Spoken Language Identification of Low-Resource Languages with Universal Phoneme Recognisers

Sparsh Rawal

Master of Science
Artificial Intelligence
School of Informatics
University of Edinburgh
2023

Abstract

The process of determining the language being used in a spoken or written text is known as language identification. The process of determining the language in an audio stream is known as spoken language identification. Because spoken languages vary so much in their auditory characteristics, it can be difficult to identify spoken languages. The goal of this research was to develop a language identification system that can identify languages by purely using the phoneme sequences generated by universal phoneme recognisers. The language identification system was further enhanced by incorporating transcript knowledge which enabled zero-shot language identification of low-resource languages and the effects of number of unseen languages was explored.

We methodically built a baseline language classifier which classifies the language based on the sequence of phonemes it observes. This system was then extended for language identification of low-resource languages and we obtained plausible model performance, indicating that zero-shot learning is possible, purely based on sequence of phonemes.

Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Sparsh Rawal*)

Acknowledgements

Peter Bell, my supervisor, for all of his support and guidance . It was an absolute honor to work on the project he recommended; during the past year, he has taught me a lot about automatic speech recognition and machine learning.

Ondřej Klejch, who jointly supervised the project, for providing a pretrained Universal phoneme recognizer without it, the project would have been far harder to complete on time.

My family and friends, for their love and support regardless of whether circumstances have been easy or hard.

Table of Contents

Chapter 1

Introduction

The process of determining the language being used in a spoken or written text is known as language identification (LID) and the process of determining the language in a stream of audio is termed as spoken language identification (S-LID). Effective S-LID systems have been created for languages like English, French, and Spanish thanks to extensive research and the availability of vast data resources in those languages. The goal of this research is to develop a LID system that can identify low-resource languages where high-quality data is scarce by using universal phoneme recognizers.

Language, as a quintessential human communication tool, plays an integral role in our daily lives, facilitating the exchange of ideas, emotions, and information across cultures and borders. Accurate and reliable LID systems are now crucial components of many technology solutions due to the increasing proliferation of digital content and the expanding diversity of languages spoken worldwide. These solutions facilitate effective communication to cross linguistic barriers, improve search engine performance, and improve user experience.

Huge advancements in LID have been realized recently as a result of the development of deep learning algorithms and the accessibility of big, annotated datasets. Modern LID systems, which frequently use deep neural networks, have displayed impressive performance in a wide range of widely spoken languages. However, dealing with languages that are poorly represented in the datasets that are available is a considerable challenge, giving rise to the concept of Low-Resource Language Identification (Low-Resource LID). In order to distinguish between distinct languages from a vast number of target languages, an ideal LID system must correctly and minutely exploit various aspects of speech information [45].

1.1 Motivation

There are now known to be 7,168 languages actively being spoken throughout the world, according to the most comprehensive source of information on the world's languages, Ethnologue [13]. Out of the 7,168 languages 3,045 are considered to be endangered. When native speakers of a language start to teach and speak another, more popular language to the younger generation, that language becomes endangered. Because of their nature, endangered languages frequently have fewer speakers than before, making it challenging to find out information about them.

Low-resource languages must be preserved because they symbolize distinctive cultural communities and traditions, protecting variety and priceless knowledge. Language technologies fill gaps in communities by facilitating better communication, resource access, and education. The use of different languages in technological solutions provides equality and accessibility. The development of language technologies for languages with limited resources can progress artificial intelligence and natural language processing.

A standardized set of phonemes (the smallest, most discrete components of sound in a language) referred to as Universal Phoneme Sequences (UPS) is intended to represent speech sounds in many languages. LID through the use of Universal Phoneme Sequences (UPS) is driven by its motivation to provide a more universal and flexible approach for language identification for diverse speech and audio processing applications. The goal of UPS is to capture phonetic characteristics that are shared by all languages. As a result, LID models can handle a wider variety of languages, including those with complex phonemic structures or minimal resource requirements. Common phonetic patterns and universal articulatory characteristics are captured by UPS. This makes it possible for LID models to concentrate on language specific phonetic features that represent linguistic identification. By using shared phonetic data, UPS may be able to support zero-shot or few-shot language recognition, in which the model may identify languages it hasn't been explicitly trained on.

1.2 Project Outline

The thesis is organized as follows, we will discuss the background and related work which has already been done in the context of S-LID in chapter 2. The Dataset preparation methods will be discussed in chapter 3. An in-depth examination of the methodology followed for building a baseline LID classifier with an emphasis on the typical

neural network topologies, training methods, and evaluation metrics are discussed in chapter 4. Additionally, this chapter introduces the idea of transfer learning as potential strategy for addressing the challenges of low-resource LID. Chapter 5 discusses the experiments and results which were conducted according to the methodologies discussed in the previous chapter. Conclusions and Future scope are discussed in the final chapter.

1.3 Contributions

This section outlines the work carried out in the thesis in pursuit of advancing language technology by understanding phonetic patterns from universal phoneme sequences. The major contr ibutions of the thesis is listed below:

- Built a Baseline classifier model purely based on phoneme sequences generated from the utterances in the FLEURS [9] Dataset.
- Adapted and investigated the model performance on zero-shot scenario.
- Determined the minimum duration of utterance required for fair few-shot performance.
- Analysed the limitations and potential solutions for the zero-shot case.

Chapter 2

Background & Related work

An analysis of the challenges of Spoken Language Identification and current research for mitigation of these challenges will be done in this chapter. Most of the conventional and contemporary LID system construction techniques are described in the first part. We will discuss the Universal Phone Recognizers (UPRs) which have been heavily used in the project and are the backbone of the developed LID system. We will also discuss the various feature engineering and classification techniques which are essential for successful development of the LID system.

2.1 Challenges of Spoken Language Identification

A number of factors make it difficult to reliably identify the language being said in an audio segment, rendering spoken language identification (SLI) a challenging task. First, it is challenging to distinguish between various languages based only on sound properties because phonetic and acoustic elements of languages are frequently shared. Second, subtle differences between languages can be introduced through dialects, accents, and regional variants within those languages.In addition, background noise and various acoustic circumstances in recordings can reduce the LID models' accuracy, particularly in real-world scenarios. Furthermore, code-switching (the switching between languages) and language mixing—the blending of many languages—are frequent features of spoken speech.

Humans on the other hand have been identified to be by far the best Language Identifiers according to Li et al. [23]. This is because human listeners characterize the languages using prominent phonetic, phonotactic, and prosodic cues which represent the various abstraction levels present in an audio signal of any language.

2.2 Cues for Spoken language Identification

Humans are able to efficiently identify languages based on different auditory cues present. Nearly all studies, including Muthusamy et al. [29], Zissman and Berkling [47], Li et al. [23], and Lee [21], grouped the cues into the categories shown in Figure 2.1.

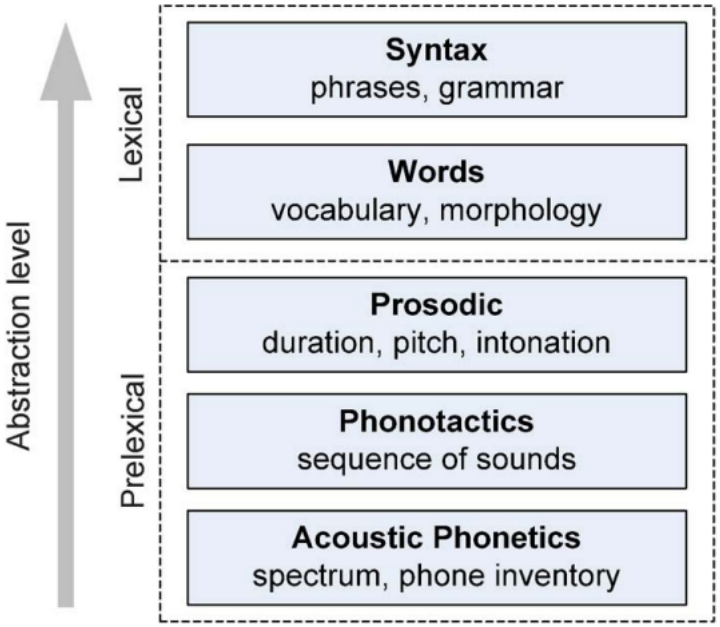


Figure 2.1: Perceptual cues at various levels that can be used for LID. [23]

2.2.1 Syntax & Words

The Lexical cues can be used for LID by building multiple Large vocabulary Continuous Speech Recognition (LVCSR) models for different languages, the intuition being that if a model knows to predict the words of an audio then it already knows the target language. This method is however not used for LID since it requires huge amount of transcribed data of each language. Hence, it does not make sense to use the Lexical features for the purpose of Low Resource languages which is the main concern of the Thesis.

2.2.2 Acoustic Phonetics

Traditional spoken language identification (LID) techniques use acoustic characteristics to distinguish between several languages. There are numerous ways to extract these acoustic features from the voice signal, including:

- Mel-frequency cepstral coefficients (MFCCs) [10]: These spectral features are frequently employed in LID and voice recognition. They are computed by first applying a mel filterbank on the logarithm of the voice signal's power spectrum.
- Coefficients of Linear Predictive Coding (LPC) [31]: LPC coefficients are a form of time-domain feature that are frequently utilized in LID and voice recognition . They are determined by fitting a linear predictive model to the speech input.
- Teager Energy Operator (TEO) [28]: TEO is a derivative feature that may be applied to derive details about the speech signal's dynamics. It is determined by multiplying the signal by the derivative of the signal.

After the audio features have been retrieved, a classifier can be trained to identify between various languages using the features. A Gaussian mixture model (GMM) [38] is the sort of classifier that was most frequently employed for LID. GMMs are a kind of probabilistic model that can be used to visualize how different languages' acoustic feature distributions vary. It has been demonstrated that traditional spoken LID techniques work well for a variety of languages. They need a lot of labeled data, though, and their training can be computationally expensive.

Deep learning techniques for LID have been more popular recently as they have been proved to perform better than conventional techniques. One specific approach involves the usage of X-Vectors which are extracted from the Deep neural (DNN) architectures such as Time Delay Neural Netowrks (TDNN) [41] and Convolutional Neural Networks (CNN) [20]. The extracted acoustic features are fed to one of these architectures followed by a bottleneck layer, pooling layer and a Linear layer to get the final X-Vectors. These X-vectors are then used to classify between different languages. This method has given state of the art performance on various evaluation metrics for LID. However, they require meticulous designing of the model architecture and can be quite computationally expensive to train.

2.2.3 Phonotactical Methods

The smallest separate unit of sound that can alter the meaning of a word in a language is called a phoneme. It is a fundamental idea in phonology, the study of the sounds of human speech, as well as linguistics. The fundamental units of spoken language, phonemes serve to distinguish between words and communicate meaning. Patterns in the sequences of phonemes can be different for different languages and by identifying these patterns we can intuitively predict the language.

There can be different rules for the arrangements of phonemes depending on the language, these rules are known as phonotactics. It has been established that the phonotactical rules contain more discriminative information about a language rather than the language-specific phonemes, i.e. the order of arrangement of the phonemes can be useful to identify the language [22].

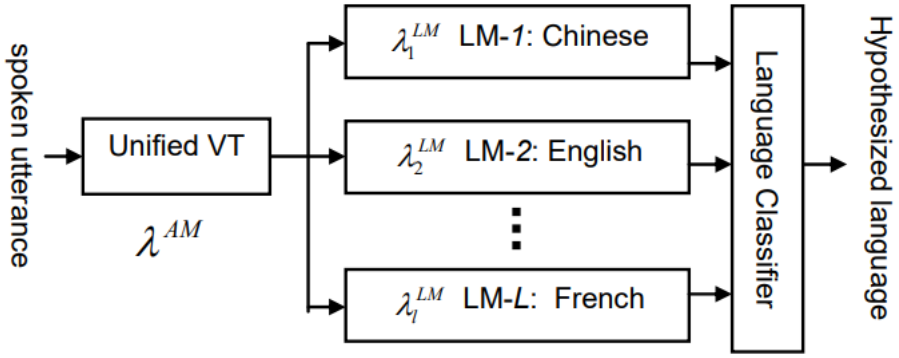


Figure 2.2: A unified Voice Tokenizer (VT) divides the component segments of the input voice feature frames into tokens by giving each segment an acoustic or phonetic identifier. A statistical language model(LM) here, L paralell Bag-of-Sounds (BoS) LMs that takes token sequences and extracts language-specific phonetic and phonotactic information, followed by BoS Classifier [22].

Figure 2.2 represents the Bag of Sounds (BoS) classifier [22] which was adapted from the traditional Parallel-Phone Recognition followed by Language Modelling (P-PRLM) architecture [46] which uses a unified Voice Tokenizer (VT) front-end, containing a unified phoneme set. A VT is a Speech Recognizer that breaks down spoken text into a series of tokens of a particular language. A unified VT can tokenize speech of multiple languages.

As we can see that although a unified VT is used, this method still requires training

of L different BoS Language models which individually score the same audio signal. Training multiple language models can be very time consuming and requires large amount of transcribed data, specifically labelled phoneme transcriptions which is very difficult to engineer.

We will use Universal Phoneme Recognizers (UPRs) to generate phoneme sequences directly from the audio signal and use these sequences to learn the language specific phonotactics which then be used for LID.

2.3 Universal Phone Recognizers

The LID developed in the thesis, depends heavily on the generated phoneme sequences as these are the main input features for language classification.

2.3.1 XLS-R Feature Extraction

XLS-R is a large language model (LLM) recently developed by Meta AI which was specifically designed for speech recognition and is based on wav2vec 2.0 [5]. XLS-R was trained on a massive dataset of 436,000 hours of speech from 128 different languages. It is the largest cross-lingual speech representation model to date.

Multilingual quantized speech units are produced by the XLSR model's shared quantization module over feature encoder representations. The embeddings of these speech units are subsequently used as targets by a Transformer that has been contrastively trained. The model learns the ability to transfer discrete tokens between languages, hence understanding the dependencies and similarities between them [8].

The XLS-R Model, which was pretrained on the CommonVoice [3],BABEL [14] and Multilingual LibriSpeech (MLS) [36] datasets. Specifically, the model was trained on 10 languages (*Spanish, French, Italian, Kyrgyz, Dutch, Russian, Swedish, Turkish, Tatar and Chinese*) plus English from the CommonVoice dataset with a total time of 1350h out of which 793h from the 10 languages and 557h of English. This XLS-R model was then fine-tuned on each language [8].

The Multilingual XLS-R Model performed exceptionally and obtained 13.6% Phone Error Rate (PER) on average , which is 49% less than the Monolingual (Trained only on English) XLS-R model [8]. This rapid improvement in PER makes it an ideal choice for generating phoneme sequences, even for Low Resource Languages.

We used this pre-trained XLS-R model to aid us in generating phoneme sequences

directly from the audio files. These generated phoneme sequences are the foundation of the research carried out in the thesis.

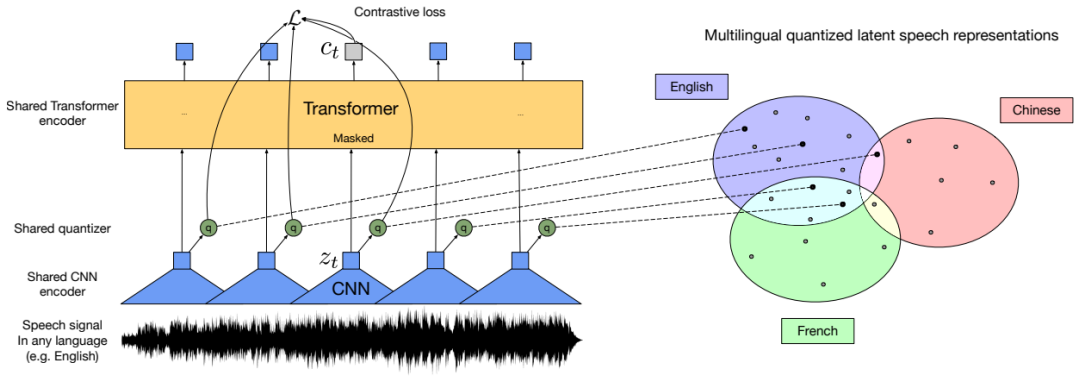


Figure 2.3: The XLSR Approach [8].

2.3.2 ALLOSAURUS

Allophones are various ways in which a language's phonemes can be pronounced that do not change the word's actual meaning. A good example is the "t" sound in the word "top", which is aspirated in English and is pronounced with a puff of air, as opposed to the "t" sound in the word "stop", which is unaspirated and is heard naturally. These two alternate English pronunciations of the "t" sound are allophones of the same phoneme because they do not change the meaning of the word.

We already know that the phoneme sequences differ depending on the language. Since most multilingual acoustic models assume that all training languages have the same union of phonemes, they only employ the existing phoneme transcriptions [25, 27, 39]. This can be problematic for unseen languages as phonemes may get misaligned with their respective phones, hence, reducing the model performance.

The ALLOSAURUS (ALLOphone System of AUtomatic Recognition for Universal Speech) method Li et al. [24] has demonstrated that it can increase phoneme recognition accuracy on unknown languages by 17%. This method will be helpful for phone recognition of low resource (LR) languages.

The ALLOSAURUS connects the phonemes that show up in the transcription of each language to a common limited phone set using an allophone layer that integrates phonological knowledge into the multilingual model. They used a typical ASR encoder to first determine the phone distribution, which the allophone layer then converts into the phoneme distribution for each language. This model was trained from scratch using only

the allophone lists provided by phoneticians and traditional phonemic transcriptions. The allophone layer was first initialized with the list of allophones, and then further optimized in a self-supervised way throughout the training phase [24].

2.3.3 Transphone Grapheme to Phoneme converter

Transphone is a Grapheme to Phoneme (G2P) conversion library introduced by Li et al. [26]. Given a piece of text and its corresponding language. Transphone can auomatically generate its respective phoneme sequences.

The G2P model used in the transphone was trained on the Wiktionary [11] dataset. Firstly, they trained numerous monolingual models for each of the High resourced languages (269 languages) in a supervised manner. To tackle the zero-shot case for unseen languages they mapped the unseen language to the k closest high resource languages by using phylogenetic tree (i.e. language family tree), this is because Nordhoff et al. [30] have already provided language family information for approximately 8000 languages.

After obtaining the k closest languages and their respective Monolingual models they then ensemble the predictions made by each of them to give the final output phoneme sequence. By using this technique they were able to reduce the Phoneme error rates to 39.8% on unseen languages.

The Transphone library can currently generate phoneme sequences for 7,456 languages. Although transphone generates phoneme sequences from text, it is essential for the LID since we will require phoneme sequences of the textual transcripts available in our data-set.

2.4 Feature Engineering Techniques

In this section we will discuss the various methods of representing the phoneme sequences. The classifiers tested in the thesis require the inputs to be numerical vectors hence the phoneme sequences need to be converted into numerical vectors. Additionally, changing the representation of the phoneme sequences helps in reducing the dimensionality and computationally efficient.

2.4.1 TF-IDF Vectors

Term Frequency-Inverse Document Frequency is referred to as TF-IDF. It is a numerical representation used in information retrieval and natural language processing to assess

the significance of a word (or phrase) within a group of documents. The TF-IDF considers a term's rarity across the entire document collection as well as its frequency in a particular document. It attempts to determine the relative weight of words in various manuscripts [16].

Term Frequency (TF) is a metric used to determine how frequently a term appears in a given document. It is determined by dividing the total number of terms in a document by the frequency at which a given term appears in that document. The intuition behind this is that the terms that appear more frequently in a document are probably significant phrases inside that document.

$$\text{TF} = \frac{\text{Number of times term appears in document}}{\text{Total number of terms in document}} \tag{2.1}$$

Inverse Document Frequency (IDF) of a term indicates the percentage of corpus documents that contain the term. terms that are specific to a small number of documents are given greater relevance ratings than terms that appear in all documents.

$$\text{IDF} = \log\left(\frac{\text{Number of documents in the corpus}}{\text{Number of documents in the corpus containing the term}}\right) \tag{2.2}$$

TF-IDF of a term is calculated by simply multiplying the TF and IDF values of the term.

$$\text{TF-IDF} = \text{TF} * \text{IDF} \tag{2.3}$$

The TF-IDF takes into account both the frequency of phonemes inside a sequence and the rarity of those phonemes over the whole dataset. This is useful for identifying phonemes that occur frequently and are significant in a sequence hence enable us to potentially use these phonemes to identify different languages. TF-IDF also captures the importance of certain terms in a document, this can be useful for LID since languages have a set of phonemes which are more frequently used than the others. Additionally, TF-IDF is immune to the variability of the length of sequences, eliminating the need of unnecessary padding to make the phone sequences of the same length. This is useful due to the fact that since we are dealing with phone sequences generated from an audio signal, sequence length variability is likely.

2.4.2 One-hot Encoding

One-hot Encoding is a technique in which a token is represented as a binary vector. Firstly, a set of all the unique tokens is generated from a corpus and assigned a numerical index. Secondly, a vector is assigned to each token where the corresponding index value

is 1 and the rest are 0s, the total length of the vector depends upon the total number of unique tokens found in the first step. This vector is known as the one-hot vector of the particular token. All the tokens have a unique one-hot vector assigned to them, due to this, each token becomes independent from the others.

One-hot vectors can be useful to represent individual phonemes as they preserve the sequential information of the phonemes, However, on the flip side, a large set of phonemes will have longer one-hot vectors and since we are dealing with phoneme sequences this would result in very high dimensional data and hence would require higher computational resources while training the LID Classifiers.

2.5 Methods of Classification

We will discuss the various methods of classification including the Statistical and Deep Learning Methods which were experimented when designing the LID model. Choice of an ideal classification technique is essential as it can significantly impact the performance, accuracy, and generalization of LID model.

2.5.1 Statistical Methods

2.5.1.1 Logistic Regression Classifier

The Logistic Regression (LR) classifier is one of the simplest statistical classifier in which a decision boundary is created to classify categorical data. LR is best suited for binary classification tasks, since we have multiple languages to classify, LR can be adapted to work with multiple classes using clever techniques such as One vs Rest (OvR) and Softmax.

In the OvR approach, a unique Binary LR model is trained for each class, where one class is used as the ”positive” class throughout training, while the other classes are categorized as the ”negative” class. Due to this, there are various binary classifiers that each differentiate one class from the others. The Softmax approach on the other hand involves training a single LR classifier that predicts the raw probabilities of each class. These probablities are then passed through a softmax function, to ensure that the sum of probabilities is one.

Although LR is a relatively naive model it is a good starting point for developing the LID classifier as it is easy to interpret and the training times of this model are relatively low, thanks to it’s simplicity. The LR model may suffer from poor performance as it

makes the assumption of the data being linearly separable hence it may not capture complex relationships in the data.

2.5.1.2 Support Vector Machines

Support Vector Machine (SVM) is a supervised Machine learning algorithm which supports both classification and regrssion tasks. SVM tries to fit a hyperplane that maximizes the distance between data points of distinct classes while best separating them. Support vectors are the nearest data points to the hyperplane.

SVMs can handle high dimensional data very efficiently as it makes use of kernel functions. With the help of these functions, the data is implicitly mapped to a higher-dimensional space where it may be possible to linearly separate it. This feature of SVMs enables it to capture complex dependencies in the data, which was the major drawback of LR classifiers. SVMs usually fit a hyperplane which can be high dimensional, making it less interpretable than the LR classifier. Dimensionality reduction techniques such as Principal component analysis and Linear Discriminant Analysis may help to visualise the fit.

2.5.2 Deep Learning Methods

In recent years, it has been proven that deep learning techniques are useful for spoken language identification (LID). These techniques can recognize intricate voice signal patterns that are difficult for traditional techniques to pick up. There are various architectures which can effectively identify patterns and dependencies in sequences of phonemes which may be indicative of the language itself.

2.5.2.1 Recurrent Neural Networks

A Recurrent Neural Network (RNN) is a type of neural network that can analyze time-series data as well as plain language and sequential data. Because they can monitor their internal state, RNNs are able to recall data from earlier time steps.

Through the use of multiple layers, a typical feed-forward neural network generates an output from the input. After the input has been processed through a recurrent layer and is preserved in an internal state, an RNN generates an output based on both the input and the internal state. The RNN can process sequential data and maintain context over time as a result. RNNs make advantage of the idea of sequential information. A

RNN is a neural network with a memory that influences future predictions based on data that is stored sequentially [2].

2.5.2.2 Long-Short Term Memory Networks

Long Short-Term Memory (LSTM) is a form of RNN designed to mitigate the drawbacks of conventional RNNs. Long-term dependencies are a difficulty for RNN that can be solved utilizing an LSTM-RNN network. LSTMs are highly suited for jobs that need sequential data, such as speech recognition and natural language processing, since they can better maintain information over lengthy time steps.

LSTM cells have a more intricate structure than conventional RNN cell. Each LSTM cell has a hidden state, an input gate, an output gate, and a forget gate. The LSTM can selectively read, write, or forget data from earlier time steps thanks to these gates. These gates protect or control the condition of the cell. The sigmoid function that makes up the LSTM network only has two possible outcomes: either it will pass all of the input data, or it won't. As a result, we may use a cell state to control the long-term dependencies that were a difficulty for RNNs [17].

Chapter 3

Data Preparation

Data preparation is a very important step for building a strong and robust classification model. The LID system heavily relies on this step. When data is prepared properly, it is accurate, consistent, and ready for use. Information regarding the data-set used and techniques to prepare the data will be discussed in this chapter.

3.1 About the Dataset

There are several publically available large-scale pretraining and evaluation datasets like Multilingual LibriSpeech [36], VoxPopuli [42], CoVoST-2 [43], CommonVoice [3] and BABEL [14].

Conneau et al. have recently released another publically available data-set, the Few-shot Learning Evaluation of Universal Representations of Speech (FLEURS) [9] data-set. They created FLEURS based on the FLoRes-101 [15] data-set which contains 3001 sentences extracted from the English Wikipedia. These sentences have been translated to 101 different languages by human translators hence the name FloRes-101. They collected three recordings of all the sentences by the native speakers of each of the languages.

Some key features of FLEURS which make it ideal for our LID System are:

- FLEURS contains n-way parallel speech and text in 102 languages.
- FLEURS provides high-quality, authentic human speech and transcripts in each language under strict quality control.
- FLEURS utilizes a bottom-up technique for gathering spoken utterances for

aligned segments, whereas the majority of other datasets use automatic segmentation and alignment for segments at the document level.

FLEURS as the name suggests is an evaluation data-set specifically designed for few shot learning which contains parallel speech and transcripts for 102 languages including low resource languages such as Asturian, Estonian, Kabuverdianu and Icelandic, hence we used this data-set for training and inference of our LID system.

3.2 Data Preprocessing

The FLEURS Dataset is publically available on the Hugging Face website [18]. The dataset is avalable by language and is already split into training, development and test sets. This eliminated the need to explicitly split the data. The audio utterances are in wav format which is the most common format used in Speech Recognition tasks. The transcripts and duration of each utterance is mapped by its respective ID.

FLEURS does not provide a master dataset containing utterances from all the 102 languages, Instead they provide the Audio and TSV files for each language separately. After extraction of all the Audio files for all the languages we had 102 folders for each language and the total size was 299GB.

The next step was to merge all the folders into one single master folder, However due to the large size of the data we simply could not merge them directly, So, we used the Kaldi Toolkit [34].

The files listed above were created easily by running a simple python script for each language folder. Kaldi has a built in script ”combine_data.sh” which automatically combines the above listed files in multiple project folders into a single project folder. This method of merging the data was very fast as merging of the text file is faster rather than the data files themselves.

The single project folder contained all these files and upon examining the line counts of the wav.scp file in the Train, Development and Test, we found that the there are a total of 27200, 34000, 78000 entries repectively, each entry represents a single utterance.

3.3 Generation of Phoneme Sequences

Generation of phoneme sequences was the most crucial step as the whole project revolves around phoneme sequences. After merging the data from all the languages, we

generated two sets phoneme sequences for each utterance, one from the audio file and the second from the respective transcripts. The procedure followed for generation of these sets of phoneme sequences is discussed below.

3.3.1 From the Audio

A Time Delay Neural Network (TDNN) [33] with 18 hidden layers each with 798 hidden units and 90 bottleneck layers was designed. The model was trained with LF-MMI [35] using the Kaldi toolkit and has 7.2M total parameters. It utilised 40 dimensional cepstral mean and variance normalised (CMVN) MFCC features as inputs [19].

The trained Universal Phone Recogniser (UPR) takes in the CMVN MFCC features of an utterance as inputs and generates the universal phoneme sequence for it. We created a simple shell script which gives these features to the model and maps the respective utterance to the phoneme sequences generated in a text file. Due to the sheer size of the data (299GB), it was not possible to run the script on our local machine, so we used the zamora server which has Quad 8 core CPUs, and 512GB of RAM. We will call these phoneme sequences as Decoded Phoneme Sequences (DPS) the following sections.

The DPS contain ‘sil’, which represent silence segments in the audio i.e. when the speaker takes a pause this phone gets generated. Silence segments are essential for breaking up audio streams into manageable chunks, which facilitates language identification. These segments affect feature extraction, which is important for identifying speech, quiet, and noise, acoustic cues that are unique to each language. They also aid in controlling background noise and improve LID accuracy by taking into account various patterns in various languages.

Despite the benefits of ‘sil’ phonemes we them from the DPS, since we will also be generating phoneme sequences from the transcripts which is text, we cannot output ‘sil’ as it is speaker dependant. The presence of ‘sil’ in DPS would only add noise in the DPS.

3.3.2 From the Transcripts

The generation of phoneme sequences from the transcripts is essential as these serve as the ground truth for a given utterance. There are a total of 384,060 total utterances. We used the Transphone [26] python library to automatically generate the phoneme

sequences of the utterances. The transphone library takes the text and lanugage as input and outputs a sequence of phonemes. These inputs were easily accessible thanks to the 'utt2lang' and 'text' files present in our Kaldi project, however, we noticed that the transphone library takes longer time to generate phoneme sequences for longer sentences i.e. more than 5 words. This was problematic as most of the transcripts are naturally longer than 5 words, and we have a lot of utterances.

To circumvent this issue, we wrote a python script which first groups all the utterances according to their language, then finds the set of unique words in each of the languages. After finding the unique words of each language, phoneme sequences were generated word by word for each language using the transphone library. The generated phoneme sequences were then mapped to the words in the transcripts hence, creating the phoneme sequence of the long sentences present in the transcripts. This method was faster because we generated the phoneme sequence word by word rather than the whole utterances. We will refer to these phoneme sequences as Transcript Phoneme Sequences in the subsequent sections.

3.4 Conversion of Generated Symbols

The Decoded Phoneme Sequences (DPS) contain phonemes which are in Extended Speech Assessment Methods Phonetic Alphabet (X-SAMPA) [44] format, whereas, the Transcript Phoneme Sequences (TPS) have phonemes which are in International Phonetic Alphabet (IPA) [44] format. Although both formats are used for phonetic representation, IPA has a significantly larger phoneme dictionary than X-SAMPA. Therefore for data consistency, both of these need to be in the same format.

We made use of the 'phones' [4] python library, and converted the DPS from X-SAMPA to IPA and also the TPS from IPA to X-SAMPA. Futhermore, we also converted both DPS and TPS into a third, 'ARPABET' [37] format. We incorporated all the converted phoneme sequences into our Dataset. All the conversions were done utterance by utterance.

3.5 Final Data-set Structure

The Final Dataset now contains 271798, 34452, 77810, rows in the Train, Development, and Test sets. The information about the columns in the data set is shown in Table 3.1

Column Name	Description
ID	The ID of the utterance
Language	The language of the utterance
DPS	The decoded phoneme sequences generated by the UPR in X-SAMPA format.
DPS_IPA	The decoded phoneme sequences generated by the UPR in IPA format.
DPS_ARPA	The decoded phoneme sequences generated by the UPR in ARPABET format.
TPS	The phoneme sequences of the transcripts generated by the transphone library in IPA format.
TPS_XSAM	The phoneme sequences of the transcripts generated by the transphone library in X-SAMPA format.
TPS_ARPA	The phoneme sequences of the transcripts generated by the transphone library in ARPABET format.
Text	This contains the transcripts themselves.
Duration	This contains the duration of each utterance in seconds.

Table 3.1: Table explaining the columns in the final data-set structure after processing

This was the final prepared data which was used in all the experiments conducted in Chapter 5.

Chapter 4

Methodology

This chapter discusses the various evaluation metrics and how they can be used for our specific task. We also discuss the rationale behind the methodology adopted for building a baseline classification model for LID and the steps followed for adapting the model for Low Resource LID.

4.1 Evaluation Metrics

Language Classification is the main focus of the thesis hence, we used traditional evaluation metrics such as Accuracy, Precision and Recall. [7] and F1-score. The percentage of correctly classified data instances over all data instances is known as accuracy. Accuracy is susceptible to class imbalance and can be misleading in the interpretation of results. For each class, Precision is the ratio of True Positives and the sum of True positive and False Positive, it helps us to understand the model's ability to correctly classify the languages. For each class, Recall is the proportion of correctly classified examples over all the examples in the same class. F1 Score is the Harmonic Mean of the Precision and Recall values and it helps to identify the trade-offs between precision and recall.

Although we chose to evaluate our LID Model on all the 4 metrics, we gave more importance to the precision,recall and F1 scores, since we observed significant class imbalance among the 102 languages used to train the LID model. These metrics were also useful for identifying the single class performance as we were concerned with the performance on Low Resource Languages.

4.2 Building the Baseline Model

This section explains the rationale behind the experiments carried out for training the Baseline LID Model. Please refer to Chapter 5 for specific experimental setups, results and discussions

4.2.1 Ideal Sequence Length

We did not observe any utterance with a duration longer than 20 seconds, So, we experimented by training multiple Logistic Regression (LR) classifiers each trained on phoneme sequences of different lengths. We simply concatenated the phone sequences of two or more consecutive utterances of the same language to lengthen phone sequences to a particular duration. All the models were evaluated under the same metrics.

We used LR clsssifiers because they are relatively faster to train. Also it is safe to assume that other model architectures would also follow the same trends of the LR model since we are manipulating the data itself. It was essential to determine the ideal sequence length at the very beginning as this was the only experiment involving data manipulation.

The effects of various lengths of phoneme sequences on model performance is essential to analyse as this will help decide the minimum utterance duration the model would need for effective LID.

4.2.2 Model Selection

This was the very first step towards the development of the LID system after fixing the sequence length, and various classifiers such as Logistic regression, Support vector machines, RNN and LSTM were trained and evaluated. We created feature representations using the TF-IDF vectorizer from the scikit-learn [32] and used the default ngram range. We chose the best performing model for the subsequent experiments.

Model selection is a crucial step for building a LID system because it has a direct impact on the system's functionality, its generalization potential, and suitability for LID. Our baseline LID system's accuracy, effectiveness, and capacity to deal with various languages and situations can all be dramatically impacted by selecting the appropriate classifier.

4.2.3 Choice of N-Gram value of TF-IDF features

We used the TF-IDF vectorizer provided by the scikit-learn [32] python library to represent the phoneme sequences as numerical vectors. The TF-IDF vectorizer has a parameter 'ngram_range' which controls the number of ngrams extracted while creating the TF-IDF feature vectors for the phonemes. We investigated the performance of the chosen model for different ngram orders.

Choice of an ideal value of ngram range is essential as a higher value would naturally result in sparse feature vectors which would thus require more computational resources. An ideal value will ensure that the trade-off between computational resource required and model performance in minimal.

4.2.4 Choice of Format of the Phone Sequences

The Dataset contains phoneme sequences in X-SAMPA, IPA and ARPABET formats. We trained three separate models on phone sequences in each of the three formats and evaluated their performance. We also calculated the phoneme error rates (PER) for DPS and TPS in different formats to help us get a better understanding behind the performance differences of the classifiers.

This experiment enabled us to understand which format best captures the dependencies and patterns in the phoneme sequences which is necessary for effective LID.

4.2.5 Effect of Number of Languages

We also investigated the effect of Number of Languages on the model performance. We trained independent models on different number of languages by simply varying the total the number of languages and their respective utterances.

This experiment has less significance with regards to Low Resource LID, Rather, it serves as a verification that the LID classifier is working correctly as a model trained to classify lesser languages should perform better than the others. This verification is essential as we want to be sure that the model is consistent and can produce reliable language classifications.

4.3 Adapting the Baseline for Low Resource Languages

This section discusses the various experiments that were carried out for adapting the Baseline LID model to allow the classification of Low resource languages. For full experimental setups, results and discussions please refer to Chapter 5.

4.3.1 Transcript knowledge

All the experiments carried out previously, were purely on the Decoded Phoneme Sequences (DPS). However, training the model solely on DPS is not enough as we want to explore the Zero-shot scenario in which the model would never see the DPS of Low resource languages. Therefore, we need to incorporate Transphone Phoneme Sequences (TPS) while training the model such that it observes the Low resource classes even if we do not give DPS of the same.

We attempted to analyse the effects of incorporating transcript knowledge along with the DPS on model performance and compared it to the baseline model which was trained solely on DPS of all the classes.

4.3.2 Zero-shot Setting

The Decoded Phone Sequences (DPS) were generated from the audio itself, hence, they are treated as **speech** input to the model, while the Transphone Phone Sequences (TPS) were generated from the transcripts.

We treat the DPS as a direct source of audio data, whereas the TPS was not treated as audio data, since it has been generated from the transcripts directly. When the model is trained only on TPS of some languages, it means that it has never seen the spoken data of the same, thus these languages can be treated as unseen.

For example, if there are 5 languages and the model is trained on DPS+TPS of 3 languages and only TPS of the remaining 2. Here, the model can classify 5 languages but it has only seen the transcripts of the 2 languages and no audio data. This way of training the model simulates the zero-shot scenario for low-resource languages. We selected some languages to be treated as unseen i.e. low resource and the rest as well resourced languages.

We conducted an experiment in which we trained the baseline model on both DPS and TPS of the well-resourced languages and only the TPS of unseen languages. We ran inference on the unseen languages by only providing their **DPS** i.e. speech data

from the test set and recorded the model performance.

This experiment was perhaps the most important out of all the experiments done until now, as it helped us answer the fundamental question, whether zero-shot LID using phoneme sequences is even possible or not.

4.3.3 Effect of Number of Unseen Languages

We conducted another experiment under the same zero-shot setting but this time we trained multiple separate models by changing the number of unseen languages and observed their performance.

This experiment was particularly helpful, since it allowed us to determine the ideal number of unseen languages for which we may observe the best inference performance.

4.3.4 Extending to the Few-shot setting

We extended the zero-shot setting to serve as few-shot setting by incorporating some number (not all) DPS for the unseen languages while training the models. The number of DPS added was decided by the duration of the DPS themselves, also, the same duration of DPS were added for all the unseen languages.

We carried out another experiment in which we trained and evaluated multiple models under the few-shot setting, we varied the number of DPS added based on different durations in minutes, and observed the model performance.

Few-shot experiments were useful for understanding the model performance on low resource languages as soon as it sees a small amount data of them. This also aided us in determining the minimum duration of spoken data the model needs to observe for successful LID of low resource languages.

Chapter 5

Experiments, Results & Discussion

This chapter discusses the Experiments carried out according to the defined Methodology in Chapter 4. We list the hypothesis, followed by the experimental setup, results and finally, discuss our findings of each experiment.

5.1 Building the Baseline Model

5.1.1 Ideal Sequence Length

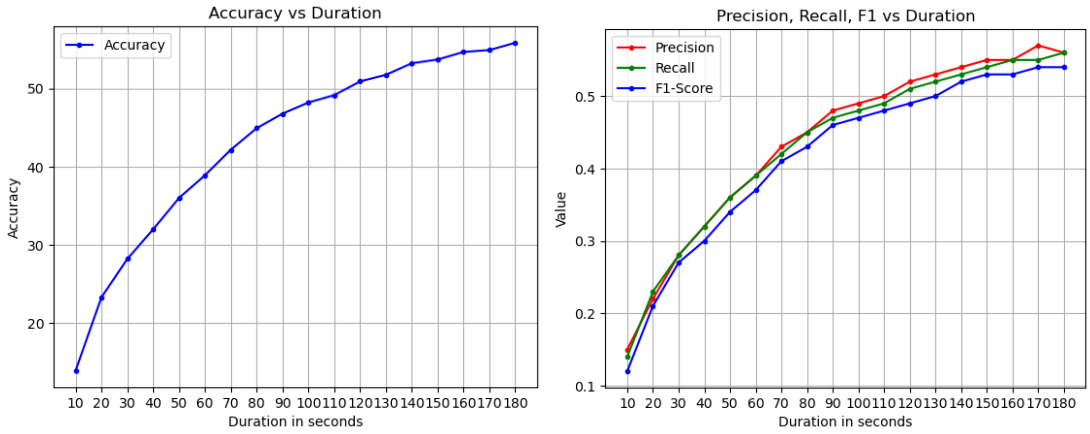
5.1.1.1 Experimental Setup and Hypothesis

We explored the effects of varying the minimum duration of sequence lengths (ranging from 10 seconds to 180 seconds of DPS) on model performance. We trained multiple Logistic Regression(LR) classifiers independently for this experiment since they are relatively easier to interpret and faster to train, also this experiments requires the training of multiple classifiers i.e. 18 classifiers each trained on DPS of the respective value of minimum duration.With the help of a simple python script we concatenated consecutive DPS of the same language class, according to the minimum duration needed.

We hypothesized that as we increase the minimum duration of DPS, the model should start performing better as it would see longer phoneme sequences for a particular language class.

5.1.1.2 Results & Discussion

The plots of Accuracy vs Duration is shown in Figure 5.1a and Precision,Recall,F1 vs Duration is shown in Figure 5.1b.



(a) (b)

Figure 5.1: (a) Graph of Accuracy vs Duration, (b) Graph of precision, recall and F1-Score vs Duration

From the results we observe that the classification performance increases as the minimum duration of DPS increases, confirming our hypothesis. We observed highest performance when the model sees input DPS lengthened to 180 seconds (3 minutes). However, it would not be ideal to use DPS of 180 seconds as the model will not be able to correctly classify utterances which are significantly less than 3 minutes because it has only seen data which has a duration of 3 minutes. Therefore, we decided to fix the minimum duration to 60 seconds, as the performance between 60 and 180 second model difference is not that great.

5.1.2 Model Selection

5.1.2.1 Experimental Setup and Hypothesis

We experimented with different types of classifiers namely, Logistic regression (LR) , Support vector Machine (SVM), Recurrent Neural Network (RNN), and Long Short Term Memory (LSTM). We trained each of the classifiers just on the DPS of all the 102 languages in our dataset.

The LR and SVM classifiers were trained using the scikit-learn [32] python Library, where as we used the TensorFlow [1] library. Both RNN and LSTM networks had 5 layers each containing 256 units, followed by a feed forward layer with a softmax activation function and 102 units, since we had 102 total languages to predict. Additionally, we trained the RNN and LSTM models using early stopping by monitoring the

validation loss, to prevent overfitting. All the trained classifiers were evaluated on the test set so the results can be compared directly.

We hypothesized that the RNN and LSTM models would perform better than the LR and SVM classifiers since they are particularly good at capturing sequential information in the data. LSTM should perform slightly better than the RNN model since they overcome some subtle drawbacks of the RNN model, for more information please refer to Section 2.5.2.

5.1.2.2 Results & Discussion

The Performance results of all the four classifiers is shown in Table 5.1

Model	Accuracy %	Precision	Recall	F1-Score
LR	38.889	0.393	0.388	0.374
SVM	38.874	0.400	0.388	0.380
RNN	41.950	0.428	0.419	0.411
LSTM	43.717	0.465	0.437	0.428

Table 5.1: Performance results of LR, SVM, RNN and LSTM classifiers upon Accuracy, Precision, Recall and F1-Score

Upon examining the results we can safely conclude that the LSTM model out performs the other classifiers, corroborating our hypothesis. Also we see that the Deep learning based classifiers i.e RNN and LSTM perform better than the statistical classifiers (LR and SVM). We selected the LSTM model as our designated classifier for LID in the subsequent experiments.

5.1.3 Choice of N-Gram value of TF-IDF features

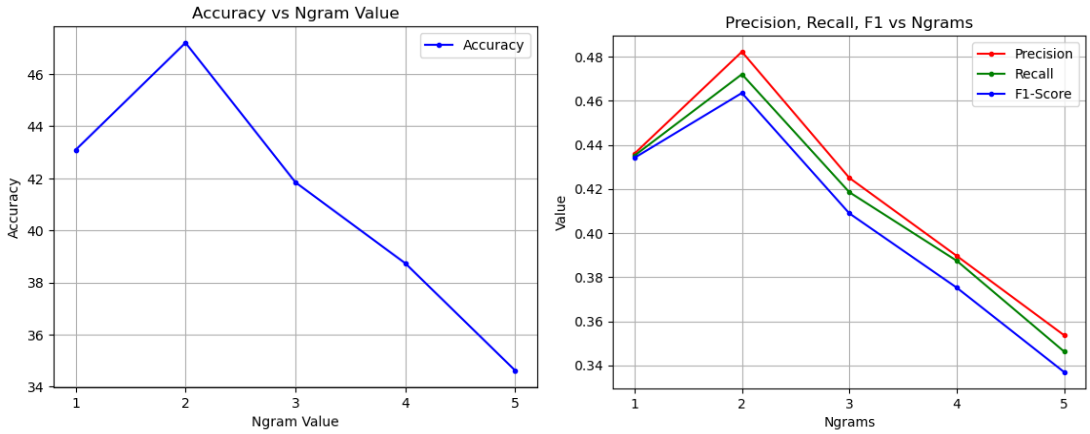
5.1.3.1 Experimental Setup and Hypothesis

All the experiments till now were conducted by representing the DPS as TF-IDF vectors but we used the default value of *ngram_range*. Hence, we experimented with various values of ngram values ranging from 1-5 to study the performance variations, we trained five separate LSTM classifiers for different ngram_range values, and analysed the performance characteristics of the models. We also observed the number of trainable parameters and the training times.

The ngram_range value determines how many ngrams the TF-IDF vectorizer will consider while creating the feature matrices for representing the phonemes. We anticipated the classifier with higher ngram value would have significantly more trainable parameters and take longer time to train. We also expected the the higher order ngram model to perform well compared to the other models.

5.1.3.2 Results & Discussion

Figure 5.2 depicts the evaluation results of the different LSTM classifiers trained on different Ngram values used by the TF-IDF vectorizer for representing the phonemes in the DPS.



(a) (b)

Figure 5.2: (a) Graph of Accuracy vs Ngrams, (b) Graph of precision, recall and F1-Score vs Ngrams.

Ngram Value	No. of Params. (in Million)	Training Time
1	6.15M	∼10 Mins
2	7.97M	∼19 Mins
3	34.48M	∼34 Mins
4	199.66M	∼3 Hours
5	708.84M	∼12 Hours

Table 5.2: Number of parameters and Training times for the LSTM Classifiers for different Ngram values

From Table 5.2, it is clear that as we increase the Ngram value the number of model

parameters increases, this was already expected as higher order ngram feature vectors are be sparse, as a result we observe the model with 5-gram feature vectors had 708.84 Million parameters whereas there were only 6.15 Million parameters.

Furthermore, from 5.2 we can see that the performance of the LSTM classifier increases as we increase the order of ngrams, but the performance decreases as we increase the order of ngram beyond 2. This is due to the fact that the number of parameters increases exponentially as seen from Table 5.2 and the model starts overfitting the data. Following the results of this experiment we used bigram (ngram order 2) feature vectors in the subsequent experiments.

5.1.4 Choice of Format of the Phone Sequences

5.1.4.1 Experimental Setup & Hypothesis

All the experiments done till now were on the DPS in X-SAMPA format, we trained three separate LSTM classifiers each trained on the DPS in X-SAMPA, IPA and ARPABET format. X-SAMPA has a relatively smaller phoneme dictionary than IPA and not all IPA symbols are present in the X-SAMPA dictionary.

ARPABET is a format which was created mainly for English, so the DPS in ARPABET format presumably may have many overlapping phonemes across different languages, However, their sequences might be more unique when compared to the DPS in X-SAMPA and IPA symbols. We might be able to see better model performance when trained on DPS in the ARPABET format.

5.1.4.2 Results & Discussion

From Table 5.3, we can see that the model performance degrades drastically when trained on DPS in the IPA format, the model is only 2.7% accurate. Whereas, we see a 7.8% increase in the model performance when trained on DPS in ARPABET format.

Also from Table 5.4 we see that the DPS and TPS coverted in the IPA format have the highest PER, whereas, the ARPABET has the lowest PER. This is because the IPA format has a significantly larger phoneme dictionary. On the other hand, ARPABET has a small phoneme dictionary.

So the conversion into ARPABET format may have created unique sequence language specific patterns in the DPS. This result was strong enough to convince us to run our further experiments on the DPS and TPS in ARPABET format.

Format	Accuracy %	Precision	Recall	F1-Score
X-SAMPA	46.934	0.473	0.469	0.458
IPA	2.712	0.020	0.027	0.016
ARPABET	54.772	0.556	0.547	0.538

Table 5.3: Model Performance Evaluation when trained using DPS of X-SAMPA, IPA and ARPABET phonetic alphabet formats.

Format	PER %
X-SAMPA	48.30
IPA	56.42
ARPABET	38.59

Table 5.4: Phoneme Error Rates (PER) between the DPS and TPS when converted into the same format.

5.1.5 Effects of Number of Languages

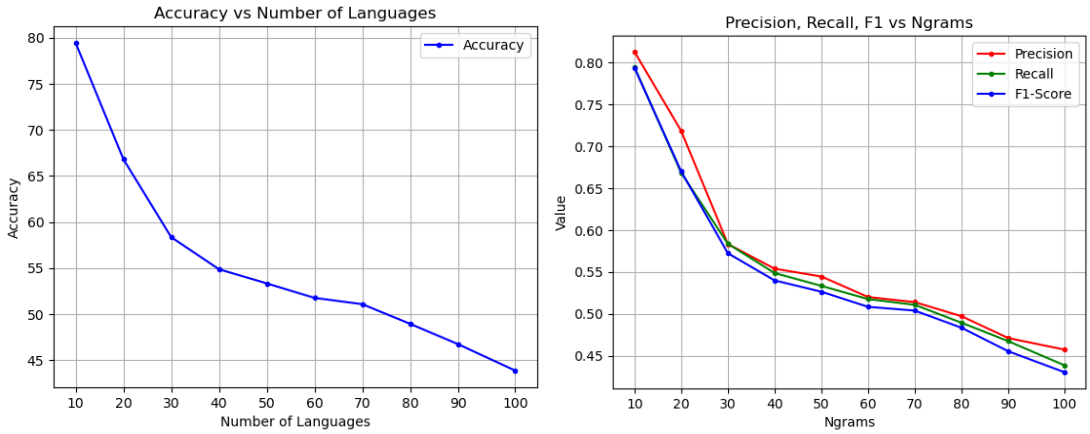
5.1.5.1 Experimental Setup & Hypothesis

We trained multiple LSTM classifiers where we gave both DPS and TPS for training and observed the model performance as we incrementally increased the number of output classes from 10 to 102 with a step size of 10.

Although it is known that models which have fewer output classes perform better, the results of this experiments would verify that the model is performing correctly, and is able to effectively learn meaningful patterns in the DPS.

5.1.5.2 Results & Discussion

The results from Figure 5.3 confirm that the model performance improves as we decrease the number of languages to classify. The model has an accuracy of around 80% when it only has 10 classes which decreases to around 45% for all the 102 languages. This result verifies that the model is performing correctly and is effective at discovering useful patterns in the DPS and TPS.



(a) (b)

Figure 5.3: (a) Graph of Accuracy vs Number of Languages, (b) Graph of precision, recall and F1-Score vs Number of Languages.

5.2 Low Resource Experiments

5.2.1 Transcript knowledge

5.2.1.1 Experimental Setup & Hypothesis

This experiment was mainly conducted to see the impact of incorporating the TPS along with the DPS when training the model. To incorporate TPS in the DPS we simply trained the LSTM model on two copies of all the utterances, one had the DPS as before and the other had TPS. Essentialy, we doubled the training data such that the model saw both DPS and TPS of the same utterance.

This experiment was mandatory to run as we need to adapt the model to be able to predict Low resource languages. Although we added the TPS while training the model was evaluated only on DPS in the test set because we are concerned about spoken LID and evaluating on TPS would mean evaluating the transcripts.

Upon seeing the TPS along with the DPS of an utterance, Theoretically,the model should be able to learn patterns in between the TPS and DPS themselves, thereby be able to improve its classification performance due to the added transcript knowledge. If this hypothesis turns out to be true, then it means the model can be able to predict the Low Resource languages for which little or no DPS would be seen.

Format	Accuracy %	Precision	Recall	F1-Score
DPS only	54.772	0.556	0.547	0.538
DPS + TPS	69.492	0.715	0.694	0.698

Table 5.5: Model Performance Evaluation of the LSTM classifier when trained only on Audio (DPS only) and Audio plus transcripts (DPS + TPS).

5.2.1.2 Results & Discussion

The results observed in Table 5.5 are significant, as we see that the addition of TPS increased the model accuracy from 54.7% to 69.4%. Furthermore it is proof that the model is able to learn some dependencies between the DPS and the TPS, hence, the model may be able to generalize in the zero-shot scenario.

5.2.2 Zero-shot Learning

5.2.2.1 Experimental Setup & Hypothesis

We trained the LSTM classifier in a special way for this experiment, we selected 20 languages out of the 102 languages to be treated as Low Resource (LR) languages. The languages were sorted in ascending order based on the total duration of utterances and the first 20 languages were selected as this ensured that the selected languages were actually low resourced. However, English was also included in the LR languages because we plainly selected the first 20 languages in the sorted according to the total duration of all the utterances combined.

We trained the model on the DPS and TPS of the remaining 82 languages and just the TPS for the 20 LR languages. After training the model we removed the top feed-forward layer of the LSTM network and fine-tuned it on the TPS of LR languages in the train-set. We ran inference on the DPS of the LR languages in test set to simulate the zero-shot setting described in Section 4.3.2.

The addition of transcript knowledge increased the model performance, so we speculate that the model may be able to identify LR languages based on their DPS as it may have learnt the inter-dependencies between the TPS and DPS of the seen languages.

Model	Accuracy %	Precision	Recall	F1-Score
Fine tuned LSTM	17.977	0.191	0.179	0.130

Table 5.6: Model performance on various evaluation metrics after fine-tuning on the low resource languages

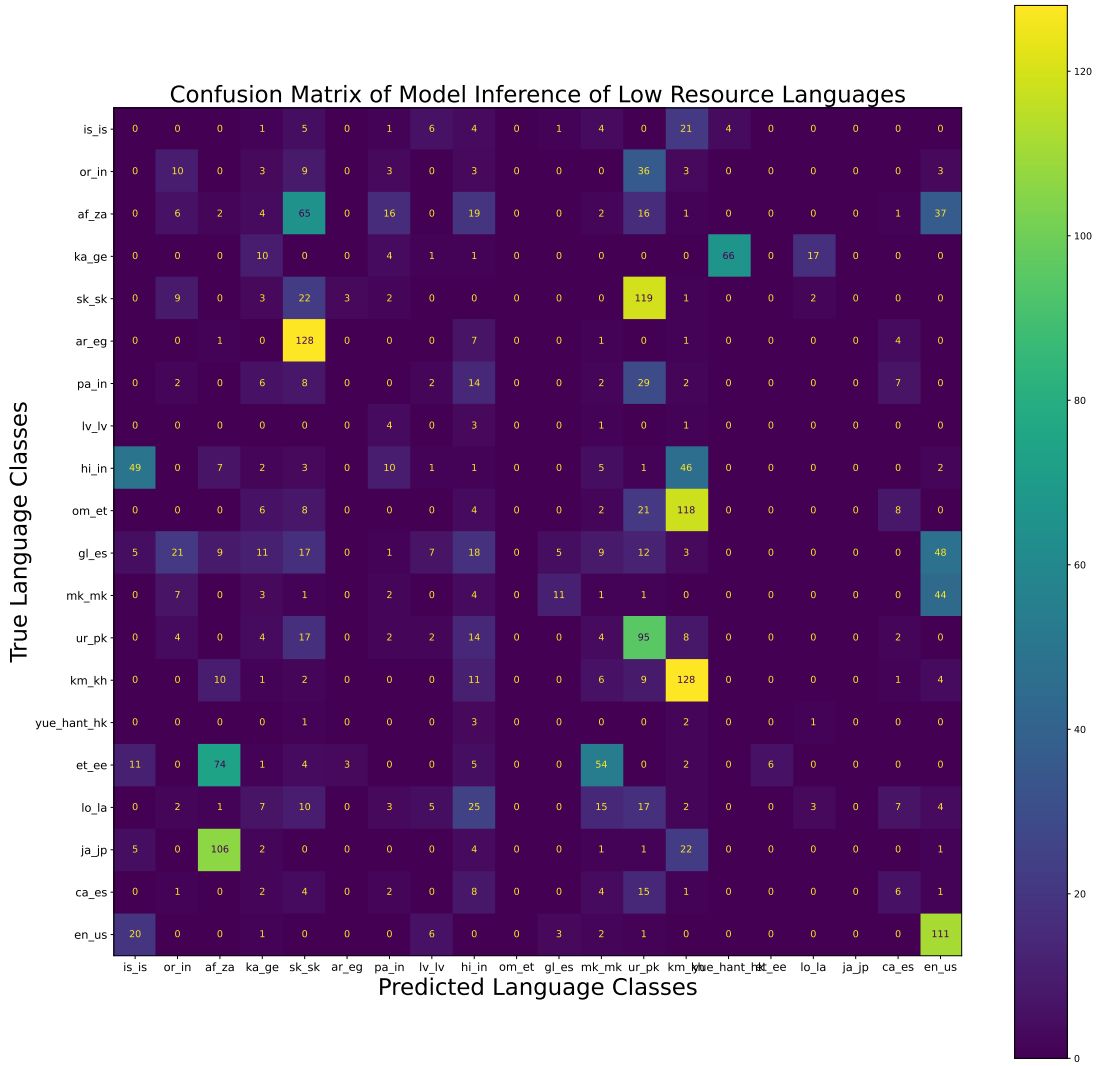


Figure 5.4: Raw Confusion matrix of the Zero-shot LID Model for 20 LR languages.

5.2.2.2 Results & Discussion

Table 5.6 depicts the Accuracy, Precision, Recall and F1-score when fine-tuned on the TPS of the low resource languages. Although the performance scores are pretty low, they are significantly higher than the chance level which in this case would be 5%. Hence, we concluded that the model is able to learn the inherent patterns in the TPS of

the low resource languages and generalize over the DPS of the unseen languages in the test set.

The confusion matrix shown in Figure 5.4, Not surprisingly, we observe a huge overlap for English. This was due to the fact that the phoneme sequences were converted in ARPABET format. We observed a rather scattered confusion matrix for this model because of the poor classification performance of the model.

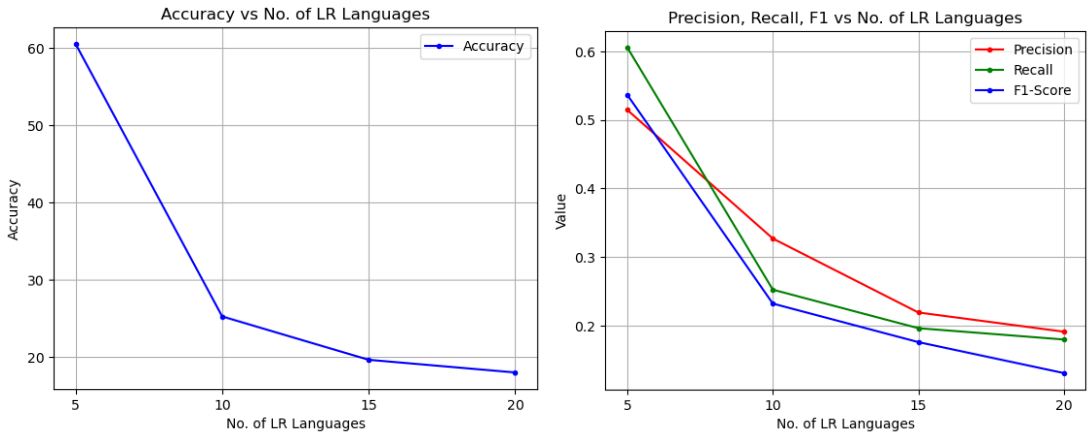
5.2.3 Effect of number of Unseen languages

5.2.3.1 Experimental Setup & Hypothesis

The Experimental setup for this experiment was the same as the previous experiment, only differnece was that we varied the Number of LR languages from 20 to 5 in steps of 5, i.e we tested the Model performance for 20,15,10 and 5 LR languages.

We hypothesized that as we decrease the number of LR languages the model should perform better, mainly because it has fewer classes to classify and also because it sees DPS of more languages so, it is able to learn more complex and intricate patterns which help in better generalization over LR languages.

5.2.3.2 Results & Discussion



(a) (b)

Figure 5.5: (a) Graph of Accuracy vs Number of Unseen Languages, (b) Graph of Precision, Recall and F1-Score vs Number of Unseen Languages.

Results observed from Table 5.5, conform our hypothesis, as we saw an improvement in the classification performance for less number of unseen languages. This is because

firstly, the model has lesser classes for classification. Secondly, the model observes more DPS of seen languages it is able identify the interdependencies in the DPS and TPS of the seen languages and generalize over the unseen languages.

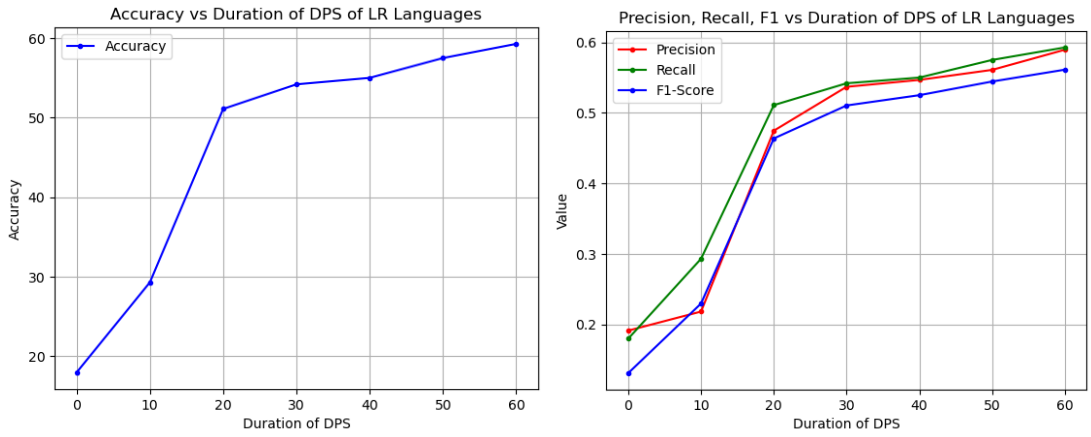
5.2.4 Few-shot Learning

5.2.4.1 Experimental Setup & Hypothesis

To simulate the few-shot scenario we incorporated the DPS of the LR languages along with the TPS. Specifically, we fine-tuned six separate models on 10, 20, 30, 40, 50, 60 minutes of DPS for the LR languages and ran inference on these.

For this experiment we kept the number of LR languages constant i.e. 20. Our hypothesis was that the inference performance would increase as we add more DPS information about the LR languages.

5.2.4.2 Results & Discussion



(a) (b)

Figure 5.6: Model performance when fine-tuned with varying durations of DPS (in minutes) knowledge of unseen LR languages (20 unseen languages). (a) Graph of Accuracy vs Duration of DPS of LR Languages, (b) Graph of Precision, Recall and F1-Score vs Duration of DPS of LR Languages.

The results seen from Figure 5.6 are consistent with our hypothesis and prove that incorporating DPS information in the Unseen languages increases the model performance. This means that our model can comfortably extend to the few-shot training scenarios.

Chapter 6

Conclusion

6.1 Results Summary

The research carried out in this thesis has shed light upon some important discoveries and ideas that advance our knowledge of this complex and important field of language technology. This study has clarified the advantages and disadvantages of employing Universal Phoneme Recognizers (UPRs) to identify languages with scarce linguistic resources.

Through the experiments that have been carried out while building the baseline model we identified some key data augmentation strategies such as varying the sequence length, changing the number of language classes and conversion of symbol formats, which have been proven to be beneficial for improving the performance of our LID system. The strengths and weaknesses of the proposed methods have been shown by thorough performance evaluation on various model designs. The effectiveness of the models has been evaluated using metrics including accuracy, precision, recall, and F1-score.

The low resource experiments that have been performed in our research shows that, although the classification performance on low resource languages might not be phenomenal, it is still possible. Furthermore, The experimental outcomes highlighted Few-Shot LID's potential to overcome linguistic limitations by demonstrating encouraging classification performance with minimal training data. The ability of the models to recognize common phonetic patterns across linguistic boundaries revealed the fundamental similarities in human speech.

In summary, the use of UPRs has emerged as a promising method for overcoming the difficulties brought on by low-resource scenarios. We investigated deep learn-

ing architectures that supported phonetic universality because we believed that these recognizers could serve as linguistic bridges across many languages. These models demonstrated their potential to cross linguistic barriers and offer a ray of hope for endangered languages by utilizing shared phonetic elements.

6.2 Future Scope

6.2.1 Alignment of Phoneme sequences

Fundamentally, the task of LID based purely on phoneme sequences, relies on finding unique patterns for each language. Errors in phoneme sequences generated from the audio as well as the transcripts will propagate while building a LID classifier as it may learn incorrect patterns and associations between the phoneme sequence and the language.

In our thesis we used two different UPRs, one for audio and the other for the transcripts, both generated phoneme sequences which were yet, in different formats and had to be aligned to a single format before they could be used. UPRs themselves are trained models which have their own error rates, these were bound to be propagated in the LID system.

We believe that phonemes are exceptional at representing phonotactical information, it would be interesting to see a UPR which can generate phoneme sequences from audio which closely align with the transcripts. Existing UPRs can also be improved, perhaps by training a machine translation model which aligns the phone sequences of audio and transcripts.

6.2.2 Deeper Network Architectures

Due to time constraints and limited computational resources we could not explore deeper and more sophisticated network architectures such as Transformer Encoder Decoder [40] networks BERT [12] and GPT-3 [6]. These networks are the current state of the art for sequence classification tasks and have shown phenomenal capabilities. These models were out of scope for this thesis mainly due to their sheer size and computational demands necessary for training.

6.2.3 Different Feature Representation Methods

We made use of TF-IDF feature vectors as they are easy and fast to construct. TF-IDF vectors preserve information of the frequency of the phonemes and lose the sequential information in the process. It would be interesting to see the performance variations of the LID classifier when different feature representation techniques are used. Due to computational limitations we were not able to test the performance of the model using one-hot feature vectors. Also it would be intriguing to see the performance when we use different feature representation methods like, Dynamic Time Warping and word embeddings to represent the phonemes.

6.2.4 Incorporating Acoustic Phonetics

This thesis focussed on building a LID classifier purely based on phoneme sequences. phoneme sequences were genereated by UPRs which are not 100% accurate themselves and we saw these errors propagate in our LID classifier.

We believe the LID has the potential to become more robust and accurate when Acoustic phonetics is used along with the phoneme sequences. Pitch, intensity, duration, and formants are only a few examples of the physical characteristics of speech sounds that are taken into account by acoustic phonetics. This additional physical characteristics may further make the LID classifier more accurate and reliable as it would be able to generalize over different speech variations and it would force the model not to solely rely on the phoneme sequences.

Bibliography

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Aditya Amberkar, Parikshit Awasarmol, Gaurav Deshmukh, and Piyush Dave. Speech recognition using recurrent neural networks. In *2018 international conference on current trends towards converging technologies (ICCTCT)*, pages 1–4. IEEE, 2018.

[3] Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M Tyers, and Gregor Weber. Common voice: A massively-multilingual speech corpus. *arXiv preprint arXiv:1912.06670*, 2019.

[4] Author(s) or Username. Phones minixc. https://github.com/MiniXC/phones, 2023.

[5] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell,

et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[7] Michael Buckland and Fredric Gey. The relationship between recall and precision. *Journal of the American society for information science*, 45(1):12–19, 1994.

[8] Alexis Conneau, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed, and Michael Auli. Unsupervised cross-lingual representation learning for speech recognition. *arXiv preprint arXiv:2006.13979*, 2020.

[9] Alexis Conneau, Min Ma, Simran Khanuja, Yu Zhang, Vera Axelrod, Siddharth Dalmia, Jason Riesa, Clara Rivera, and Ankur Bapna. Fleurs: Few-shot learning evaluation of universal representations of speech. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 798–805. IEEE, 2023.

[10] Steven Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE transactions on acoustics, speech, and signal processing*, 28(4):357–366, 1980.

[11] Aliya Deri and Kevin Knight. Grapheme-to-phoneme models for (almost) any language. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 399–408, 2016.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[13] David M. Eberhard, Gary F. Simons, and Charles D. Fennig. *Ethnologue: Languages of the World*. SIL International, Dallas, 23 edition, 2020.

[14] Mark JF Gales, Kate M Knill, Anton Ragni, and Shakti P Rath. Speech recognition and keyword spotting for low-resource languages: Babel project research at cued. In *Fourth International workshop on spoken language technologies for under-resourced languages (SLTU-2014)*, pages 16–23. International Speech Communication Association (ISCA), 2014.

[15] Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and

Angela Fan. The flores-101 evaluation benchmark for low-resource and multilingual machine translation. *Transactions of the Association for Computational Linguistics*, 10:522–538, 2022.

[16] Lukáš Havrlant and Vladik Kreinovich. A simple probabilistic explanation of term frequency-inverse document frequency (tf-idf) heuristic (and variations motivated by this explanation). *International Journal of General Systems*, 46(1):27–36, 2017.

[17] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[18] Hugging Face. Hugging Face. https://huggingface.co/datasets/google/fleurs, Year Accessed.

[19] Ondrej Klejch, Electra Wallington, and Peter Bell. Deciphering speech: a zero-resource approach to cross-lingual transfer in asr. *arXiv preprint arXiv:2111.06799*, 2021.

[20] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[21] Chin-Hui Lee. Principles of spoken language recognition. *Springer Handbook of Speech Processing*, pages 785–796, 2008.

[22] Haizhou Li and Bin Ma. A phonotactic language model for spoken language identification. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 515–522, 2005.

[23] Haizhou Li, Bin Ma, and Kong Aik Lee. Spoken language recognition: from fundamentals to practice. *Proceedings of the IEEE*, 101(5):1136–1159, 2013.

[24] Xinjian Li, Siddharth Dalmia, Juncheng Li, Matthew Lee, Patrick Littell, Jiali Yao, Antonios Anastasopoulos, David R Mortensen, Graham Neubig, Alan W Black, et al. Universal phone recognition with a multilingual allophone system. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8249–8253. IEEE, 2020.

[25] Xinjian Li, Siddharth Dalmia, David Mortensen, Juncheng Li, Alan Black, and Florian Metze. Towards zero-shot learning for automatic phonemic transcription. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8261–8268, 2020.

[26] Xinjian Li, Florian Metze, David R Mortensen, Shinji Watanabe, and Alan W Black. Zero-shot learning for grapheme to phoneme conversion with language ensemble. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2106–2115, 2022.

[27] Hui Lin, Li Deng, Dong Yu, Yi-fan Gong, Alex Acero, and Chin-Hui Lee. A study on multilingual acoustic modeling for large vocabulary asr. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4333–4336. IEEE, 2009.

[28] V. S. Mahalle, G. N. Bonde, S. S. Jadhao, and S. R. Paraskar. Teager energy operator: A signal processing approach for detection and classification of power quality events. In *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)*, pages 1109–1114, 2018.

[29] Yeshwant K Muthusamy, Etienne Barnard, and Ronald A Cole. Reviewing automatic language identification. *IEEE Signal Processing Magazine*, 11(4):33–41, 1994.

[30] Sebastian Nordhoff and Harald Hammarström. Glottolog/langdoc: Defining dialects, languages, and language families as collections of resources. In *First International Workshop on Linked Science 2011-In conjunction with the International Semantic Web Conference (ISWC 2011)*, 2011.

[31] D. O'Shaughnessy. Linear predictive coding. *IEEE Potentials*, 7(1):29–32, 1988.

[32] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[33] Daniel Povey, Gaofeng Cheng, Yiming Wang, Ke Li, Hainan Xu, Mahsa Yarmohammadi, and Sanjeev Khudanpur. Semi-orthogonal low-rank matrix factorization for deep neural networks. In *Interspeech*, pages 3743–3747, 2018.

[34] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. The kaldi speech recognition toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, December 2011. IEEE Catalog No.: CFP11SRW-USB.

[35] Daniel Povey, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahremani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur. Purely sequence-trained neural networks for asr based on lattice-free mmi. In *Interspeech*, pages 2751–2755, 2016.

[36] Vineel Pratap, Qiantong Xu, Anuroop Sriram, Gabriel Synnaeve, and Ronan Collobert. Mls: A large-scale multilingual dataset for speech research. *arXiv preprint arXiv:2012.03411*, 2020.

[37] L. R. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ: Prentice Hall, 1993.

[38] Douglas A Reynolds et al. Gaussian mixture models. *Encyclopedia of biometrics*, 741(659-663), 2009.

[39] Tanja Schultz and Alex Waibel. Language-independent and language-adaptive acoustic modeling for speech recognition. *Speech Communication*, 35(1-2):31–51, 2001.

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[41] Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. Phoneme recognition using time-delay neural networks. In *Backpropagation*, pages 35–61. Psychology Press, 2013.

[42] Changhan Wang, Morgane Riviere, Ann Lee, Anne Wu, Chaitanya Talnikar, Daniel Haziza, Mary Williamson, Juan Pino, and Emmanuel Dupoux. Voxpopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation. *arXiv preprint arXiv:2101.00390*, 2021.

[43] Changhan Wang, Anne Wu, and Juan Pino. Covost 2 and massively multilingual speech-to-text translation. *arXiv preprint arXiv:2007.10310*, 2020.

[44] John C Wells. Computer-coding the ipa: a proposed extension of sampa. *Revised draft*, 4(28):1995, 1995.

[45] Kim-Yung Eddie Wong. *Automatic spoken language identification utilizing acoustic and phonetic speech information*. PhD thesis, Queensland University of Technology, Brisbane, 2004.

[46] Marc A Zissman. Comparison of four approaches to automatic language identification of telephone speech. *IEEE Transactions on speech and audio processing*, 4(1):31, 1996.

[47] Marc A Zissman and Kay M Berkling. Automatic language identification. *speech communication*, 35(1-2):115–124, 2001.