the community for speech related analysis. To tackle that problem, we have used transfer learning from English to Sinhala.

Initially we were considered the transfer learning from pretrained English model to Sinhala without considering any data augmentation and with or without considering the language model optimization. According to the results which can be seen in table 5.2, it shows that it gives better results in terms of WER and CER when we are using transfer learning without considering LM optimization.

Later we have applied data augmentation and with or without considering the optimization on LM by changing the default alpha and beta values. Those values are calculated by considering the data coverage which we have used for the LM creation and replace the default value with the new values. In Mozilla's DeepSpeech it has 4 domains of augmentations and we have conducted few experiments on selecting the most appropriate augmented types with its probabilities as it described in the section 4.4.1. Through that we can identify the appropriate values for each hyperparameters and the most suitable probabilities for each augmentation types are it helps to identify the most suitable target Sinhala model for ASR in the transfer learning. However, we have identified corresponding values for the selected augmentation sub types those by only changing the probabilities and keeping other parameters as defaults ones which mentioned in the documentation.

According to the results obtained from the table 5.6, it concludes that when we have applied 10% of reverb type of augmentation with 40% of other augmentation types except overlay, together with dropout rate as 0.22 gives much better. Through that experiment it achieved an improvement of 11.92% in WER and 4.75% in CER compared with the transfer learning model without augmentation. Later we have applied overlay augmentation type and observed that much better result with it gives when we were applying 30% of overlay with the previous model by considering plateau handling on 0.08 rate. Through that it achieved an improvement of 12.07% in WER and 4.83% in CER compared with the transfer learning model without augmentation.

Based on the all the experiments conducted with data augmentation on target language it showed that this model gives the highest results on Sinhala utterance predictions and we have selected it as the most appropriate target model for Sinhala ASR.

6.3 Future works

In recent studies mainly focused on concerning low resource language in the speech analysis by using the new technique called transfer learning. Most of the time we are using English as our source model because in that pre-trained model it covered almost all the language specific features align with English and it can be transfer for other languages which has low resources. According to the observations on Sinhala ASR we can improve the performances by applying data augmentation. Through this study we identify the target model architecture by relying on the default values for augmentations. In future we can identify those parameters with respect to the data set coverage from various accent, style and gender based with an external support from a speech engineer. In addition to that we can incorporate this results in order to identify the spoken utterances with respect to the domain and we can integrate it into the real-world applications such as voice assistant for insurance company, supermarket, shopping mall in Sinhala.

Bibliography

Agarwal, A. & Zesch, T. (2020). Ltl-ude at low-resource speech-to-text shared task: Investigating mozilla deepspeech in a low-resource setting, *SwissText/KONVENS*. 31, 40, 47

Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G. et al. (2016). Deep speech 2: End-to-end speech recognition in english and mandarin, *International conference on machine learning*, PMLR, pp. 173–182. 29

Arora, S. J. & Singh, R. P. (2012). Automatic speech recognition: a review, *International Journal of Computer Applications* **60**(9). 23

Asgarian, A. (2020). An introduction to transfer learning.
URL: *https://georgian.io/an-introduction-to-transfer-learning/* 6, 18

Bansal, S., Kamper, H., Livescu, K., Lopez, A. & Goldwater, S. (2018). Pre-training on high-resource speech recognition improves low-resource speech-to-text translation, *arXiv preprint arXiv:1809.01431* . 29

Dahl, G. E., Sainath, T. N. & Hinton, G. E. (2013). Improving deep neural networks for lvcsr using rectified linear units and dropout, *2013 IEEE international conference on acoustics, speech and signal processing*, IEEE, pp. 8609–8613. 16

Dahl, G. E., Yu, D., Deng, L. & Acero, A. (2011). Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition, *IEEE Transactions on audio, speech, and language processing* **20**(1): 30–42. 24

Gamage, B., Pushpananda, R., Weerasinghe, R. & Nadungodage, T. (2020). Usage of combinational acoustic models (dnn-hmm and sgmm) and identifying the impact of language models in sinhala speech recognition, *2020 20th International Conference on Advances in ICT for Emerging Regions (ICTer)*, IEEE, pp. 17–22. 2, 3, 5, 9, 25, 33, 34, 35, 37, 39

Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep learning*, MIT press. 14

Graves, A., Fernández, S., Gomez, F. & Schmidhuber, J. (2006). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks, *Proceedings of the 23rd international conference on Machine learning*, pp. 369–376. 15

Håkansson, A. & Hoogendijk, K. (2020). Transfer learning for domain specific automatic speech recognition in swedish: An end-to-end approach using mozilla's deepspeech. 20, 30, 34, 37, 46

Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A. et al. (2014). Deep speech: Scaling up end-to-end speech recognition, *arXiv preprint arXiv:1412.5567* . 8, 9

Joshi, V., Zhao, R., Mehta, R. R., Kumar, K. & Li, J. (2020). Transfer learning approaches for streaming end-to-end speech recognition system, *arXiv preprint arXiv:2008.05086* . 4

Karunanayake, Y., Thayasivam, U. & Ranathunga, S. (2019). Sinhala and tamil speech intent identification from english phoneme based asr, *2019 International Conference on Asian Language Processing (IALP)*, IEEE, pp. 234–239. 3, 7, 31

Kunze, J., Kirsch, L., Kurenkov, I., Krug, A., Johannsmeier, J. & Stober, S. (2017). Transfer learning for speech recognition on a budget, *arXiv preprint arXiv:1706.00290* . 28

Lee, K.-F. (1988). On large-vocabulary speaker-independent continuous speech recognition, *Speech communication* **7**(4): 375–379. 24

Nadungodage, T. & Weerasinghe, R. (2011). Continuous sinhala speech recognizer, *Conference on Human Language Technology for Development, Alexandria, Egypt*, Citeseer, pp. 2–5. 2, 3, 7, 23, 24, 25

Nadungodage, T., Weerasinghe, R. & Niranjan, M. (2013). Efficient use of training data for sinhala speech recognition using active learning, *2013 International Conference on Advances in ICT for Emerging Regions (ICTer)*, IEEE, pp. 149–153. 26

Nadungodage, T., Welgama, V. & Weerasinghe, R. (2013). Developing a speech corpus for sinhala speech recognition, *ICON* **10**. 26

Pan, S. J. & Yang, Q. (2009). A survey on transfer learning, *IEEE Transactions on knowledge and data engineering* **22**(10): 1345–1359. 17, 18, 19

Pulugundla, B., Baskar, M. K., Kesiraju, S., Egorova, E., Karafiát, M., Burget, L. & Cernockỳ, J. (2018). But system for low resource indian language asr., *Interspeech*, pp. 3182–3186. 28

Sarkar, D. (2018). A comprehensive hands-on guide to transfer learning with real-world applications in deep learning, *Towards Data Science* **20**: 2020. 4

Sharma, S., Sharma, S. & Athaiya, A. (2017). Activation functions in neural networks, *towards data science* **6**(12): 310–316. 15, 18

Srivastava, B. M. L., Sitaram, S., Mehta, R. K., Mohan, K. D., Matani, P., Satpal, S., Bali, K., Srikanth, R. & Nayak, N. (2018). Interspeech 2018 low resource automatic speech recognition challenge for indian languages., *SLTU*, pp. 11–14. 28

Stoian, M. C., Bansal, S. & Goldwater, S. (2020). Analyzing asr pretraining for low-resource speech-to-text translation, *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, pp. 7909–7913. 4

Talasila, A. (2020). Fine-tuning mozilla deepspeech for the indian accent.
URL: *https://towardsdatascience.com/automatic-speech-recognition-for-the-indian-accent-91bb011ad169* 45, 52

Tomashenko, N., Caubrière, A. & Estève, Y. (2019). Investigating adaptation and transfer learning for end-to-end spoken language understanding from speech, *Interspeech 2019*, ISCA, pp. 824–828. 30

Torrey, L. & Shavlik, J. (2010). Transfer learning, *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, IGI global, pp. 242–264. 17, 19

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

Welcome to Deepspeech's documentation (n.d.).
URL: *https://deepspeech.readthedocs.io/en/r0.9/* 10, 14, 33, 38, 44, 49

Xu, H., Van Hai Do, X. X., Xiao, X. & Chng, E. (2015). A comparative study of bnf and dnn multilingual training on cross-lingual low-resource speech recognition., *Interspeech*, pp. 2132–2136. 27

Yi, J., Tao, J., Wen, Z. & Bai, Y. (2018). Language-adversarial transfer learning for low-resource speech recognition, *IEEE/ACM Transactions on Audio, Speech, and Language Processing* **27**(3): 621–630. 26, 27

Yu, C., Chen, Y., Li, Y., Kang, M., Xu, S. & Liu, X. (2019). Cross-language end-to-end speech recognition research based on transfer learning for the low-resource tujia language, *Symmetry* **11**(2): 179. 29

Zajechowski, M. (2014). Automatic speech recognition (asr) software–an introduction, *Web: https://usabilitygeek. com/automatic-speechrecognition-asr-software-an-introduction* . 2