SPOKEN LANGUAGE IDENTIFICATION FOR ETHIOPIAN LANGUAGES USING DEEP LEARNING APPROACHES

A Thesis Present

By

Tatek Shenkute

To

The Faculty of Informatics

Of

St. Mary's University

Advisor: Mulugeta Adbaru(PHD)

Addis Ababa, ETHIOPIA

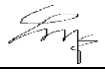
JUNE, 2024

Spoken language Identification for Ethiopian languages using deep learning approaches

By

Tatek Shenkute

Approval by Board of Examiners

Mulugeta Adbaru(PHD)		09/06/2024
Advisor	Signature	Date
_____	_____	_____
Internal Examiner	Signature	Date
Minale Ashagrie(PHD)		**10/07/2024**
External Examiner	Signature	Date
_____	_____	_____
Chairman	Signature	Date

DECLARATION

I, here by affirm that this thesis is my own creation and has not been submitted for a degree at this or any other institution. Furthermore, I acknowledge and appropriately credit all sources used in this thesis.

Declared by:

Author Name: Tatek Shenkute

Signature: _____

Date: _____

Place: Saint marry institute of Technology, Saint marry University, Addis Ababa

This thesis has been submitted for examination under my authorization as a university advisor. Verified by:

Advisor Name: Mulugeta Adbaru (PHD)

Signature:  _____

Date: 05/07/2024 _____

Acknowledgment

I am deeply thankful to all those who have contributed to the completion of my thesis entitled "Spoken Language Identification for Ethiopian Languages Using Deep Learning Approaches." I would like to express my utmost gratitude to my advisors, [Mulugeta Adbaru (PhD)] and [Tesfu (PhD)], for their invaluable guidance, encouragement, and steadfast support throughout this endeavor. I am also appreciative of Saint Marry University for providing the essential resources and facilities required for conducting this research.

I am grateful to the participants who generously contributed their time and linguistic knowledge, essential for the completion of this study. Furthermore, I appreciate the contributions of experts in the fields of deep learning and natural language processing, whose pioneering work provided the basis for this research.

I am grateful to my family for their patience, understanding, and constant encouragement during the challenging phases of this thesis. Their unconditional love has been my source of strength and motivation.

Last but not least, I extend my appreciation to all my friends and colleagues who supported me in various ways, whether through insightful discussions, constructive feedback, or simply lending an empathetic ear. Your encouragement has been instrumental in shaping this work.

Table of Contents

List of Tables

List of figures

List of abbreviations

CNN	Convolutional neural network
DNN	Deep neural networks
LSTM	Long Short-Term Memory
BLSTM	Bidirectional LSTM
SLI	Spoken Language Identification
NLP	Natural Language Processing
LID	Language identification
DL	Deep Learning
SLID	Spoken language identification
GMM	Gaussian mixture model
SVM	Support Vector Machine
LDA	Linear Discriminant Analysis
MFCC	Mel frequency cepstral coefficients
DFT	discrete Fourier transform
DCT	Discrete cosine transform
ReLU	Rectified Linear Un
BPTT	backpropagation through time
GPU	Graphical processing unit
CPU	Central processing unit

Abstract

This thesis investigates the application of Deep Neural Networks (DNN), Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and Bidirectional LSTM (BLSTM) algorithms for Spoken Language Identification (SLI) in the context of Ethiopian languages. With Ethiopia's rich linguistic diversity presenting a unique challenge, this research endeavors to develop robust models capable of accurately identifying spoken utterances across a spectrum of Ethiopian languages. The study involves the collection and preprocessing of a comprehensive dataset encompassing diverse linguistic variations and dialectal nuances prevalent within Ethiopian speech. Through rigorous experimentation and evaluation, the efficacy of DNNs, CNNs, LSTMs, and BLSTMs in classifying spoken language samples is assessed, considering factors such as model accuracy, computational efficiency, and generalization capability. Furthermore, particularly in scenarios with limited labeled data. The outcomes of this research not only contribute to the advancement of SLI technologies but also hold significant implications for communication systems, language preservation efforts, and cultural heritage preservation in Ethiopia and beyond. Our experiment results indicate that the BLSTM algorithm, utilizing MFCC features, performed best for the Ethiopian language identification dataset. Specifically, it achieved an accuracy of 87.5% for 30 seconds, 95% for 10 seconds, and the highest accuracy of 95% for 3 seconds, particularly for Amharic, Tigrigna, and Wolaytigna languages, surpassing other algorithms tested. And DNN model followed achieved the maximum accuracy with a value of 92.5% at a speech duration of 10 s, for all languages. We utilized Librosa library in Python on a CPU with (Hp pro 1 tera) and 8 GB of RAM to tests all experiments.

Keywords: Spoken Language Identification, Ethiopian Languages, Deep Learning, DNN, CNN, LSTM and BLSTM, Language Diversity, Speech Recognition.

CHAPTER ONE

1. Introduction

1.1. Background

Language serves as a tool for human communication, whether through spoken words or written text. When people speak, their primary goal is to convey messages. However, speech not only carries the intended message but also reflects aspects such as the speaker's characteristics and the language being used. The language itself is communicated through a series of sound units[1]. In spoken communication, the information conveyed through language can be subtle and implicit, heavily influenced by the content of what is being said[2].Humans recognize languages with a little training and effort. Automating this ability is the hot issue in the field of computer science and engineering [3],[4]; and the field of studying language identification is called Natural Language Processing (NLP).

Natural language processing (NLP) is a discipline grounded in theoretical principles that utilizes computational methods to automate the analysis and representation of human language. NLP enables computers to perform diverse tasks involving natural language at different levels, including parsing, part-of-speech (POS) tagging, machine translation, and dialogue systems [5]. The methodologies employed in NLP implementation have progressed from modeling techniques such as vector quantization (VQ) [6] and Gaussian mixture modeling (GMM).

Currently, advanced techniques such as Deep Neural Networks (DNNs), Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM), and Bidirectional LSTM (BiLSTM) have demonstrated significant effectiveness in language identification tasks.

Recent advancements in deep learning architectures and methodologies have significantly advanced fields such as computer vision and pattern recognition [2]. Unlike traditional approaches in natural language processing (NLP), such as support vector machines (SVM) and logistic regression, which often employ shallow models trained on high-dimensional and sparse features, deep learning allows for the automatic extraction of intricate hierarchical features. This capability has made deep learning particularly effective in systems focused on speech and speaker recognition, and more recently, in language identification and recognition systems [7,8,5,2].

Language identification (LID) finds extensive applications across various multilingual services, such as directing incoming calls to operators fluent in the caller's language. Humans are currently the most accurate entities in identifying languages, often able to do so within seconds of hearing speech, even if it's unfamiliar, by making subjective comparisons like "sounds like French" [1]. However, automating LID offers numerous advantages including cost reduction and quicker training periods [1,4]. Unlike tasks such as speech recognition or speaker identification where speaker identity or utterance content is known, LID faces the challenge of identifying both the language and speaker from speech data, which varies due to different speakers, channels, and background noise [9]. Effectively capturing language information from short, content-limited utterances requires finding robust representations of language [9].

There are approximately 6,900 known languages spoken worldwide [1]. Therefore, an effective language identification (LID) system should swiftly and accurately leverage various aspects of speech information to distinguish among a vast number of languages. Additionally, it should be adaptable to accommodate the variability introduced by different speakers.

An automatic language identification (LID) system is crucial for ensuring effective communication between users and the system. It quickly determines the likely languages spoken in incoming speech, significantly reducing the time needed to find an interpreter [10]. Our research focuses on developing a spoken LID system based on deep learning specifically for four Ethiopian languages: Amharic, Oromigna, Tigrigna, and Wolaytigna.

Ethiopia is home to approximately 200 dialects categorized into Semitic, Cushitic, Omotic, and Nilo-Saharan language groups, with about 86 distinct languages [11]. The Semitic family includes Amharic and Tigrigna, Cushitic includes Oromigna, and Omotic includes Wolaytigna. These languages were selected due to their significant number of native speakers and diverse usage across various regions in Ethiopia.

Amharic serves as the primary language in administrative functions within the Amhara regional state and the Federal Government. Tigrigna is the official language of the Tigray regional state and Eritrea. Oromigna is spoken by approximately 37 million people in the Oromia regional state. Wolaytigna is the official language for around 6 million people in the southern part of Ethiopia [11].

In our research, we applied an innovative deep learning method for Ethiopian Language Identification (LID), utilizing algorithms including Deep Neural Networks (DNN), Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and Bidirectional LSTM (BLSTM). LSTM, known for its effectiveness in handling sequential data such as audio recordings [12], played a crucial role. By integrating these advanced algorithms, our goal was to achieve high accuracy by harnessing the output data from preceding layers.

We utilized Mel-frequency cepstral coefficients (MFCC) for feature extraction, which are effective in capturing relevant information from audio signals, thus enabling reliable identification of the targeted Ethiopian languages. Various front-end and back-end techniques have been explored in existing LID systems, demonstrating the versatility and adaptability required to handle different types of speech data [7, 8, 13].

figure 1. 1 Levels of Language Identification System

As Figure 1.1 depicts, the LID system has two major architectural levels; front-end level (i.e., feature extraction) and back-end level (i.e., classification and prediction). In the back-end level multiple classification techniques have been employed and one of them is Deep Neural Networks (DNN), Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and Bidirectional LSTM (BiLSTM) is best language identification algorithms.

1.2. Statement of the Problem

Acoustic features encompass the physiological aspects of a speaker, such as vocal tract characteristics, which aid in distinguishing between languages [9].They are also the easiest to be captured from a speech; but they might not be as efficient as the combination of all the features that are acoustic with phonetic and prosodic features. In spite of its being easy and telling small amount of information about a language, what will happen if we use better training technique made us to be eager. Present telephone-based information services utilize interactive

voice response (IVR) systems. IVR is an automated system that engages with callers, gathers information, and directs calls to the appropriate recipient. Creating a Language Identification (LID) system would enable rapid processing of spoken customer data, identifying the language used, and swiftly connecting calls to an operator fluent in that language.

The second thing is that Ethiopia is a multilingual country with over 80 existing languages [1]. But as per our knowledge in doing this research, there are insufficient studies made to automate identifying languages. In our baseline paper[1] of this study we figured out that it is possible to study the effect of using different algorithms and techniques based on a given countries context. This is very important in a means that; if someone who wants to implement LID on Ethiopian languages they might find our study useful. But it doesn't mean that there was no research at all. From Addis Ababa University (AAU) [1], we have found out one research which was studied by utilizing the modeling technique of GMM. Therefore, if it has been done based on the earlier technique than the-state-of-the-art, then we could also study LID based on the more recent and currently shining deep learning classification technique. To make it more sensible, we use DNN, CNN, LSTM and BLSTM algorithm as our baseline framework and we can compare the accuracy of the proposed LID system based on each other's. The number of languages it considered was also only four that is, Amharic, wolayitegna, Oromigna and Tigrigna .

In various studies, such as those focusing on speech recognition and speaker verification, DNNs have demonstrated superior performance compared to previous dominant techniques like CNN, LSTM, and BLSTM. This advancement has been noted in a wide range of demanding machine learning applications, such as acoustic modeling and visual object recognition.

The proposed system addresses some of the limitations encountered in DNN, CNN, LSTM, and BLSTM algorithms when applied to language identification (LID), particularly concerning the number of languages to be trained and tested. Moreover, the system effectively leverages different aspects of speech information to accurately and rapidly discriminate between various languages, even within a vast array of target languages. Additionally, it exhibits flexibility in accommodating variations among different speakers.

1.3. Research questions

The primary goal of this study is to design and compare the performance of DNN, CNN, LSTM,

and BiLSTM algorithms in identifying spoken Amharigna, Oromigna, Tigrigna, and wolayitegna languages. The specific research questions are as follows:

1. How to prepare testing and training dataset?

2. Which feature selection algorithm is employed?

3. How accurate and appropriate are the DNN, CNN, LSTM, and BiLSTM algorithms in identifying spoken Amharigna, Oromigna, Tigrigna, and wolayitegna languages?

4. What are the factors that influence the performance of these algorithms in language identification?

1.4. Objective

1.4.1 General Objective

The primary aim of this research is to create an effective Spoken Language Identification system for Ethiopian languages using Deep Neural Network (DNN), Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and Bidirectional LSTM (BiLSTM) algorithms.

1.4.2 Specific Objectives

- ✓ To gather a substantial dataset of spoken utterances in the four Ethiopian languages for training and evaluating the identification models.
- ✓ To design and implement the best DNN, CNN, LSTM and BLSTM model for spoken language identification to classifying the audio samples.
- ✓ To choose the most suitable feature selection algorithm for language audio signals.
- ✓ To assess and compare the performance of the developed models using relevant evaluation metrics like accuracy, precision, recall, and F1-score, to identify the most effective algorithm for spoken language identification in the Ethiopian context.

1.5. Motivation

The motivation of this thesis work is that since Ethiopia is a multi-lingual country; this can be considered as a blessing for those who know its value; because it creates a colorful culture. But it is

also a concern around security agencies, emergency services and telephone companies to give service for their customers in their own language. If we consider it, this could be an opportunity for a researcher to solve by studying and implementing automated systems. Language Identification system is a widely used system and researches are flooding from all over the world. But, the researches around this field in Ethiopia are not going as it supposed to be; also it is usually done on earlier algorithms. So this thesis work will be timely in regard to applying the most recent algorithm. This research work also will be supportive for anybody who is interested in this research area in Ethiopian context. My motivation came from a research titled LID based on GMM from AAU in 2017 G.C [1]; so I become interested to work on this area and contribute my part.

1.6. The Scope of the Study

This study will focus on implementing SLID system with the state-of-the-art and most efficient pattern classification technique using DNN, CNN, LSTM and BLSTM for four selected Ethiopian languages. In the SLID system given that every language that is found in the whole world has its own unique identifying characteristics; it is required to study the uniqueness of the specific language. It is also important to make sure we are using the best algorithm by considering and comparing the accuracy of the result. We have implemented the more accurate DNN, CNN, LSTM and BLSTM algorithms are evaluated the best results by comparing to each other's.

In every country there are researches in progress to include their languages to the LID system [3],[16],[17]; in Ethiopia there was a research that is based on GMM algorithm at AAU in 2017 G.C so this study can be considered as the proceeding of [1].

Due to the vast number of languages spoken in Ethiopia, this study focuses specifically on four languages: Amharic, Oromo, Tigrinya, and Wolaytta. These languages were selected based on their speaker population and prevalence in the country. Also the length of data samples taken is in 3, 10 and 30 seconds to make the learning and testing process measureable. Samples taken for each language at the training level is 50 frames; total numbers of training samples are 4*50=200 frames. Total number of 8,600 sec languages for all training parts. In the testing process we used total number of data samples 20, which are 5 frames for each language 4*5=20 frames. Total number of 860 sec recorded languages for all training parts

1.7. Limitation of the Study

The proposed system's focus is on extracting the acoustic features of the sample data of each selected language at the front-end level using the most efficient feature extraction algorithm MFCC [30], for training the DNN, CNN, LSTM and BLSTM [11], [10],[1] system with the given dataset and predicting the languages based on the trained model and unknown utterance features of the testing dataset. The DNN, CNN, LSTM and BLSTM based LID will be trained on four different selected Ethiopian languages (i.e., Amharic, Oromigna, Tigrigna and Wolaytigna) in the training phase and the corresponding languages are predicted in the testing phase.

Additionally due to the lack of time and resource constraints we did not include all the utterances/speeches of the respected language; instead we have used 3, 10 and 30 second speech from each language. Since learning systems have high resource consumption, we didn't consider the CPU usage and time issues as parameters at this thesis work.

1.8. Significance of the Study

Language Identification (LID) is utilized in numerous multilingual services, such as routing incoming telephone calls to human switchboard operators fluent in the caller's language. Automating LID offers several advantages, including cost savings and shorter learning periods compared to human language identification systems [3].

Therefore, this project is important in the identification system applications for Ethiopian languages and will be stepping stone for future researches.

1.9. Organization of the Thesis

After this introduction chapter we will proceed to the second chapter which is the literature survey with introduction, literature reviews and Conclusion of literature reviews. Chapter three, focusing on methodologies, provides a comprehensive description of the proposed system architecture and algorithms. It begins by explaining the preparation of raw speech data and datasets. Following an overview of the system's overall pipeline, the chapter delves into the specifics of each step within the system model. It concludes by examining the preprocessing of speech data, feature extraction, and classification methods.

Chapter four begins with an overview of the tools utilized in the experiment. It then progresses through presenting experimental results and comparing the accuracy of various LID systems based on CNN, DNN, LSTM, and BLSTM architectures as researched in this work.

The final chapter is conclusion and future work followed by references used in the thesis work. And the last chapter is conclusion, recommendation, future work and reference of this work.

CHAPTE TWO

2. Literature review

2.1 Introduction

Spoken Language Identification (SLID) is the process of automatically recognizing the language spoken in an audio recording. It plays a crucial role in applications like speech recognition, speaker verification, and language translation. Research in LID systems did not emerge until the 1970[2], and progress was initially slow for nearly two decades. However, advancements gained momentum with the availability of openly accessible multilingual speech corpora[2].

LID systems have since evolved, primarily differing in their approaches to modeling languages. Recently, deep learning algorithms have shown substantial progress and promising outcomes in Spoken Language Identification (SLID).

2.2 Literature reviews

A Deep Neural Network (DNN), also referred to as a deep feedforward neural network or multilayer perceptron (MLP), is a type of deep learning model specifically created to approximate complex functions. In the context of classification tasks, such as image classification where f(x;θ)=yf(x; \theta) = yf(x;θ)=y, a feedforward neural network maps an input xxx to an output category yyy. This mapping is defined by a set of parameters θ\thetaθ, which the network learns through training to achieve the optimal approximation of the function [14].

Deep learning involves training systems to emulate the learning capabilities of the human brain. It employs multiple layers of nonlinear processing units to extract and transform features from data.

The term "deep" signifies the depth of these layers in the network architecture[14].

Different deep learning architectures, such as deep neural networks, deep belief networks, and recurrent neural networks, have shown success in various domains like computer vision, speech recognition, and natural language processing. They frequently achieve results that are comparable to or even surpass human performance in specific tasks.

Deep neural networks are characterized by their feedforward structure, which includes multiple hidden layers. These layers empower the network to learn complex transformations from input to output, handling both linear and nonlinear relationships effectively. As data passes through these layers, the network computes probabilities associated with each output class or category.

For example, a DNN trained to classify dog breeds from images would analyze each input image and calculate probabilities for different dog breeds. Users can then interpret these probabilities, potentially filtering out results below a certain threshold to determine the most likely breed.

In speech recognition tasks, DNNs have been effective as acoustic models or for extracting bottleneck features from audio data, contributing to significant performance improvements[2].

DNNs excel at modeling complex nonlinear relationships and leverage the composition of layered representations to effectively handle data complexity. During training, weights between neurons are adjusted iteratively based on the network's performance, enhancing the network's ability to recognize patterns and make accurate predictions[2].

In summary, deep neural networks are pivotal in modern machine learning, enabling efficient processing of complex data and consistently achieving state-of-the-art results across diverse domains.

Convolutional Neural Networks (CNN) has demonstrated high effectiveness in various speech and audio processing tasks, making them an ideal choice for SLI[12]. CNNs are deep learning models that can automatically extract meaningful features from audio signals, capturing both phonetic and acoustic characteristics. The approach for SLI using CNN starts with preprocessing and feature extraction from the audio recordings [12]. This may involve converting the audio to a spectrogram or other suitable representation that CNN can process effectively. The extracted features are subsequently input into the CNN model, which is trained using a extensive dataset of labeled audio recordings from the four languages.

The CNN model undergoes an extensive training process, optimizing its parameters to accurately identify the spoken language. The model learns to differentiate the unique phonetic and acoustic patterns present in each language, enabling it to recognize the target language accurately.

By accurately identifying the spoken language in audio recordings of Amharic, Oromo, Tigrigna,

and Wolayitegna, the SLI system using CNN can contribute to various applications that require language-specific processing [11]. It can facilitate more effective communication services, language-based information retrieval, and enable better access to technology for the people of Ethiopia.

Long Short-Term Memory (LSTM) Language identification, also known as language detection, is the process of automatically identifying the language of a given text or speech signal. In this case, the task is to identify spoken languages from four Ethiopian languages using the Long Short-Term Memory (LSTM) algorithm.

LSTM is a variant of recurrent neural network (RNN) known for its ability to capture long-term dependencies in sequential data [11], making it well-suited for tasks like natural language processing (NLP) and speech recognition. It has been extensively applied in diverse language-related applications, including language identification.

To develop language identification using the LSTM algorithm, you would need a dataset consisting of spoken recordings in each of the four Ethiopian languages. This dataset would need to be properly labeled with the corresponding language for each recording [1,11].

Bidirectional long short-term memory (BLSTM) belongs to the category of bidirectional recurrent neural networks (BRNN). Initially introduced in [11], It consists of two hidden layers that provide distinct directions to the same output. This setup allows the output layer to integrate information from both preceding (backward) and subsequent (forward) states simultaneously. Each LSTM is composed of interconnected and progressively intricate subnetworks known as "memory cells" or gates. These components enable the model to effectively capture long-range dependencies in the flow of information.

Consequently, as data progresses through the layers, fresh values are added to the activation, mitigating the vanishing gradient problem often encountered in LSTM networks and making them comparable to residual neural networks [11].The combination of these algorithms ensures comprehensive coverage across different aspects of audio data, enabling robust language identification even when dealing with the subtle differences between the targets Ethiopian languages. Improved SLI on these languages can have transformative benefits for various applications, such as transcription services, language translation, and speech recognition technology, ultimately facilitating effective communication and language understanding in

Ethiopia. It is possible to develop highly accurate SLI systems.

2.3 Related works

Mikias Wondimu and Tekeba[1]: In a language identification system designed for four Ethiopian languages Amharigna, Oromifa, Gurgegna, and Tigregna a Gaussian mixture model was employed. The system utilized the (MFCC) for feature extraction and GMM for classification. Notably, the study did not enforce a fixed duration or segment size for audio segmentation.

Across the four languages, the average accuracy in the utterance-dependent Language Identification (LID) test was 93%. In the utterance-independent test, the average accuracy was approximately 70%. In the speaker-independent test, evaluated under the utterance-dependent scenario, the system achieved an average accuracy of around 91%.

Alemu, A.A., Melese, M.D. & Salau, A.O.[11]: This paper presents an Ethio-semitic language identification system using audio, leveraging Recurrent Neural Network (RNN) technology. Given the high similarity among characters within each language, identifying distinct features that can accurately differentiate between languages is a significant challenge. To tackle this issue, the system integrates Recurrent Neural Network (RNN) with Mel-frequency cepstral coefficients (MFCCs) for feature extraction, achieving promising results.

The study aims to identify the best model for Ethio-semitic language identification, focusing on languages such as Amharigna, Geez, Guragegna, and Tigregna. An 8-hour collection of audio recordings is used for testing the models. Experiments are conducted using a single dataset, employing an extended version of RNN, including Long Short Term Memory and Bidirectional Long Short Term Memory for 5 and 10 sec durations, respectively.

The results indicate that the Bidirectional Long Short Term Memory (BLSTM) model outperforms the Long Short Term Memory (LSTM) model. Specifically, the BLSTM model achieves average training, validation, and testing accuracies of 98.1%, 92.9%, and 89.9%, respectively.

Panji Wijonarko, Amalia Zahra [12]. This paper investigates the use of spoken language identification technology to improve tourism and digital content in Indonesia, with a specific focus on four local languages: Javanesse, Sundanesse, Minangkabau, and Buginesse. The study employs deep learning classification techniques such as artificial neural network (ANN), convolutional

neural network (CNN), and long short-term memory (LSTM). Feature extraction utilizes Mel-frequency cepstral coefficients (MFCC) for analyzing audio data.

Datasets for each local language are sourced from the internet, specifically from openslr.org for Javanese and Sundanesse, and from YouTube.com for Minangkabau and Buginesse. Each dataset comprises a total duration of 200 minutes.

Experimental results indicate that for a speech duration of 30 seconds, the LSTM (4) and CNN (2) models achieves the highest accuracy at 88.1%, followed closely by the ANN (4) model with 87.5%. For shorter durations of 10 seconds, the LSTM (4) model succeeds the highest accuracy of 88.8%, followed by CNN (4) at 87.1%, CNN (2) at 85.6%, and ANN (2) at 85.4%. Similarly, for a duration of 3 seconds, the LSTM (4) model shows the highest accuracy at 87.2%, followed by CNN (4) at 86.1% and CNN (2) at 85.5%.

Throughout the study, across all speech durations (3s, 10s, and 30s), the LSTM model consistently performs the best in terms of accuracy, followed by the CNN model in second place and the ANN model in third place.

Chithra Madhu, Anu George and Leena Mary.[13]. Languages vary in their phonological units, their frequency, and the order in which these units appear in words. Detecting and leveraging these differences among languages is crucial for automatic language identification (LID). This paper focuses on exploring phonotactic and prosodic features extracted from speech signals to address various aspects of language identification.

Unlike traditional approaches that require constructing language models based on phonotactic patterns, this work eliminates that need. Instead, it uses syllables as the basic units for feature representation. Syllables, comprising multiple phonemes, can capture significant language-specific co-articulation effects. The segmentation of speech into syllable-like units is achieved using valley points detected in the short-time energy contour.

Phonotactic and prosodic features are extracted using syllable boundaries. Prosodic feature vectors are created by combining features from three consecutive syllables.

At the backend, a multilayer feedforward neural network classifier is employed for language identification. This NN classifier utilizes the extracted phonotactic and prosodic features to distinguish between different languages.

In summary, this approach leverages syllables for feature representation, extracts phonotactic and prosodic features from these syllables, and employs a neural network classifier for effective language identification without the need for constructing conventional language models based on phonotactic patterns.

Pinki Roy and Pradip K. Das [18]: The research demonstrates the effectiveness of a language identification (LID) system designed for speech, focusing on four distinct Indian languages: Indian English, Hindi, Assamese, and Bengali. The evaluation of these languages utilizes established recorded databases, with feature extraction carried out using Mel frequency cepstral coefficients (MFCC), and classification performed using Gaussian mixture models (GMMs). Experiments are conducted using a proprietary standardized language database compiled from recordings by 50 speakers. The results show that the highest accuracy in LID across all languages is achieved with a mixture order of 1024. The accuracy of LID ranges from a minimum of 93% for Assamese to a maximum of 100% for Bengali, Hindi, and Indian English.

David Martinez, Lukas Burget:[19] A language recognition system has been created to automatically extract prosodic information from speech and employ a generative classifier based on iVectors. The system's performance was evaluated using the NIST LRE09 dataset.

By incorporating a prosodic system with 2048 Gaussians and 400-dimensional iVectors, along with a fusion approach integrating both systems, notable performance improvements were achieved across all testing conditions.

Relative to the acoustic system, the enhancements in accuracy were substantial: 10.93% improvement for 3-second segments, 15.24% improvement for 10-second segments, and 9.39% improvement for 30-second segments. These results demonstrate the efficacy of integrating prosodic information and utilizing iVectors for enhancing the accuracy of language recognition systems.

v. Ramasubramanian A. K. V. Sai Jayram T. V. Sreenivas [20]: The study explores innovative challenges in the parallel phone recognition (PPR) system for automatic language identification (LID). It assesses three critical score types in LID: acoustic score, language model score, and joint acoustic-language score. Each score type is utilized to develop classifiers using three distinct methods: maximum likelihood classifier (MLC), Gaussian classifier (GC), and K-nearest-neighbor classifier (KNNC). The study rigorously compares the performance of these classifiers across various combinations of scoring methods.

The findings highlight that MLC, particularly when applied with bias-removal techniques, achieves the highest performance when utilizing either the acoustic score or the language model score independently. Additionally, GC utilizing the joint acoustic-language score closely follows in performance. These results underscore the effectiveness of specific classifier-score combinations in enhancing the accuracy of LID systems, addressing critical aspects in PPR system development previously unexplored.

Kinnunen, Tomi et al. (2017)[21]: The study proposed a BiLSTM approach for spoken language identification using frame-level acoustic features. They compared the performance of BiLSTM with other traditional machine learning algorithms and found that BiLSTM achieved superior accuracy. The researchers also experimented with an ensemble of BiLSTM models and reported further improvement in identification accuracy.

Zhang, Xinglei et al. (2019)[22]: In this research, LSTM-based DNN architecture was used for spoken language identification. The LSTM layers were employed to model both frame-level and sequence-level features. The study demonstrated that using LSTM layers in the DNN model significantly improved the identification accuracy, outperforming traditional DNN models that neglected the temporal dependencies.

Gundeep Singh, Sahil Sharma, Vijay Kumar, Manjit Kaur, Mohammed Baz, Mehedi Masud[23],In this paper, we introduced a deep learning-based framework for spoken language identification. This framework involves converting audio utterances into spectrograms, which represent the frequency and time components of the speech signal. Next, a convolutional neural network (CNN) processes these spectrogram images to extract pertinent features for classification. The softmax activation function is then employed for multi-language classification. The dataset consists of recordings from 90 distinct speakers, encompassing both male and female voices.

We conducted experiments using four different datasets, each containing audio samples. The first dataset includes recordings in three languages, the second dataset comprises samples from 22 languages, and the third dataset contains recordings in 16 languages. All datasets are available on Kaggle, while the fourth dataset, which contains samples from four languages, can be accessed on the Mozilla website.

In the image domain, 2D convolutional neural networks achieved an impressive accuracy of 98%. Furthermore, utilizing word embeddings with a retrained model on a CSV dataset resulted in an accuracy of 95%. Employing the Bernoulli Naïve Bayes approach yielded an accuracy of 93% on the 22-language dataset. Finally, SVM and random forest classifier models achieved accuracies of 82.88% and 72.42%, respectively, on the 16-language dataset.

Shukla, Harsh et al. (2020) [24]: The study explored the effectiveness of BiLSTM networks in combination with spectrogram features for spoken language identification. The researchers designed a novel architecture that combined 1D CNN layers with BiLSTM layers. This approach captured both local spectral features and long-term dependencies, leading to improved identification accuracy.

2.4 Conclusion of literature reviews

While many authors have explored various languages, our paper focuses specifically on four languages. It's worth noting that there is currently no existing dataset for Spoken language identification for Ethiopian languages within the existing online literature. Additionally, we employ novel methods that have not been previously utilized, such as Deep Neural Network (DNN), Convolutional Neural Network (CNN), Long Short Term Memory (LSTM), and Bidirectional Long Short Term Memory (BLSTM), for the identification of four Ethiopian languages: Amharigna, Oromigna, Tigregna, and Wolayitegna. These models demonstrate superior performance compared to traditional methods.

CHAPTER THREE

3. Methodology

This section elaborates on the methodologies utilized to achieve the research objectives successfully. The study is divided into several subsections, covering data collection and annotation, audio preprocessing, data partitioning, feature extraction, model training, and model evaluation.

Particular emphasis is placed on a unique aspect of the research: the identification of specific spoken languages within the context of Ethiopian languages.

3.1 Preparing data samples

Since there was no existing dataset for spoken language identification in Ethiopian languages, we compiled our own dataset by capturing speeches from FM97.1 Ethiopian Radio and downloading videos from youtube.com. This approach enabled us to conduct the automatic language identification task effectively despite the absence of pre-existing data for the selected languages. After downloading and recording[12]; it was necessary to make format change and break apart the file in the size required by the systems. After remove silence and make equal size of a large number of 16 kHz 16-bit wave data for the four selected different Ethiopian languages (i.e.,Amharigna, Oromigna, Tigrigna and Wolaytigna). Each corpus is prepared using a diverse collection of audio recordings that encompass various ages, genders, and accents. Each language's prepared for 200 frames (above 2:30 h) for training (i.e., 50 data samples for each languages and each with 3, 10, 30 second length) and 20 frames for testing (i.e., 5 data samples for each languages with 3, 10 and 30 seconds long). The length of the whole training samples is 200 x3 seconds (i.e., 10 minutes), 200 x 10 seconds (i.e., 40 minutes) and 200 x 30 (i.e., 100 minutes =2 hours and 30 minutes long for training).

3.2 system architecture

This section provides an overview of the methods used to develop the proposed Language Identification (LID) model, as depicted in Figure 1. It focuses on the stages of corpus preparation, the system's architecture, preprocessing techniques, the generation of machine-readable CSV datasets, and procedures for model evaluation. Figure 1 illustrates the architecture of the proposed LID model, encompassing essential steps like data preprocessing, frame, feature extraction, and evaluation model. Each of these components is detailed to demonstrate how the model was constructed and evaluated for effective language identification.

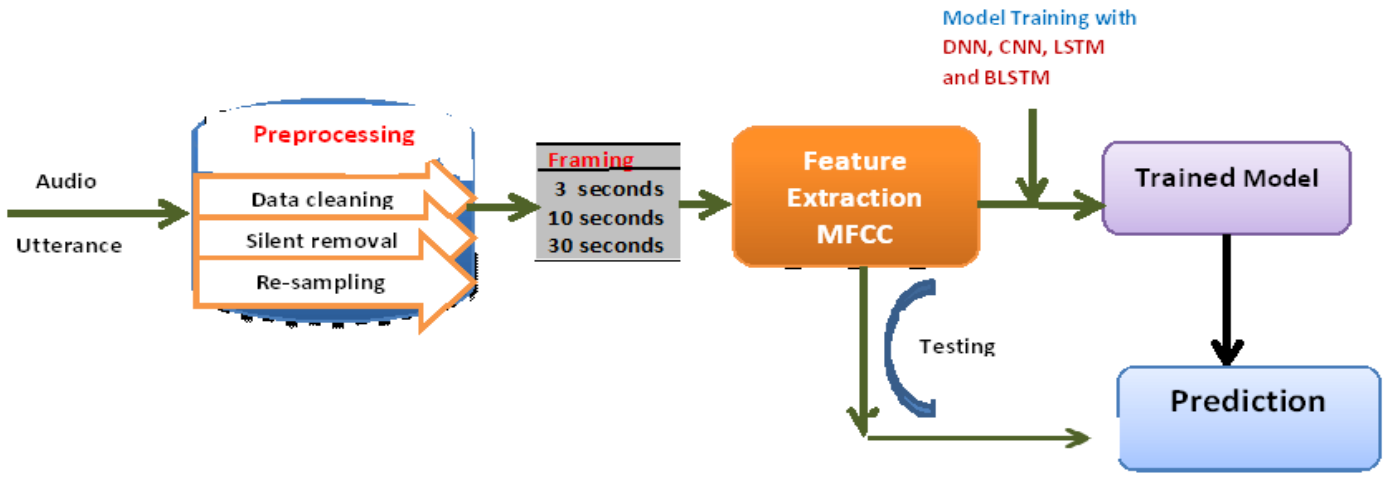


Figure 3.1 Proposed system architecture

3.2.1 Data Preprocessing

After data collection, the initial stage of the Language Identification (LID) system involves preprocessing the raw speech data to prepare it for the intended algorithms. Data preprocessing encompasses all operations conducted on raw data to ready it for subsequent processing stages. This crucial step establishes the foundation for converting the data into a format that can be processed efficiently and accurately.

3.2.1.1 Data Cleaning

Audio data cleaning is imperative prior to any further processing steps. Often, during data collection through recordings, speakers may use mixed languages such as Amharic with English or Amharic with Oromigna. Therefore, to formulate clean data for the proposed system, these mixed audios are removed using the pydub package, and the sound files are kept in .WAV format.

3.2.1.2 Silent Removal

To eliminate the silent sound from the actual speech .wav files, we utilized the librosa trim function. The criterion for removing silence is based on normal sound intensity, which is set to be above 10 dB. Decibel measures sound intensity and a threshold of 10 dB is employed to filter out silence from speech. Any speech falling below this threshold is removed from the entire sound file. This threshold value is selected based on literature review, as several studies have employed the same sound intensity level. It is considered a standard value, ensuring that speech quality for communication is adequate, with a signal-to-noise ratio (SNR) of -10 dB $\leq$ SNR. Speech equal to or greater than 10 dB ensures satisfactory quality for speech communication.

3.2.1.3 Framing

Framing in the context of speech processing to dividing a continuous flow of speech samples into fixed-length segments, allowing for systematic processing of the signal in blocks. Speech signals are characterized by their non-stationary nature, where sound waves constantly vary over time. To accommodate this variability, speech signals are segmented into frames, which are divided into a small subdivisions typically measured in seconds or milliseconds. These frames are essential for analysis and processing. To prevent the loss of important signal information during segmentation, consecutive frames often overlap by 30% to 50%. This overlap ensures that critical aspects of the voice signal are not missed due to windowing effects. Researchers can choose from various segment lengths such as 3 s, 10 s, 30 s, and others like 5 seconds and 20 seconds, depending on their specific needs. In the current study, the researchers opted for intermediate segment lengths of 3 seconds, 10 seconds, and 30 seconds. These segments were chosen to frame audio segments from a library for each language's sounds, facilitating effective analysis and processing.

3.2.2 Feature extraction

In this study, the speech signal undergoes preprocessing before inputting it into the Language Identification (LID) system. This preprocessing includes amplifying weaker signals, removing prolonged silences, and isolating speech from background noise to enhance clarity for further analysis.

Feature extraction is then employed to convert the processed speech signal into acoustic feature vectors that contain speaker-specific information. The researchers utilize the Librosa package, a tool commonly used for audio analysis, to extract relevant components from the audio data. Several techniques exist for feature extraction, including Linear Predictive Coding (LPC), Mel-frequency cepstral coefficients (MFCC), mel-spectrogram, Spectral Contrast, and Relative spectral filtering of log domain coefficients (RASTA), among others. Each of these techniques captures distinct aspects of an audio speech segment and can be utilized in designing various LID system configurations, each with its own complexity and effectiveness.

In this particular study, the researchers opt for an acoustic-phonetics approach, employing MFCC as their chosen feature extraction technique. MFCCs are widely recognized in speech and audio processing for their ability to represent the spectral characteristics of sound signals through a set of coefficients. These coefficients are derived using the Mel scale, which mimics the nonlinear frequency resolution of the human auditory system. The process of computing MFCC involves steps such as applying the discrete Fourier transform (DFT), mapping the resulting spectrum onto the Mel scale, and taking the logarithm of the amplitude values.

The Mel scale is pivotal in MFCC computation as it aligns more closely with human auditory perception of sound frequencies. This scale utilizes a series of band-pass filters that analyze the speech signal across various frequency bands, with the positions of these filters mapped according to the Mel scale. The formula for converting from frequency to the Mel scale is integral to this process.

In line with previous research findings, the study employs 20 MFCC coefficients, as these are generally sufficient to encapsulate the most significant information within the speech signal. This approach ensures that the extracted features effectively capture the essential aspects of the speech needed for robust language identification.

$$\text{Mel}(f) = 1125 \ln(1 + f/700) \quad (3.1)$$

Triangular filters are utilized in MFCC to weight the Discrete Fourier Transform (DFT) of the speech, ensuring that the outputs represent the energies of the filter-bank signals. The outputs from the band-pass filters are subsequently processed to compute the MFCC using the Discrete Cosine Transform (DCT), which is defined as:

$$w[n] = 0.54 - 0.46 \cos(2\pi n/L) \; 0 \leq n \leq L - 1 \quad (3.2)$$

Where, $[n]$ is represents window.

L is the speech extracted window or frame

Most of the audio clips range from zero to three seconds in length for CNN, LSTM, BLSTM, and fixed three seconds for DNN-based LIDs. The speech is segmented into frames of 25ms duration, with each frame overlapping by 10ms, and each frame is multiplied by a Hamming window to smooth out discontinuities in the signal. From these frames, we extract features known as mel-frequency cepstral coefficients (MFCCs). The Mel-frequency cepstrum represents the audio signal on the Mel scale, which is a nonlinear mapping of frequencies that simulates the human ear's response to sound by down-sampling higher frequencies. In our implementations, we use the first 13 cepstral coefficients as the primary features, as is common in similar applications. Figure 3.2 illustrates a visualization of these features.

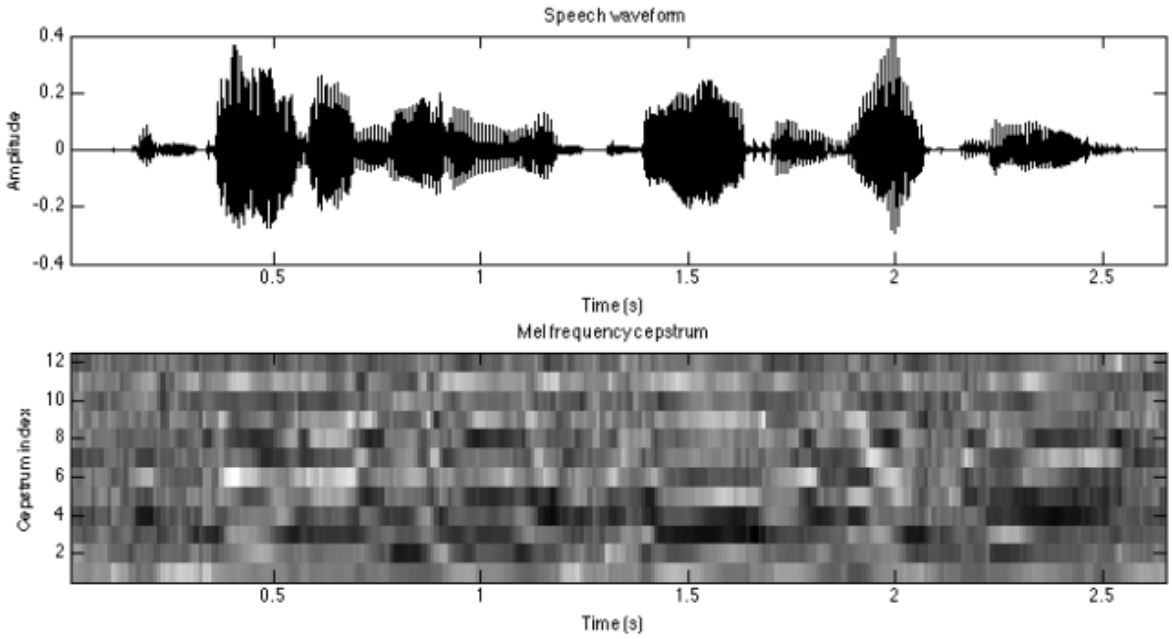


Figure 3. 2 The audio sound wave and its cepstral coefficients

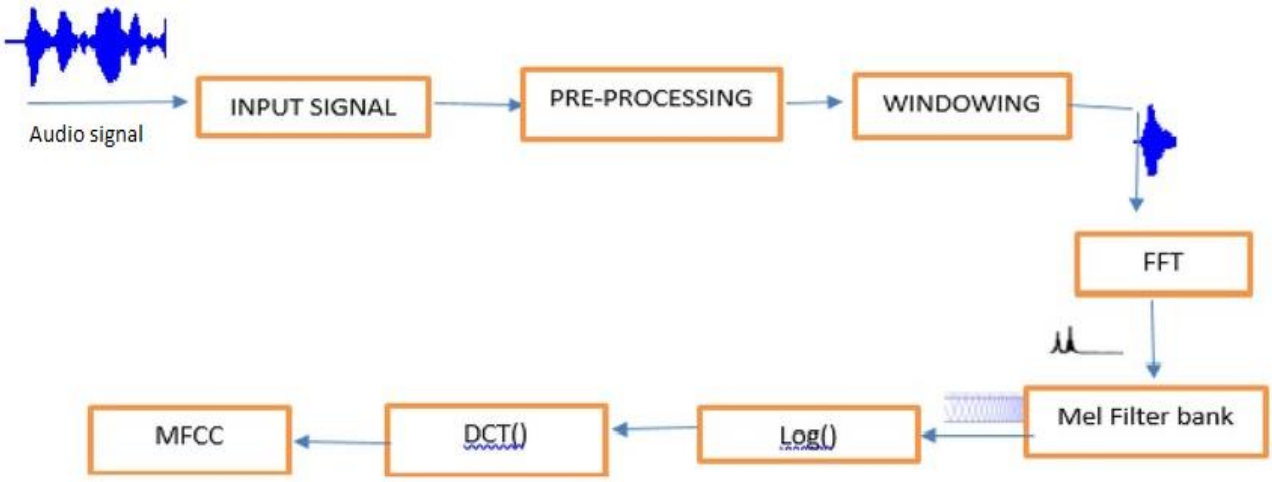


Figure 3. 3 The process of feature extraction

Steps

- Divide the signal into short frames.
- Calculate the periodogram estimate of the power spectrum for each frame.
- Utilize the Mel filter bank to analyze the power spectrum and aggregate the energy within

each filter.

- Logarithmically transform all values obtained from the filter bank.
- Perform a Discrete Cosine Transform (DCT) on the logarithmically transformed filter bank energies.

The primary objective of feature extraction is to obtain a spectral attribute that aids in constructing classifiers for phones or sub-phones. Given speech's non-stationary nature where statistical attributes vary rapidly over time it's impractical to extract spectral features from the entire utterance. Instead, we focus on small windows of speech that can be assumed to exhibit stationary characteristics. Each of these extracted segments is referred to as a frame, capturing spectral features specific to a particular sub-phone.

3.3 Classification

In the realm of language identification, several prominent classification techniques are widely utilized: Deep Neural Network (DNN), Convolutional Neural Network (CNN), Long Short-Term Memory network (LSTM), and Bidirectional Long Short-Term Memory network (BLSTM).

Deep Neural Networks (DNN) are artificial neural networks distinguished by their multiple layers positioned between the input and output layers. Each layer comprises neurons interconnected with neurons in adjacent layers. DNNs are renowned for their capacity to achieve high accuracy in classification tasks. However, training DNNs can be computationally intensive, necessitating powerful hardware such as GPUs or substantial CPU resources. The training process is intricate due to its non-guarantee of convergence to an optimal solution.

For example, in our approach to spoken language identification (SLID) for four Ethiopian languages (Amharic, Oromo, Tigrigna, and Wolayitegna), we utilize a DNN architecture. This DNN architecture consists of four hidden layers, each comprising 64 units and utilizing the Rectified Linear Unit (ReLU) activation function. At the top, a Softmax layer is included with fifty classes, corresponding to the identified languages. During training, batches of 30 samples are processed per iteration, and training is conducted over 50 epochs.

The DNN works by iteratively adjusting the weights of connections between neurons to reduce the error between predicted and actual outputs, thereby learning to classify input data accurately. This methodology leverages the expressive power of deep learning to discern linguistic patterns indicative of specific languages, as outlined in the formulas and techniques tailored for SLID in Ethiopian languages.

In feedforward operation, DNNs process input data by passing it through the network in a forward direction until it reaches the output layer. Each neuron in a layer computes a weighted sum of its inputs, adds a bias, and applies an activation function to generate its output. Here are the formulas for DNN operations:

Weighted Sum: $z=Wx+b$

Z represents weighted sum of inputs.

W is the weight matrix that connects the current layer to the previous layer.

B is represents vector.

Activation Function: a=σ(z) where a is the output of the neuron of activation function. σ is the activation function (e.g., ReLU).

Layer-by-Layer Computation the feedforward operation is performed layer by layer until the final layer is reached.Each layer applies the weighted sum and activation function to its inputs to produce outputs, which serve as inputs to the next layer.

- ✓ Output Layer and Softmax: In SLID, the output typically consists of neurons representing the probability distribution over the languages. Softmax activation is often used in the output layer to ensure that the outputs represent probabilities.Formula for softmax activation (assuming zi is the output of neuron i in the output layer):

softmax(zi)= $e^{zi} / \sum_{j-1}^{N} e^{zj}$ N is the number of neurons output layer.

Prediction language with the highest probability output by the softmax layer is selected as the predicted language for the input audio.

Training:

DNNs are trained using backpropagation with gradient descent-based optimization algorithms.

Categorical cross-entropy is the loss function computed based on the network's output and the ground-truth labels.

Gradients are backpropagated through the network to update the weights and biases.

By training a DNN on features extracted from audio recordings in the four Ethiopian languages, you can build a SLID system capable of accurately identifying spoken Amharic, Oromo, Tigrigna, and Wolayitegna languages.

Convolutional neural network (CNN) In the proposed deep learning-based framework for spoken language identification, the process begins by converting audio utterances into spectrograms, which represent the frequency and time components of the audio signals. These spectrograms are then processed using a (CNN) to extract features suitable for classification. Finally, softmax activation function is employed to classify the spectrograms into multiple languages.

To delve into how a Convolutional Neural Network operates, we can break down its key operations with the following formulas:

The convolution operation is the core building block of CNNs. It involves applying a filter (also known as a kernel) to the input data to produce feature maps. Formula for a 2D convolution operation (assuming a single channel input and filter):

Formula $(I \star K)(i,j) = \sum_m \sum_n I(m,n) \cdot K(i-m, j-n)$

I represent the input data (image), K is the filter/kernel, and i, j are the spatial dimensions of the output feature map. The summation is performed over all spatial locations (m, n) of the input data.

Activation Function: Following the convolution operation, an activation function is applied element-wise to introduce non-linearity into the network. A commonly used activation function is ReLU (Rectified Linear Unit). The formula for ReLU is straightforward: ReLU(x) = max(0, x).

Pooling Operation: Pooling layers decrease the spatial dimensions of feature maps while preserving essential information. Max pooling is a widely used pooling operation that selects the maximum value within a defined window.

Formula for max pooling (assuming a pooling window of size 2x2): $\text{MaxPooling}(i,j) = \max(I(2i,2j), I(2i,2j+1), I(2i+1,2j), I(2i+1,2j+1))$

Fully Connected Layers: Fully connected layers establish connections between every neuron in one layer to every neuron in the subsequent layer, resembling conventional artificial neural networks.

Formula for a fully connected layer (assuming \(x \) is the input vector, \(W \) is the weight matrix, and \(b \) is the bias vector): $z = Wx + b$

$a = activation(z)$

a , output of the layer, and activation is the chosen activation function (e.g., ReLU).

Softmax activation is often used to ensure that the outputs represent probabilities. Formula for softmax activation (assuming zi is the output of neuron i in the output layer):

$$softmax(zi) = e^{zi} / \sum_{j-1}^{N} e^{zj}$$ N is the number of neurons in the output layer.

Training:

CNNs are trained using backpropagation with gradient descent-based optimization algorithms.

The loss function (e.g., categorical cross-entropy) is computed based on the network's output and the ground-truth labels.

Gradients are backpropagated through the network to update the weights and biases.

Formula for updating weights using gradient descent:

$W_{new} = W_{old} - \eta \cdot \partial L / \partial W$

$b_{new} = b_{old} - \eta \cdot \partial L / \partial b$

L represent loss function, η represent learning rate, and $\partial L/\partial W$ and $\partial L/\partial b$ are represent the gradients of the loss function with respect to the weights and biases, respectively.

By training a CNN on features extracted from audio recordings in the four Ethiopian languages, you can build a SLID system capable of accurately identifying spoken Amharic, Oromo, Tigrigna, and Wolayitegna languages.

Long Short-Term Memory (LSTM): LSTMs are a specialized type of recurrent neural network crafted to capture long-term dependencies in sequential data, which makes them well-suited for tasks such as language identification from audio sequences. In SLID, the input sequences would typically be the extracted features (e.g., MFCCs) from the audio recordings.LSTM Cell Operations are consists of several gates and memory cells that control the flow of information

and manage long-term dependencies. The key operations in an LSTM cell are the forget gate, input gate, cell state update, and output gate.

Formulas for LSTM Operations: Let's denote xt as the input at time step t, ht−1 as the previous hidden state, and Ct−1

=1 as the previous cell state.

- ✓ Forget Gate (ft): ft=σ(Wf·[ht−1,xt]+bf) where Wf and bf are represent the weights and biases of the forget gate, respectively. ft, is represent a vector of values between 0 and 1 that determines which information to discard from the cell state.

- ✓ Input Gate (it): it=σ(Wi·[ht−1,xt]+bi) where Wi and bi are represent the weights and biases of the input gate, respectively. And it is a vector of values between 0 and 1 that determines which new information to store in the cell state.

- ✓ Candidate Cell State (~C~t): C~t=tanh(WC·[ht−1,xt]+bC) where WC and bC are represent the weights and biases for the candidate cell state, respectively. And C~t represents the new candidate values that could be added to the cell state.

- ✓ Cell State Update (Ct): Ct=ft⊙Ct−1+it⊙C~t where Ct is represent the updated cell state. And ⊙ represents element-wise multiplication.

- ✓ Output Gate (ot): ot=σ(Wo·[ht−1,xt]+bo) Wo and bo are represent the weights and biases of the output gate, respectively. And ot determines which information from the cell state should be passed to the output.

- ✓ Hidden State (ht): ht=ot⊙tanh(Ct) where ht is represent the output hidden state at time step t. and tanh is the hyperbolic tangent activation function.

These formulas describe the operations within an LSTM cell. By stacking multiple LSTM cells together in a recurrent neural network architecture, LSTMs can successfully capture sequential patterns in the input audio data, making them suitable for spoken language identification tasks.

Bidirectional Long Short-Term Memory (BiLSTM) networks are an extension of LSTM networks that incorporate information from both past and future time steps, making them

effective for capturing context in sequential data. Here's how BLSTM algorithms work and how they can be applied to spoken language identification (SLID) for the four languages (Amharic, Oromo, Tigrigna, and Wolaytaegna), along with their formulas. A Bidirectional LSTM (BLSTM) comprises two LSTM layers: one processes input data in the forward direction, while the other processes input data in the backward direction. At each time step, the outputs from both LSTM layers are concatenated to capture information from both past and future contexts. This bidirectional processing helps in capturing dependencies in both directions, which can be beneficial for SLID tasks. Formulas for BiLSTM Operations: Let's denote xt as the input at time step t, ht−1 as the previous hidden state, and Ct−1 as the previous cell state.

Forward LSTM Operations: The forward LSTM processes the input sequence from left to right.It computes the forward hidden states →ht and forward cell states →Ct using the standard LSTM operations.

Backward LSTM Operations: The backward LSTM processes the input sequence from right to left. It computes the backward hidden states ←ht and backward cell states ←Ct using the standard LSTM operations.

Output Concatenation: At each time step t, the forward and backward hidden states are concatenated to form the final hidden state ht.ht=[ht→,←ht]

BiLSTM Output:The concatenated hidden states ht serve as the output of the BiLSTM network at each time step.These outputs can be further processed or fed into additional layers (e.g., fully connected layers) for classification tasks such as SLID.

BiLSTM Training:

BiLSTM networks are trained using backpropagation through time with gradient descent-based optimization algorithms.

The loss function (e.g., categorical cross-entropy) is computed based on the network's output and the ground-truth labels.

Gradients are backpropagated through both forward and backward LSTM layers to update the network's parameters.

BiLSTM networks are powerful models for sequence processing tasks like SLID because they can effectively capture dependencies in both forward and backward directions. By training a BiLSTM network on features extracted from audio recordings in the four Ethiopian languages, we can build a robust SLID system capable of accurately identifying spoken Amharic, Oromo, Tigrigna, and Wolayitegna languages.

3.4 Model training

The system proposed in this study employs DNN (Deep Neural Network), CNN (Convolutional Neural Network), LSTM(Long Short-Term Memory) and BLSTM(Bidirectional Long Short-Term Memory models). These models utilize MFCC features as inputs and are trained using a dataset of acquired audio recordings.

3.4.1 DNN (Deep Neural Network)

Following data processing, we extract sound data features such as Mel-frequency cepstral coefficients and normalize them to attain zero mean and unit variance. Techniques involved in this process may include introducing background noise, adjusting pitch, or modifying playback speed.

Once the features are extracted and normalized, we split the dataset into training, validation, and test sets. A collective practice is to allocate 70% of the data for training, 15% for validation, and another 15% for testing purposes.

Model Architecture Design: Design DNN architecture suitable for language identification steps:

- ✓ Input Layer: Accepts the preprocessed audio features (50 data samples for each language and each with 3, 10 and 30 seconds length).
- ✓ Hidden Layers: Multiple layers of neurons with activation functions (ReLU).

Output Layer: Produces probabilities of each language class using softmax activation. Experiment with the number of layers, neurons per layer, and activation functions.

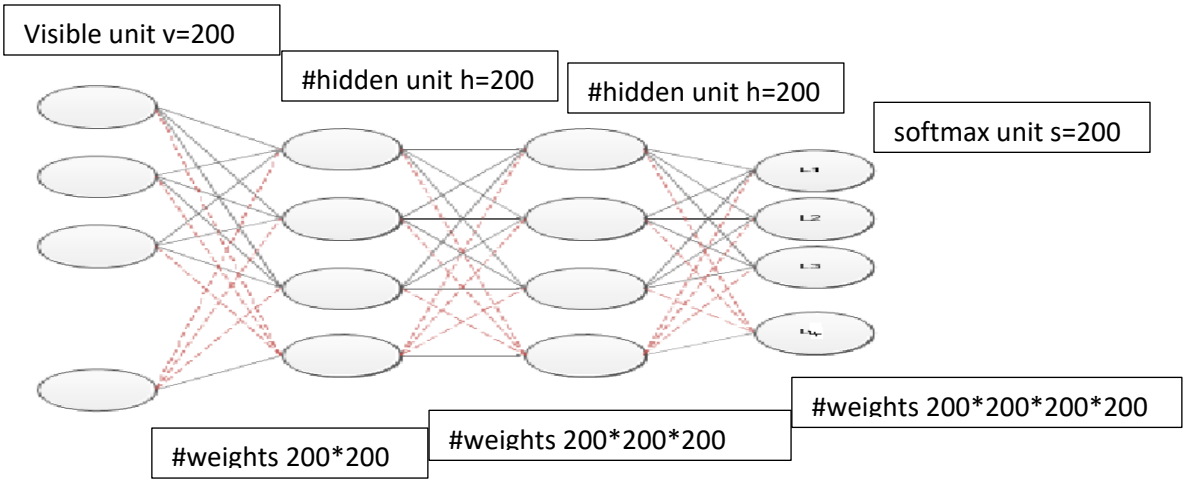


figure 3. 4 Proposed DNN Architecture for LID System

Model Training steps:

- ✓ Initialize the DNN's parameters (weights).
- ✓ Define a loss function, such as categorical cross-entropy, to measure the difference between predicted and true language labels.
- ✓ Choose an optimization algorithm Adam to minimize the loss function.
- ✓ Train the model on the training data by feeding the preprocessed audio features through the network.
- ✓ Monitor the performance on the validation set and adjust hyperparameters accordingly to learning rate or batch size to prevent overfitting.

Hyperparameter Tuning: Experimenting with various hyperparameters to maximize the model's performance on the validation set.

3.4.2 Convolutional Neural Network (CNN)

After preprocess the audio data by converting it into MFCCs (Mel-frequency cepstral coefficients) or other time-frequency representations that capture the linguistic content of the speech. Augment the dataset to increase its variability. Techniques may include adding background noise, changing pitch, or altering speed. After MFCCs made we divide the dataset into training, validation, and test sets. Typically, you might use 70% for training, 15% for validation, and 15% for testing.

Model Architecture Design: Design a CNN architecture suitable for language identification:

- ✓ Input Layer: Accepts the spectrogram images.
- ✓ Convolutional Layers: Apply convolutional filters to extract features from the MFCCs (Mel-frequency cepstral coefficients)
- ✓ Pooling Layers: Down sample the feature maps to reduce dimensionality.
- ✓ Flatten Layer: Flatten the feature maps into a vector.
- ✓ Fully Connected Layers: Process the flattened features to make predictions.
- ✓ Output Layer: Produces probabilities of each language class using softmax activation.
- ✓ Experiment with different architectures, including the number of convolutional layers, filters sizes, and pooling strategies.

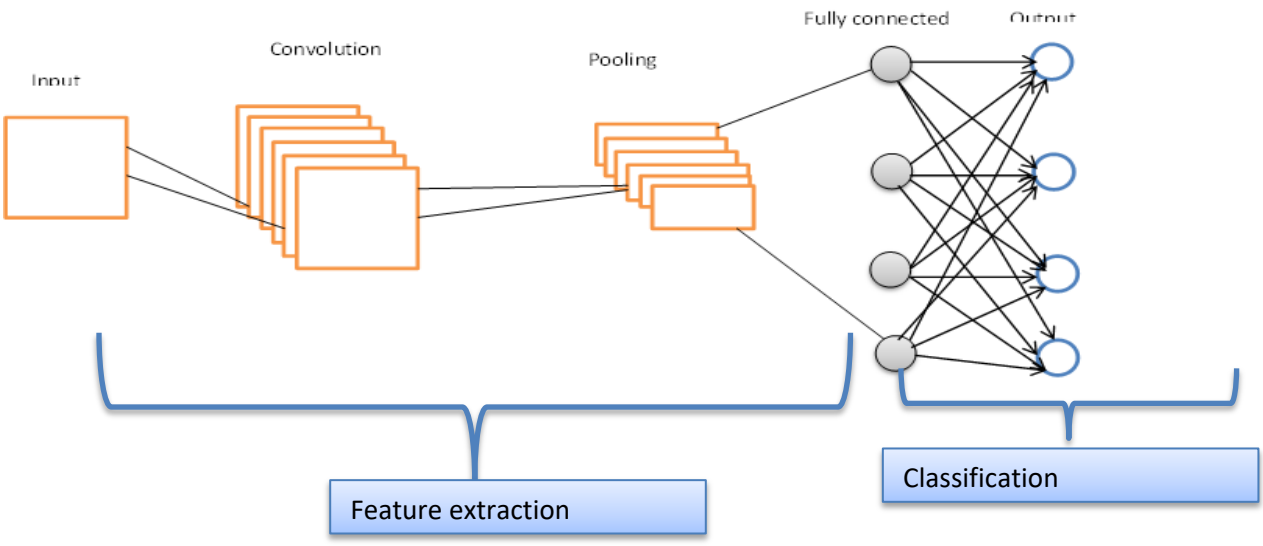


figure 3. 5 Proposed CNN Architecture for LID System

Model Training:

- ✓ Initialize the CNN's parameters (weights).
- ✓ Specify a loss function appropriate for multi-class classification tasks, such as categorical

cross-entropy.

- ✓ Choose an optimal algorithm such as Adam to decrease the loss function.
- ✓ Train the model on the training data by feeding the MFCCs (Mel-frequency cepstral coefficients) images through the network.
- ✓ Continuously monitor the performance on the validation set and adjust hyperparameters as needed to mitigate overfitting.

Hyperparameter Tuning: Experimenting with various hyperparameters to maximize the model's performance on the validation set.

3.4.3 Long Short-Term Memory (LSTM)

Building a language identification system based on LSTM networks involves adapting the LSTM architecture to process sequential data effectively. After gathering a dataset containing audio samples from multiple languages and Preprocess the audio data by converting into MFCCs (Mel-frequency cepstral coefficients),other time-frequency representations that capture the linguistic content of the speech. Augmenting the dataset to increase its variability. Techniques may include adding background noise, changing pitch, or altering speed. After feature extraction MFCCs we divided the dataset into training, validation, and test sets. Typically, you might use 70% for training, 15% for validation, and 15% for testing.

Model Architecture Design

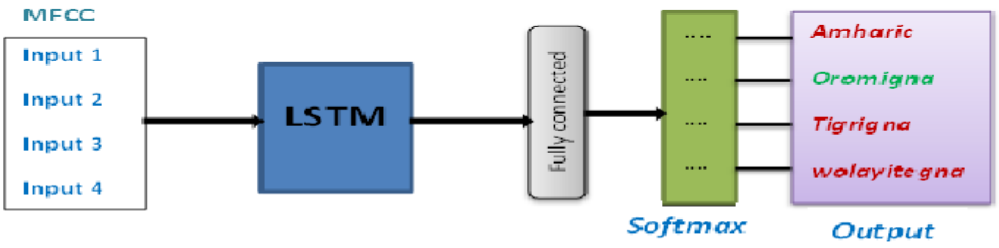


figure 3. 6 Proposed LSTM Architecture for LID System

- ✓ Input Layer: Accepts sequences of features (e.g., MFCC vectors or spectrogram frames).
- ✓ LSTM Layers: To effectively capture temporal dependencies in the data, stack one or more LSTM layers.
- ✓ Optional Bidirectional LSTM: Consider using Bidirectional LSTMs to capture information both past and future contexts.
- ✓ Output Layer: Produces probabilities of each language class using softmax activation.
- ✓ Conduct experiments with various LSTM configurations, including different numbers of LSTM layers, varying numbers of hidden units, and adjusting dropout rates.

Model Training:

- ✓ Initialize the LSTM's parameters (weights).
- ✓ Define a loss function suitable for multi-class classification tasks, such as categorical cross-entropy.
- ✓ Choose an optimal algorithm such as Adam to decrease the loss function.
- ✓ Train the model on the training data by feeding the sequences of features through the network.
- ✓ Continuously monitor the performance on the validation set and adjust hyperparameters as needed to mitigate overfitting.

Hyperparameter Tuning: Experimenting with various hyperparameters to maximize the model's performance on the validation set.

3.4.4 Bidirectional Long Short-Term Memory (BLSTM)

Networks involve leveraging the bidirectional processing capabilities of BLSTMs to capture contextual information from both past and future time steps effectively. After gathering a dataset containing audio samples from multiple languages, Preprocess the audio data by converting into MFCCs (Mel-frequency cepstral coefficients) other time-frequency representations that capture the linguistic content of the speech. Augmenting the dataset to increase its variability. Techniques may include adding background noise, changing pitch, or altering speed. After

feature extraction MFCCs we divided the dataset into training, validation, and test sets. Typically, you might use 70% for training, 15% for validation, and 15% for test.

Model Architecture Design: Design a BLSTM-based architecture suitable for language identification:

- ✓ Input Layer: Accepts sequences of features (e.g., MFCC frames).
- ✓ Bidirectional LSTM Layers: Stack one or more Bidirectional LSTM layers to capture information from both past (Backward) and future (Forward) contexts effectively.
- ✓ Optional Additional Layers: You can add fully connected layers or other types of layers for further processing before the output layer.
- ✓ Output Layer: Produces probabilities of each language class using softmax activation.
- ✓ Experiment with different BLSTM configurations, including the number of BLSTM layers, hidden units, and dropout rates.

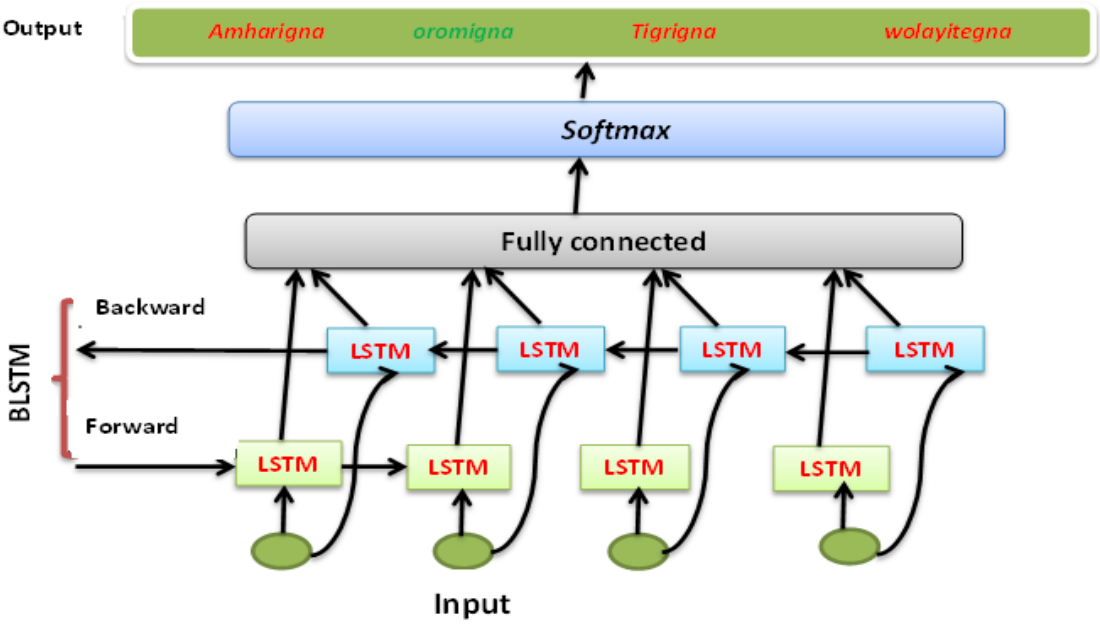


figure 3. 7 Proposed BLSTM Architecture for LID System

Model Training:

- ✓ Initialize the BLSTM's parameters (weights).
- ✓ Specify a loss function appropriate for multi-class classification tasks, such as categorical cross-entropy.
- ✓ Choose an optimal algorithm such as Adam to reduce the loss function.
- ✓ Train the model on the training data by feeding the sequences of features through the network.
- ✓ Monitor the performance on the validation set and adjust hyperparameters accordingly to prevent over fitting.

Hyperparameter Tuning: Experimenting with various hyperparameters to maximize the model's performance on the validation set.

3.5 Evaluation metrics

This study employs a confusion matrix evaluation method to evaluate the classification model's performance and predict outcomes on test data. The experiment is divided into two phases: initial parameter tuning using a validation dataset, followed by final model testing using an independent test set. Upon evaluating the results, an analysis will be conducted to address the research problem.

The dataset consists of four labels representing the languages Amharic, Oromiffa, Tigray, and Wolayta. Therefore, the confusion matrix used pertains to multi-class classification. This method is employed to evaluate models proposed for these four languages. Evaluation metrics such as accuracy, precision, recall, and F1 score are utilized to gauge performance:

- **Accuracy** measures the overall correctness of the model, calculated as (TP + TN) / (TP + TN + FP + FN), where true positives (TP), true negatives (TN), false positives (FP), and false negatives(FN).

- **Precision** measures the fraction of correctly identified positive samples out of all samples identified as positive, calculated as TP / (TP + FP).
- **Recall** measures the fraction of correctly identified positive samples out of all actual positive samples, calculated as TP / (TP + FN).
- **The F1 score** is the harmonic mean of precision and recall, providing a balanced measure that combines both metrics. It is calculated as 2 * (Precision * Recall) / (Precision + Recall).

When assessing a language identification model, these metrics collectively offer a thorough evaluation of its performance, encompassing accuracy, precision, recall, and overall discriminatory capability.

3.6 Rectified Linear Unit (ReLU)

The Rectified Linear Unit activation function is widely adopted in various neural network architectures such as DNNs, CNNs, LSTMs, and BLSTMs due to its straightforward implementation and effectiveness in training deep models. Let's explore its application in each of these architectures for tasks related to spoken language identification:

DNN (Deep Neural Network): In DNNs designed for spoken language identification, ReLU activation is typically applied to the hidden layers of the network [12]. Following the processing of input features through each hidden layer, the resulting outputs undergo element-wise transformation through the ReLU activation function, defined as f(x)=max⁡(0,x)f(x) = \max(0, x)f(x)=max(0,x). This function outputs the input value if it is positive and zero otherwise, thereby introducing non-linearity to allow the network to capture complex patterns and relationships in the data.

CNN (Convolutional Neural Network): In CNNs used for spoken language identification, ReLU activation is frequently employed following each convolutional layer [18]. After applying convolutional filters to the input Mel-frequency cepstral coefficients (MFCC) and performing operations such as pooling, the resultant feature maps undergo ReLU activation. This use of ReLU introduces non-linearity, enabling the CNN to effectively capture intricate patterns and features inherent in the MFCC data.

LSTM (Long Short-Term Memory): In systems based on LSTMs for language identification, ReLU activation is not typically used directly within the LSTM layers themselves. LSTMs typically employ activation functions such as hyperbolic tangent (tanh) or sigmoid within their gates to control the information flow. However, ReLU activation may be employed in feedforward connections or after the LSTM layers to introduce additional non-linearity [11].

BLSTM (Bidirectional Long Short-Term Memory): Similar to LSTMs, ReLU activation is not typically used within BLSTM layers themselves [24]. Instead, ReLU can be applied in feedforward connections or after the BLSTM layers to introduce non-linearity and enhance the network's ability to learn complex relationships.

In summary, the ReLU activation function is commonly utilized in the hidden layers of DNNs and CNNs for spoken language identification tasks. It plays a essential role in introducing non-linearity, enabling these models to effectively learn intricate patterns in the data. While not commonly used within LSTM and BLSTM layers directly, ReLU can still be incorporated elsewhere in the network architecture to enhance its non-linear capabilities.

3.7 Cross-entropy

The cross-entropy loss function finds application across various neural network architectures, including DNNs, CNNs, LSTMs, and BLSTMs, particularly for classification tasks like spoken language identification. Here's how it is utilized in each of these architectures:

DNN (Deep Neural Network): In a DNN designed for spoken language identification, the cross-entropy loss function is computed following the application of softmax activation to the output layer [12]. After processing input features through multiple hidden layers, the final layer of the DNN comprises neurons corresponding to the number of language classes. Softmax activation is applied to these neurons to generate a predicted probability distribution across language classes. The cross-entropy loss function measures the disparity between these predicted probabilities and the actual labels. This loss is computed across the entire training dataset and minimized using optimization techniques like stochastic gradient descent (SGD).

CNN (Convolutional Neural Network): Similarly, in a CNN for spoken language identification, the cross-entropy loss function is computed after applying softmax activation to the output layer. The input spectrogram passes through convolutional and pooling layers to extract features, and the resulting feature map is flattened and served into one or more fully connected layers [23]. Softmax activation at the final layer provides the predicted probability distribution over language classes, with the cross-entropy loss function evaluating the disparity between these predictions and the ground truth labels.

LSTM (Long Short-Term Memory): In systems based on LSTMs for language identification, the cross-entropy loss function is applied in a comparable manner [11]. Sequential input data (e.g., MFCCs) is processed through one or more LSTM layers, and the final LSTM layer produces logits for each time step. Softmax activation converts these logits into a probability distribution across language classes, and the cross-entropy loss function is then computed to assess the agreement between predicted probabilities and actual labels.

BLSTM (Bidirectional Long Short-Term Memory): Similarly, the cross-entropy loss function is employed in BLSTM-based language identification systems [11],[12]. BLSTMs differ in their bidirectional processing of input sequences, where both forward and backward LSTM layers capture information from past and future contexts. Softmax activation is applied to combined outputs from these layers to derive the probability distribution over language classes, with the cross-entropy loss function subsequently measuring the discrepancy between predicted and true labels.

In summary, the cross-entropy loss function operates in the output layer of DNNs, CNNs, LSTMs, and BLSTMs within spoken language identification systems. Its role is crucial in quantifying the difference between predicted and actual probabilities, guiding the optimization process during training to enhance the model's performance.

3.8 Softmax activation

Softmax is a commonly employed activation function in the output layer of neural networks designed for multi-class classification tasks, including those implemented with DNNs, CNNs,

LSTMs, and BLSTMs for spoken language identification. Here's how Softmax is employed in each of these architectures:

- **DNN (Deep Neural Network):** In a DNN designed for spoken language identification, Softmax is typically applied to the output layer. After processing input features through multiple hidden layers, the final layer of the DNN consists of neurons corresponding to the number of target language classes [12]. Softmax activation transforms the raw scores from these neurons into a probability distribution across the language classes. Each neuron's output represents the probability of its corresponding language class.
- **CNN (Convolutional Neural Network):** In a CNN tailored for spoken language identification, Softmax is used in the final fully connected layer. Following feature extraction from the input spectrogram via convolutional and pooling layers [12], the resulting features are flattened and fed into one or more fully connected layers. Softmax activation is applied to the neurons in the final fully connected layer to generate a probability distribution over the language classes.
- **LSTM** In an LSTM-based language identification system, Softmax is applied to the output layer [11]. After processing sequential input data (such as MFCC frames) through one or more LSTM layers, the final LSTM layer produces a vector of logits for each time step. Softmax activation converts these logits into probabilities for each language class. Typically, the probabilities across all time steps are averaged to derive the final prediction.
- **BLSTM** In a BLSTM-based language identification system, Softmax is employed similarly to the LSTM architecture [24]. The distinction lies in the bidirectional processing of input sequences, where both in forward and backward LSTM layers extract information from past and future contexts. Softmax activation is applied to the combined outputs of these layers to produce a probability distribution over the language classes.

In essence, Softmax activation in DNNs, CNNs, LSTMs, and BLSTMs for spoken language identification systems involving Amharigna, Oromigna, Tigregna, and Wolayitegna languages converts raw outputs into probabilities. This enables the models to predict the likelihood of different language classes effectively.

3.9 Overfitting Problem

Deep Neural Networks (DNNs), CNNs, LSTMs, and related architectures introduce layers of abstraction that can potentially lead to overfitting. Overfitting occurs when a model fits the training data closely but fails to generalize well to predict unseen or unknown data accurately. To determine if a model is overfitting, we employ a technique known as cross-validation. This method divides the data into two sets: the training set and the validation set. The training set is utilized to train the model, while the validation set is used solely to evaluate the model's performance.

Metrics calculated on the training set indicate how effectively the model learns during training. Meanwhile, metrics computed on the validation set evaluate the model's generalization ability, specifically its accuracy in predicting new data. Evaluation Metrics such as loss and accuracy are measured on the training set, while val_loss and val_accuracy represent these metrics on the validation set. For example, if a model achieves approximately 86% accuracy on the training set and 84% on the validation set, it is expected to perform around 84% accuracy on new data. However, if the accuracy metric increases while val_accuracy decreases, it indicates that the model is fitting the training set closely but failing to generalize to new data. This scenario suggests overfitting, where the model starts capturing noise rather than meaningful patterns in the data.

3.9.1 Overfitting Solution

To evaluate the issue of overfitting, we employ a regularization practice known as dropout. Dropout involves randomly omitting units or neurons from the hidden layers during the training process [26]. This technique helps mitigate overfitting by reducing the model's reliance on specific neurons, thereby excluding rare dependencies. Dropout regularization has proven highly effective, especially when training Deep Neural Networks with restricted amounts of data [17].

CHAPTER FOUR

4. DISCUSSION AND RESULT

4.1 OVERVIEW

Detecting languages from audio files is essential in applications like speech recognition, language identification, and natural language processing(NLP). On this Thesis, we conducted experiments to assess the performance of various deep learning algorithms for this task using audio data.

Our primary goal was to assess the efficiency of four deep learning algorithms: CNN, DNN, LSTM and Bidirectional Long Short-Term Memory (BiLSTM) networks, specifically for identifying languages from audio files. Additionally, we aimed to compare their performance across different durations of audio segments (3 seconds, 10 seconds, and 30 seconds).

We utilized a dataset comprising audio recordings from four Ethiopian languages: Oromiffa, Amharic, Wolayta, and Tigray. Each language category contained audio files sampled at a standardized rate. Mel-frequency cepstral coefficients were mined from these audio files to represent their spectral features.

Separate models were trained for each algorithm using the extracted MFCC features, varying the duration of audio segments. Models were trained on a subset of the dataset and evaluated on a held-out test set to measure their performance.

The outcomes of our study are crucial for advancing language detection systems and deep learning applications. Understanding how different algorithms perform in language detection tasks helps in selecting appropriate models for practical applications, thereby increasing the accuracy and efficiency of language processing systems.

In the subsequent sections, we will present detailed results from our experiments for each machine learning algorithm individually. We will begin by discussing the performance of CNN, followed by DNN, LSTM, and BLSTM models across 3s, 10s, and 30s audio segments. Each

section will include a comprehensive analysis of results, followed by discussions on their implications.

4.2 Overview of methods

4.2.1 Dataset Collection

Selection Criteria: Choose audio recordings that represent the diversity of the target languages (Oromiffa, Amharic, Wolayta, and Tigray) in terms of speakers, accents, and contexts.

Source: We compiled our own dataset by recording speeches from FM97.1 Ethiopian Radio and downloading videos from youtube.com specifically for conducting the automatic language identification task in this study.

Each language's prepared for about 200 (above 2 h) for training (i.e, 50 data samples for each languages and each with 3, 10, 30 second length) and 20 for testing (i.e., 5 data samples for each language also with 3, 10 and 30 seconds long). The length of the whole training samples is 200 x3 seconds (i.e., 10 minutes), 200 x 10 seconds (i.e., 40 minutes) and 200 x 30 (i.e., 100 minutes =2 hours and 30 minutes long).

4.2.2 Feature Extraction

MFCC Extraction: Utilize feature representations Mel-frequency cepstral coefficients (MFCCs) for the audio data.

Librosa Library: Employ the Librosa library in Python to extract MFCC features from the audio records.

Duration Segmentation: Extract MFCC features for audio segments of different durations, including 3 seconds, 10 seconds, and 30 seconds.

4.2.3 Model Training

- Convolutional Neural Networks (CNN)
    - ✓ Design and train CNN models using MFCC features as input.
    - ✓ Experiment with various CNN architectures, including the number of convolutional layers, filter sizes, and pooling strategies.

- ✓ Implement methods such as batch normalization and dropout to enhance the generalization of the model and mitigate overfitting.

- Deep Neural Networks (DNN)
  - ✓ Construct deep neural network architectures to learn complex patterns from the MFCC features.
  - ✓ Explore different configurations of hidden layers, activation functions, and regularization techniques.
  - ✓ Use optimization algorithms such as Adam to train the DNN models.

- LSTM (Long Short-Term Memory Networks)
  - ✓ Implement LSTM architectures to capture temporal dependencies in the sequential MFCC representations.
  - ✓ Configure the LSTM layers with appropriate hidden units and recurrent dropout to prevent vanishing gradients and overfitting.
  - ✓ Train LSTM networks using backpropagation through time (BPTT) to optimize the model parameters.

- Bidirectional LSTM (BLSTM) Networks
  - ✓ Extend the LSTM models with bidirectional connections to capture information from both past and future contexts.
  - ✓ Stack multiple bidirectional LSTM layers to learn hierarchical representations of the MFCC features.
  - ✓ Apply dropout regularization to the BLSTM layers to improve model robustness and prevent overfitting.

4.2.4 Model Evaluation

Table 4. 1 : standard test accuracy achieved by each model

Model	Test Accuracy
CNN	0.85
DNN	0.82
LSTM	0.87
BLSTM	0.89

Train-Test Split

- ✓ **Purpose:** The dataset was partitioned into a training set for model training and a test set for model evaluation.
- ✓ **Ratio:** The data was divided using a 70:30 ratio, with 70% allocated to the training set, 15% for validation, and 15% for testing.
- ✓ **Randomization:** Before partitioning, the dataset underwent random shuffling to remove any inherent order or biases.

Performance Metrics such as accuracy, precision, recall and F1-score

Cross-Validation

- K-Fold Cross-Validation divides the dataset into k equal-sized folds, where k represents the number of folds. The model is trained iteratively on k-1 folds and validated on the remaining fold.

- Purpose: This technique allows for evaluating the model's performance across various subsets of the data, reducing the risks of overfitting or underfitting to a specific data split.

- Average Metrics: Average accuracy, precision, recall, and F1-score are calculated across all folds to offer a more robust evaluation of the model's performance.

Statistical Analysis

- ✓ Significance Level: A determined significance level (e.g., $\alpha = 0.05$) was set to determine

whether observed differences in performance were statistically significant.

- ✓ Multiple Comparisons Correction: Corrections like Bonferroni or Holm's method were applied when conducting multiple hypothesis tests to control the familywise error rate.

4.2.5 Overview of Results

Based on the provided results, we can observe the demonstration of different models (CNN, DNN, LSTM, and BLSTM) for language detection at different time intervals (3 s, 10 s, and 30 s). The models were trained with a learning rate of 0.0001, a batch size of 32, and the Adam optimizer over 50 epochs. The reported percentages represent the accuracy achieved by each model on the test dataset.

4.3 Implementation

4.3.1 For the 3-second results

Table 4.2 The BLSTM model demonstrated the highest accuracy among the languages tested, specifically for Amharigna, Tigregna, and Wolayitegna, achieving an accuracy rate of 95% over a 3-second duration. Additionally, it achieved an F1-score of 95.7%, with precision at 95.2% and recall at 95%. Following this, the LSTM model performed well for Wolayta002, Tigray004, and Oromiffa, achieving an accuracy of 92.5%. It obtained an F1-score of 93.5% for Wolayta002 and 92.8% for Tigray004 and Oromiffa, with precision scores of 93.1% and 91.6%, and a recall of 95%. The DNN model for Wolayta002 achieved an accuracy of 92.5% and an F1-score of 93.5%, with precision at 93.1% and recall at 95%. Lastly, the CNN model for Tigray004 achieved an accuracy of 92.5%, with an F1-score of 92%, precision at 91%, and recall at 94%.

Model	Language	Learning rate	Batch size	Optimizer	Epoch	Accuracy(%)	loss	F1 score	precision	Recall
CNN	Amharic	0.0001	32	adam	50	85	0.57	84.9	85	89.1
	oromiffa	0.0001	32	adam	50	89.9	0.52	89.8	88.7	92.5
	tigray	0.0001	32	adam	50	92.5	0.49	92	91	94
	wolayta	0.0001	32	adam	50	85	0.6	84.1	84.3	87.5
DNN	amharic	0.0001	32	adam	50	85	0.58	84.8	84.1	86.9
	oromiffa	0.0001	32	adam	50	90	0.42	89.6	89.2	91.3
	tigray	0.0001	32	adam	50	87.5	0.54	86.9	86.3	89.7
	wolayta	0.0001	32	adam	50	92.5	0.39	93.5	93.1	95
LSTM	amharic	0.0001	32	adam	50	90	0.43	91.4	91.4	93.3
	oromiffa	0.0001	32	adam	50	92.5	0.62	92.8	91.6	95
	tigray	0.0001	32	adam	50	92.5	0.45	93.5	93.1	95
	wolayta	0.0001	32	adam	50	92.5	0.47	93.5	93.1	95
BLSTM	amharic	0.0001	32	adam	50	95	0.5	95.7	95.2	96.6
	oromiffa	0.0001	32	adam	50	92.5	0.55	93.5	93.1	95
	tigray	0.0001	32	adam	50	95	0.62	95.7	95.2	96.6
	wolayta	0.0001	32	adam	50	95	0.52	95.7	95.2	96.6

Table 4. 2 The results of 3s test data duration

4.3.2 For the 10-second results

Table 4.3 illustrates that with a duration of 10 seconds, the BLSTM model achieved the highest accuracy among the tested languages, particularly for Oromiffa005, with an accuracy of 95%. It also attained an F1-score of 94.5%, precision of 94.1%, and recall of 95.5%. Following this model is the DNN model across all test languages, achieving an accuracy of 92.5%, an F1-score of 91.6%, precision of 91.2%, and recall of 93%. The CNN model for Tigray005 achieved an accuracy of 90%, with an F1-score of 89.3%, precision of 88.7%, and recall of 91.3%. Lastly, the LSTM model performed well for Tigray005 and Oromiffa005, both achieving an accuracy of 87.5%.

Model	Language	Learning rate	Batch size	Optimizer	Epoch	Accuracy(%)	loss	F1 score	precision	Recall
CNN	amharic	0.0001	32	adam	50	80	0.86	82.5	85.4	85.8
	oromiffa	0.0001	32	adam	50	85	0.42	85.6	85	88
	tigray	0.0001	32	adam	50	90	0.36	89.3	88.7	91.3
	wolayta	0.0001	32	adam	50	80	0.56	79.6	80	83.6
DNN	amharic	0.0001	32	adam	50	92.5	0.3	91.6	91.2	93
	oromiffa	0.0001	32	adam	50	92.5	0.25	91.5	91.2	93
	tigray	0.0001	32	adam	50	92.5	0.25	91.6	91.2	93
	wolayta	0.0001	32	adam	50	92.5	0.24	91.6	91.2	93
LSTM	amharic	0.0001	32	adam	50	80	0.6	79.3	80.3	83
	oromiffa	0.0001	32	adam	50	87.5	0.58	86.5	86.8	89.4
	tigray	0.0001	32	adam	50	87.5	0.56	86.9	86.4	88.6
	wolayta	0.0001	32	adam	50	77.5	0.64	76.5	76.8	79.4
BLSTM	amharic	0.0001	32	adam	50	87.5	0.42	85.2	85.6	87.2
	oromiffa	0.0001	32	adam	50	95	0.24	94.5	94.1	95.5
	tigray	0.0001	32	adam	50	92.5	0.3	91.4	90.9	93
	wolayta	0.0001	32	adam	50	87.5	0.67	87.1	87.4	89.4

Table 4. 3 The results of 10s test data duration

4.3.3 For the 30-second results

Table 4.4 indicates that with a duration of 10 seconds, the BLSTM model achieved the highest accuracy among the tested languages, specifically for Oromiffa003 and Tigray005, achieving an accuracy of 87.5%. It also obtained an F1-score of 82.3% for Oromiffa003 and 81.9% for Tigray005, with precision scores of 85% and 84.8%, and recall scores of 83.4% and 83.2%, respectively. Following closely is the DNN model for Amharic004, with an accuracy of 85% and an F1-score of 86.7%, precision of 88.5%, and recall of 87.5%. The LSTM model for Amharic004 also achieved an accuracy of 85%, with an F1-score of 84.6%, precision of 84.6.7%, and recall of 85%. Lastly, the CNN model for Wolayta002 attained an accuracy of 82.5%, with an F1-score of 81%, precision of 85%, and recall of 77%.

Model	Language	Learning rate	Batch size	Optimizer	Epoch	Accuracy(%)	loss	F1 score	precision	Recall
CNN	amharic	0.0001	32	adam	50	80	0.6	79.2	84.2	79.7
	oromiffa	0.0001	32	adam	50	80	0.57	80.2	83	79
	tigray	0.0001	32	adam	50	75	0.9	74.2	85.9	72.2
	wolayta	0.0001	32	adam	50	82.5	0.7	81	85	77
DNN	amharic	0.0001	32	adam	50	85	0.5	86.7	88.5	87.5
	oromiffa	0.0001	32	adam	50	80	0.56	80	82	83
	tigray	0.0001	32	adam	50	77.5	0.52	77.5	77.9	77.7
	wolayta	0.0001	32	adam	50	82.5	0.5	84.4	85	84.7
LSTM	amharic	0.0001	32	adam	50	85	0.49	84.6	84.6	85
	oromiffa	0.0001	32	adam	50	80	0.6	79	79.8	81
	tigray	0.0001	32	adam	50	77.5	0.63	76	76.4	78.6
	wolayta	0.0001	32	adam	50	80	0.57	79	78	82.6
BLSTM	amharic	0.0001	32	adam	50	80	0.7	75.3	77.7	75.2
	oromiffa	0.0001	32	adam	50	87.5	0.6	81.9	85	83.4
	tigray	0.0001	32	adam	50	87.5	0.62	82.3	84.8	83.2
	wolayta	0.0001	32	adam	50	85	0.58	81.6	82.1	80.9

Table 4. 4 The results of 30s test data duration.

Overall, the BLSTM model consistently achieved the highest average accuracy across all time intervals, followed by the DNN model. The DNN and LSTM models showed competitive

performance but generally achieved slightly lower accuracies. It's important to notice that these results are specific to the given dataset and settings. Further evaluation and experimentation may be necessary to assess the models performance on different datasets and tasks

4.3.4 Comparison of each model

Accuracy			
Model	**3s (%)**	**10s (%)**	**30s (%)**
CNN	92.5	90	82.5
DNN	92.5	92.5	85
LSTMs	92.5	87.5	85
BiLSTM	95	95	87.5

Table 4. 5 Comparison of the accuracy of each model

Prediction accuracy histogram of 3s, 10s and 30 s data samples tested in DNN, CNN, LSTM and BLSTM based LID system

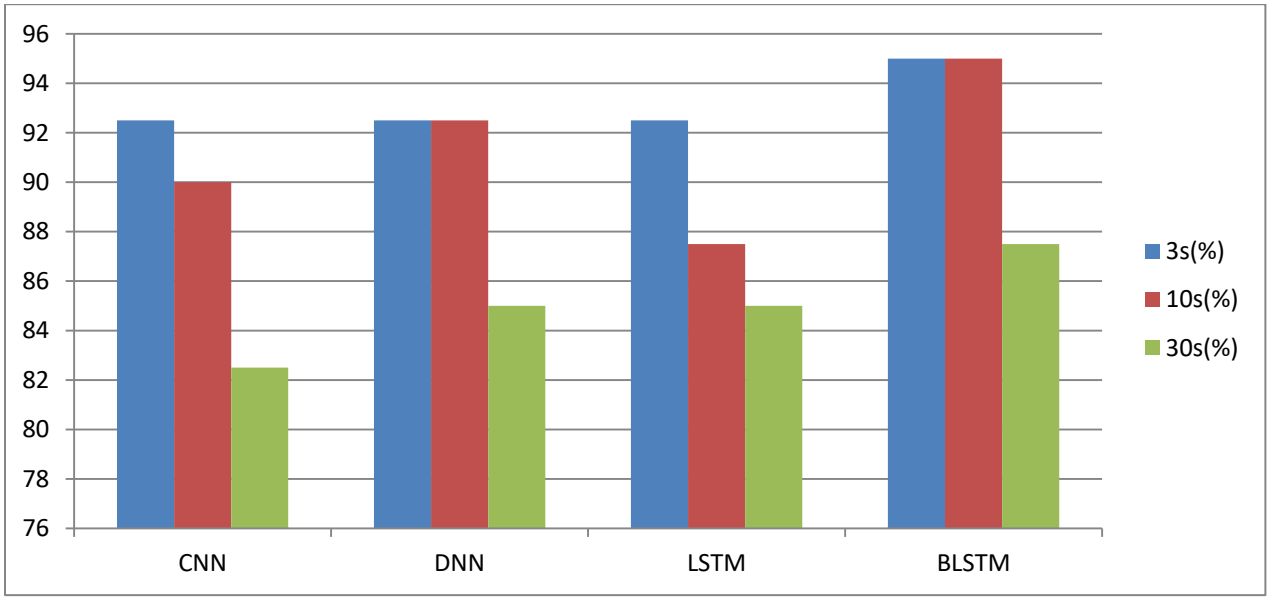


Figure 4. 1 Comparison of CNN, DNN, LSTM and BLSTM using MFCC features based LID systems using prediction accuracy

The Comparison of model CNN, DNN, LSTM and BLSTM in 3, 10 and 30 sec based LID systems using prediction Loss.

Loss				
Model	**3s**	**10s**	**30s**	**Average Loss**
CNN	0.545	0.55	0.6925	0.595833333
DNN	0.4825	0.26	0.52	0.420833333
LSTMs	0.4925	0.595	0.5725	0.553333333
BiLSTM	0.5475	0.4075	0.625	0.526666667

Table 4. 6 comparison of each model loss average values

4.3.5 The confusion matrices of the models proposed

When utilizing the confusions matrices to predict outcomes of each language, the examined instances of confusion that arose. Specifically, we analyzed the proposed confusion matrix of models, focusing on a specific feature: the MFCC (Mel-frequency cepstral coefficients). Section 4.2, Figure 4.2 presents this confusion matrix, evaluating the performance of model's on the test dataset by quantifying correct and incorrect classified test samples within each class.

Figure 4.2 (a, b, c) illustrates the distribution of sample data classified as (TP) True Positive, (FP) False Positive, (TN) True Negative, and (FN) False Negative across the 4 classes. For instance, regarding Amharic as the actual class, the model correctly identified 51 samples as Amharic. However, it misclassified 5 samples as Wolayta and 4 samples as Tigray.

In summary:

- ✓ The diagonal symbols represent correct predictions.
- ✓ Off-diagonal symbols represent misclassifications.

Based on this confusion matrix: The model performed well overall, correctly classifying the majority of samples. However, it made some errors, particularly misclassifying Amharic samples as Wolayta and Tigray.

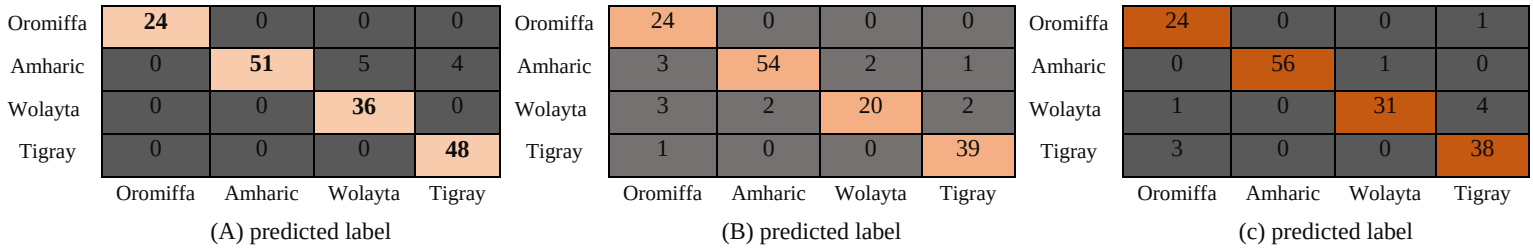


Figure 4. 2 Confusion matrices for the BLSTM model at (a) 3s, (b) 10s, and (c) 30s

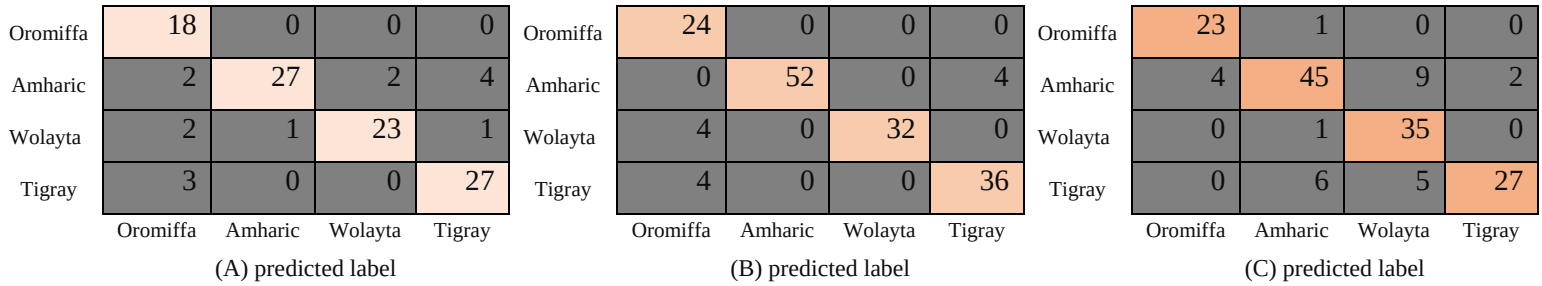


Figure 4. 3 Confusion matrices for the CNN model at (a) 3s, (b) 10s, and (c) 30s

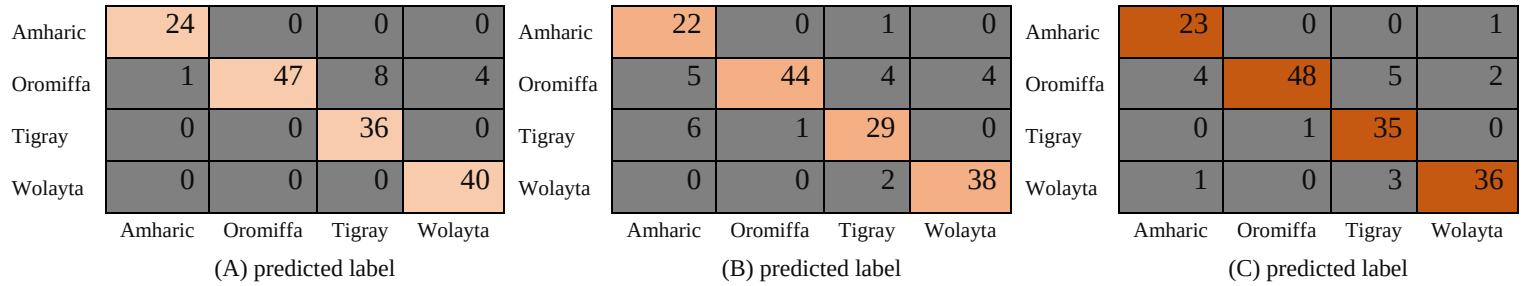


Figure 4. 4 Confusion matrices for the LSTM model at (a) 3s, (b) 10s, and (c) 30s

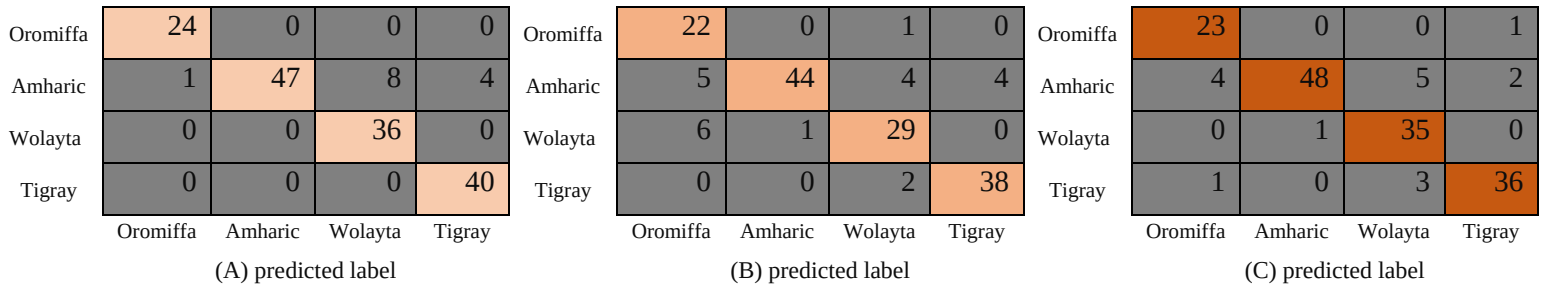


Figure 4. 5 Confusion matrices for the DNN model at (a) 3s, (b) 10s, and (c) 30s

Validation and Training accuracy and also validation loss and training of 3s, 10s and 30 s data samples tested in DNN, CNN, LSTM and BLSTM based LID system.

Additionally, samples from Figures 4.2 to 4.4 provide a perfect illustration of the BLSTM model's accuracy and cross-entropy (loss) performance evaluation throughout both the validation and training stages. At epoch 50, the training accuracy is 95%, with a precision of 95.2% and a recall of 96.5%. Similarly, the BLSTM architecture exhibits validation losses and training of 0.573 and 0.05820, respectively.

4.3.6 Evaluation metrics accuracy and loss of proposed system

4.3.6.1 CNN Evaluation metrics accuracy and loss

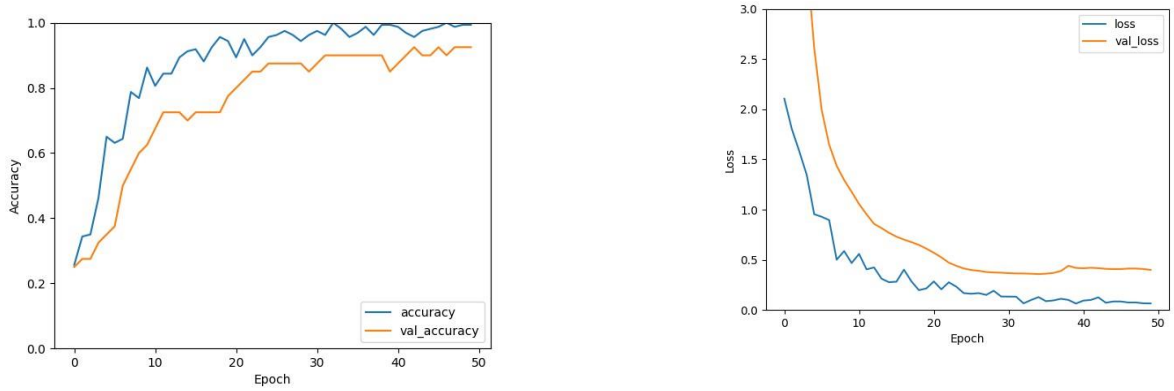


Figure 4.6 CNN model with a 3-second duration: accuracy and loss.

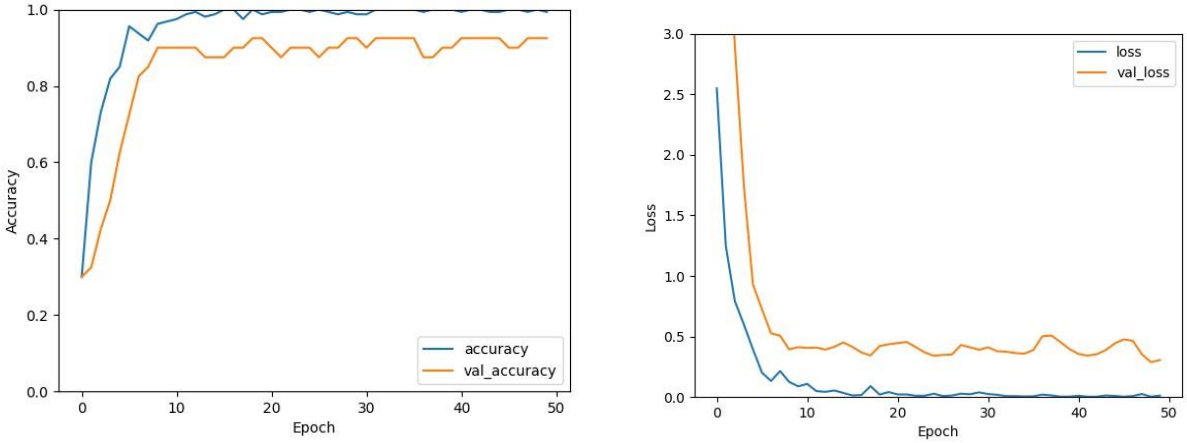


Figure 4.7 CNN model with a 10-second duration: accuracy and loss.

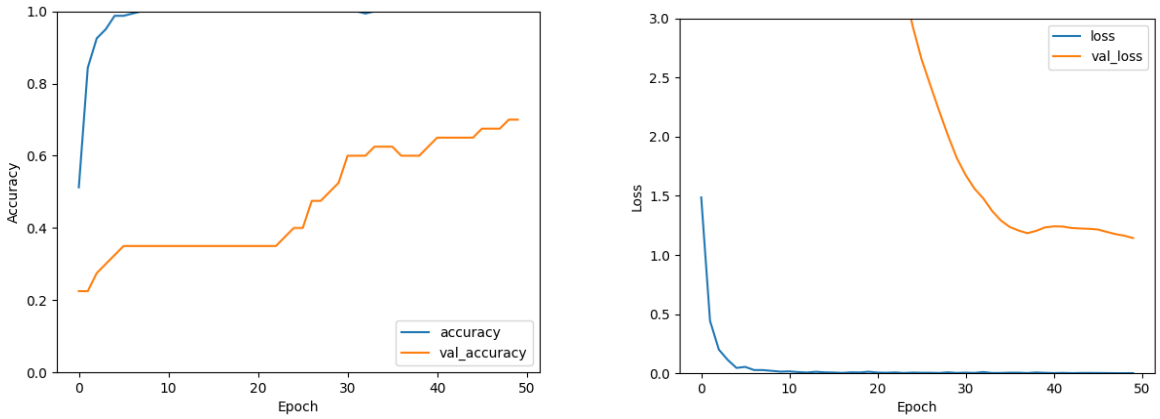


Figure 4.8 CNN model with a 30-second duration: accuracy and loss.

4.3.6.2 DNN Evaluation metrics accuracy and loss

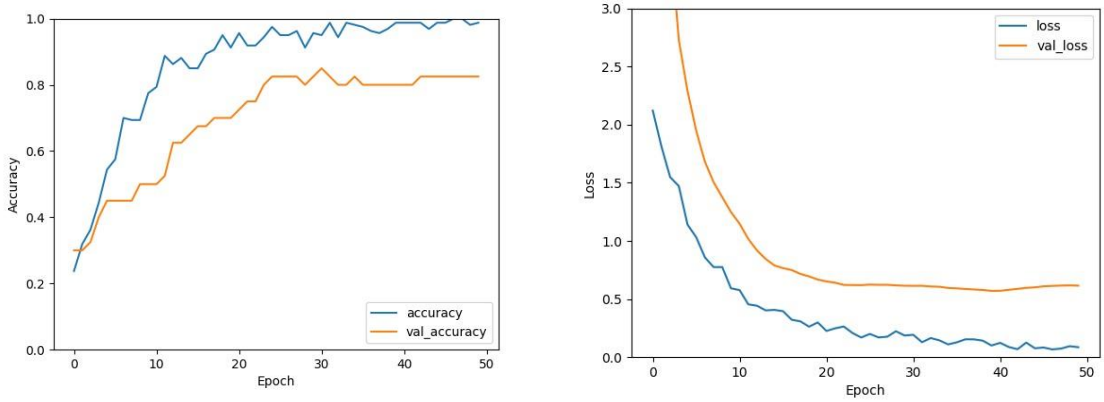


Figure 4.9 DNN model with a 3-second duration: accuracy and loss.

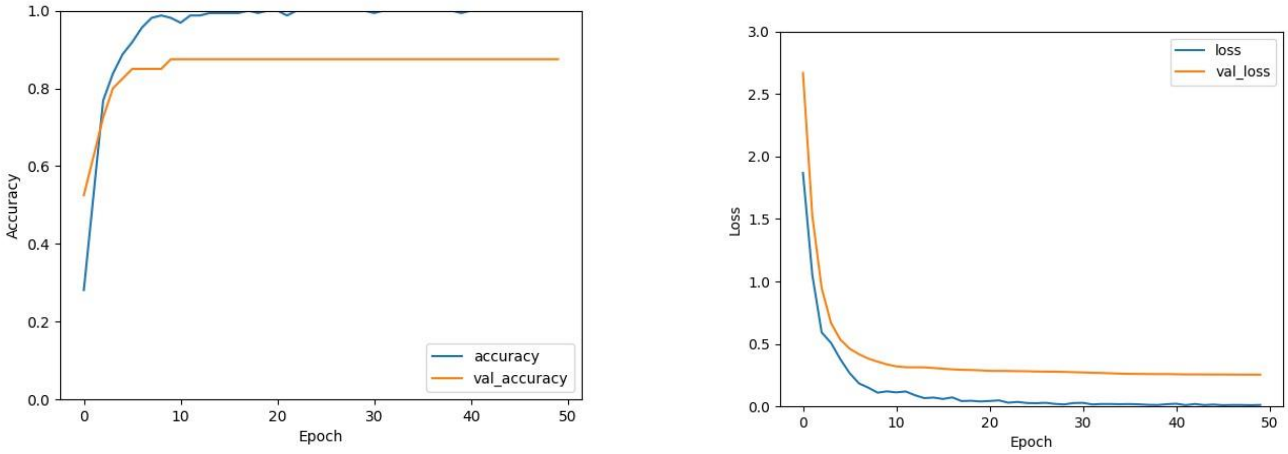


Figure 4.10 CNN model with a 10-second duration: accuracy and loss.

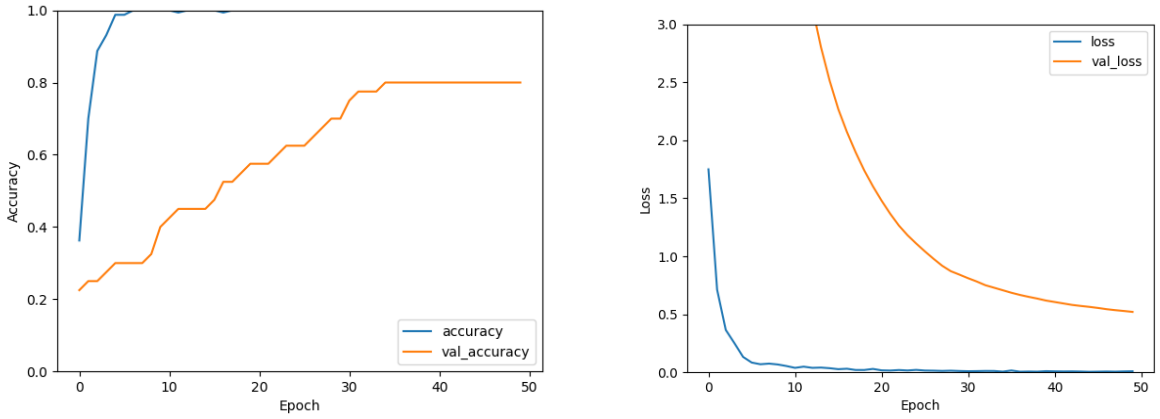


Figure 4.11 DNN model with a 30-second duration: accuracy and loss.

4.3.6.3 LSTM Evaluation metrics accuracy and loss

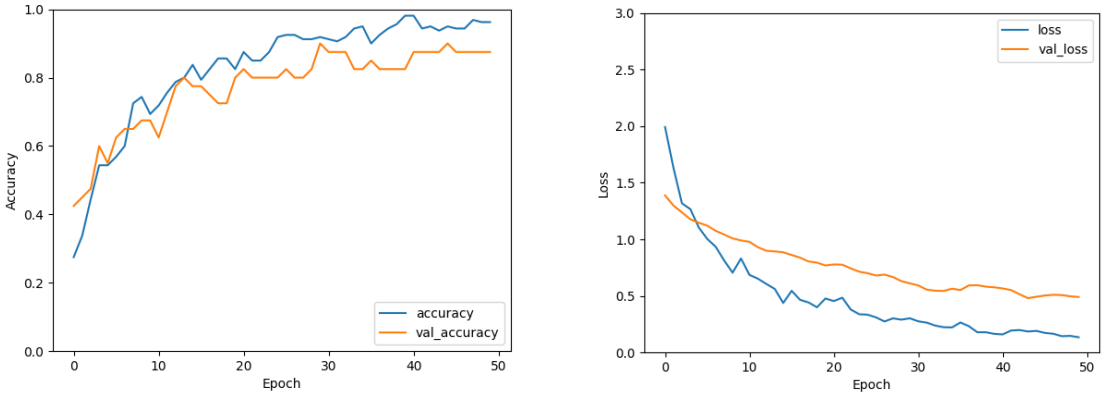


Figure 4.12 LSTM model with a 3-second duration: accuracy and loss.

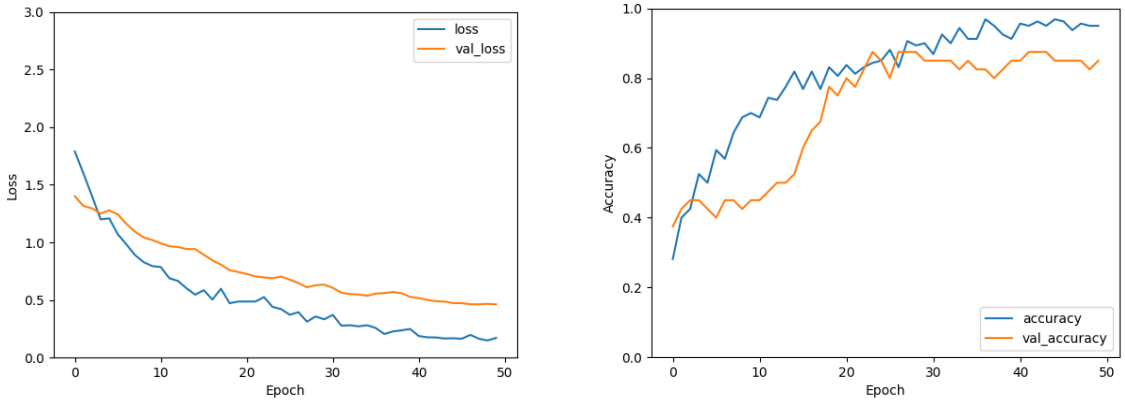


Figure 4.13 LSTM model with a 10-second duration: accuracy and loss.

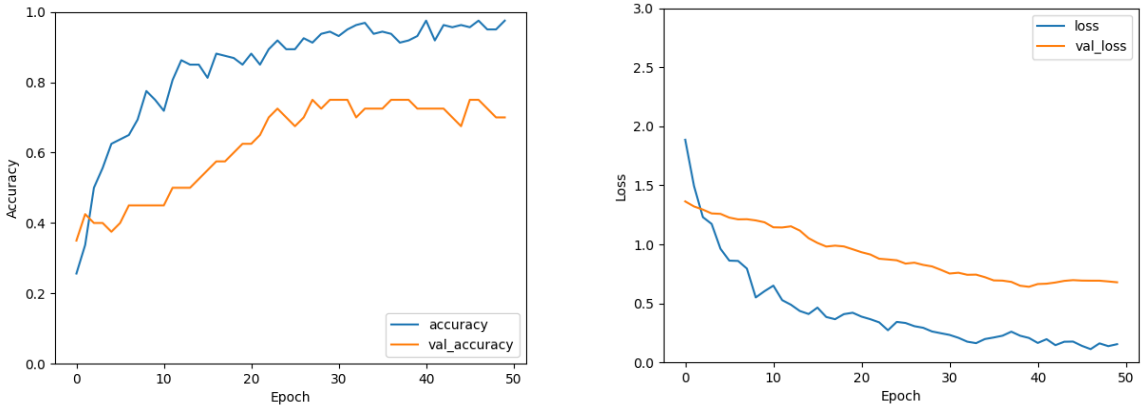


Figure 4.14 LSTM model with a 30-second duration: accuracy and loss.

4.3.6.4 BLSTM Evaluation metrics accuracy and loss

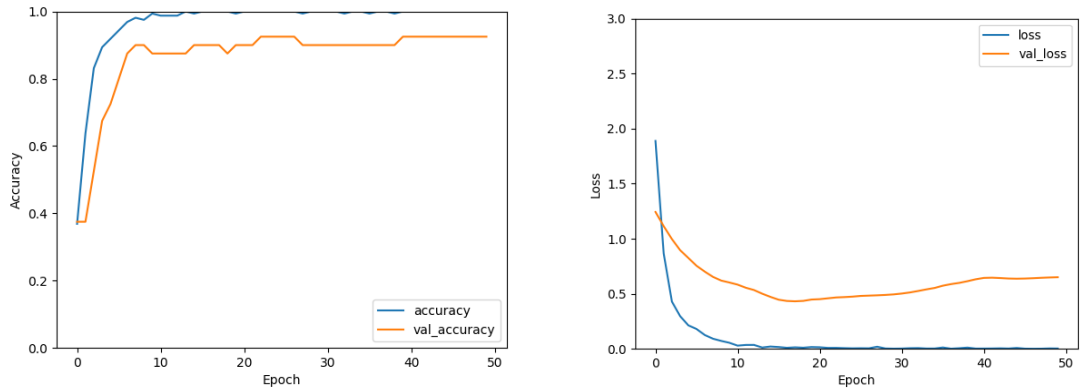


Figure 4.15 BLSTM model with a 3-second duration: accuracy and loss.

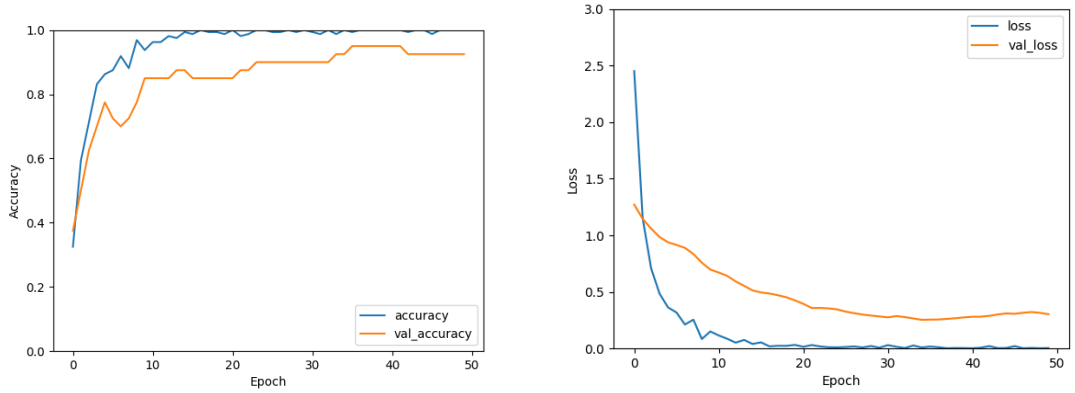


Figure 4.16 BLSTM model with a 10-second duration: accuracy and loss.

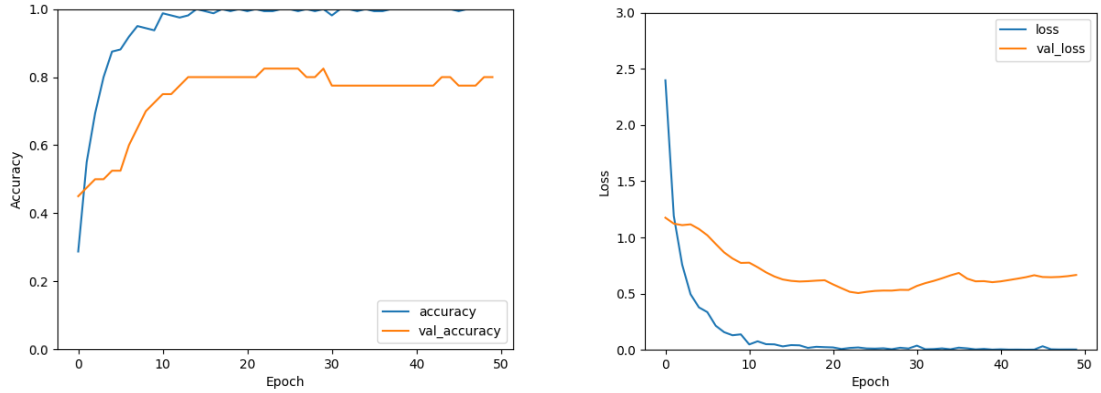


Figure 4.6 BLSTM model with a 30-second duration: accuracy and loss

4.3.7 Average Execution time of proposed system

Time of execution				
Model	**3s(%)**	**10s(%)**	**30s(%)**	**Average time(sec)**
CNN	99.25	169.25	606.75	291.75
DNN	31.645	57.525	123.625	70.93166667
LSTM	69.2	105.175	470.25	214.875
BLSTM	106.4	521.65	1521.25	716.4333333

Table 4.7 the comparison of each model Average execution time (sec)

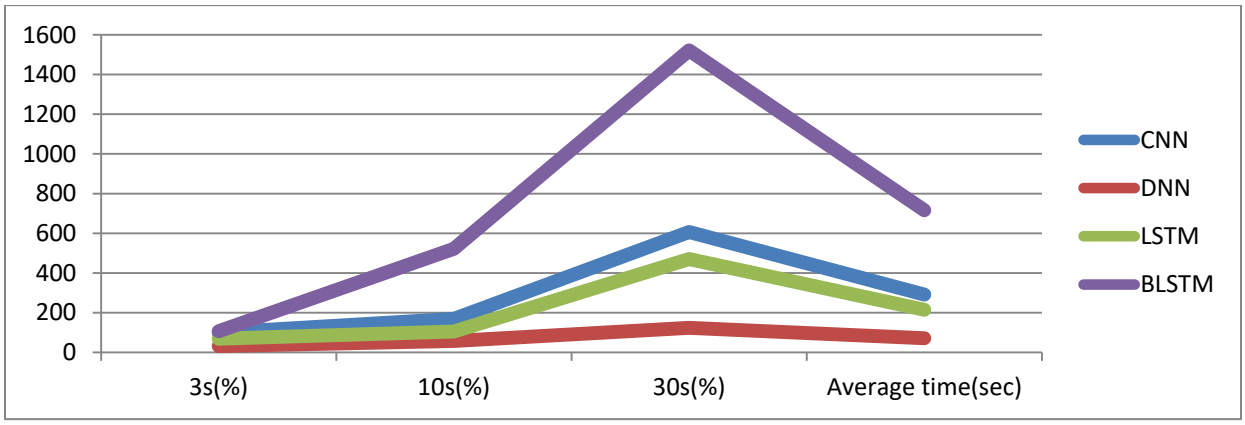


Figure 4. 7 Comparison of each model Average execution time (sec) histogram.

Comparison between the proposed model and existing language identification models

Authors	Models(algorithms)	Features extraction	Performed (Accuracy)
1	GMM	MFCC	91%
11	LSTM and BLSTM	MFCC	93%
18	GMM	MFCC	93%
20	MLC, GC and KNNC	no feature identified	MLC achieves the highest score and GC follows
21	BLSTM	no feature identified	BLSTM achieves the highest score
22	DNN and LSTM	no feature identified	LSTM achieves the highest score
23	CNN and SVM	Spectrogram	CNN classification achieved 93%, svm achieved 82.8%
12	ANN, CNN and LSTM	MFCC	LSTM achieve 88.8%, CNN 88.1% and ANN 87.5
24	The combined 1D CNN layers with BLSTM layers method	Spectrogram	improved best accuracy
Proposed	**DNN, CNN, LSTM and BLSTM**	**MFCC**	**BLSTM achieves 95% the highest score**

Table 4.8 Comparison of the proposed model with existing language identification models.

4.3.8 Effect of Duration

Based on the above results, we can note that the accuracy of the models tends to decrease as the duration of the input increases. This trend is evident when comparing the accuracy at 3 s, 10 s, and 30 s. Generally, the models perform better at shorter durations, with higher accuracy scores. As the duration increases, the models struggle to maintain the same level of accuracy.

4.3.9 Impact of Language

The results also suggest that the impact of languages on model performance varies. Different languages, such as Amharic, Oromiffa, Tigray, and Wolayta, were used in the evaluation. The accuracy scores achieved by the models differ across these languages. For example, the CNN model achieved higher accuracy for Tigray language compared to other languages.

4.3.10 Discussion of Accuracy and Misclassifications

Overall, the models achieved covered accuracy scores for shorter durations, ranging from 75% to

95%. However, as the duration increased, the accuracy dropped, indicating the difficulty of accurately classifying longer speech segments. Misclassifications were likely to occur more frequently for longer durations, leading to lower average accuracy scores. This suggests that the models may have limitations in capturing long-term dependencies and context.

4.3.11 Limitations and Challenges

There are several limitations and challenges associated with the models and the experimental setup. One limitation is the fixed learning rate, batch size, and optimizer used for training. These hyperparameters might not be optimal for all languages and durations, potentially affecting the model performance. Additionally, the small batch size of 32 might limit the model's ability to generalize well. Another challenge is the limited availability of training data, which might delay the models' ability to learn diverse patterns and improve accuracy.

CHAPTER FIVE

5. CONCLUSION and RECOMMENDATION

5.1 Conclusion

In conclusion, we developed a Spoken Language Identification (SLID) system for four Ethiopian languages using DNN, CNN, LSTM, and BLSTM algorithms. Our corpus was prepared specifically for Amharigna, Tigregna, Oromigna, and Wolaytgna due to the absence of a large-scale corpus suitable for SLID purposes. The study aimed to evaluate classification results across different durations 3, 10, and 30 seconds using MFCC feature extraction.

Experimental findings revealed that models performed more effectively with shorter speech durations, with varying accuracy observed across different languages. While achieving decent accuracy scores for shorter durations, maintaining accuracy posed challenges for longer speech segments. Notably, the BLSTM algorithm emerged as the most proper method for the Ethiopian language identification dataset, achieving the highest accuracy scores: 95% for Amharic, Tigregna, and Wolaytgna at 3 seconds, 95% at 10 seconds, and 87.5% at 30 seconds. The DNN model followed closely, achieving maximum accuracy of 92.5% at 10 seconds for all languages.

All experiments were conducted using the Librosa library in Python on a CPU with 1 TB of storage and 8 GB of RAM

5.2 Future works

As future work any excited can improve the research by adding more Ethiopian languages and neighboring country language, selecting the more recent classification for training and increasing the number of features for training and test the system.

To address the limitations and challenges, future research could explore the influence of different hyperparameters for model performance. Increasing the amount of training data, especially for longer durations, could also be beneficial. Additionally, investigating more advanced architectures, such as transformer-based models, could potentially improve accuracy by capturing long-term dependencies more effectively.

5.3. Recommendation

In Ethiopia, there is currently no widely available, publicly accessible corpus of speech in various languages that can serve as a standard for comparing system performance. Therefore, it is recommended to create a shared speech corpus encompassing local languages. This initiative would significantly support research in the field of NLP and enhance ongoing investigations in the region.

References

[1] M. Wondimu, "Signal-Based Ethiopian Languages Identification Using Gaussian

Mixture Model," AAU January, 2017.

[2] Y. Song, B. Jiang, Y. Bao, Si Wei And Li-R. Dai, "i-Vector Representation Based On Bottleneck Features for Language Identification," IEEE ELECTRONICS LETTERS November 2013.

[3] Y. K. Muthusamy, N. Jain, And R. A. Cole, ‘‘Perceptual Benchmarks For Automatic

[4] H. Li, Bin Ma, And K. A. Lee, "Spoken Language Recognition: From Fundamentals to Practice," Proc., IEEE, Vol 101, No.5, May 2013.

[5] T. Young, D. Hazarika, S. Poria, E. Cambria, " Recent Trends In Deep Learning

Based Natural Language Processing," IEEE Comp. Intell … ,August 2018.

[6] R. Gray, "Vector Quantization," IEEE, Acoustic …, 1984.

[7] I. L. Moreno, J.G. Dominguez, D. Martinez, O. Plchot, J.G. Rodriguez, P.J. Moreno, "On the Use of Deep Feedforward Neural Networks For Automatic Language Identification," Brno, 2016.

[8] I. L. Moreno, J.G. Dominguez, D. Martinez, O. Plchot, J.G. Rodriguez, P.J. Moreno, "Automatic Language Identification Using Deep Neural Networks" IEEE International Conference On Acoustic … 2014.

[9] A. Waibel, P. Geutner, L. M. Tomokiyo, T. Schultz, And M. Woszczyna,

[10] A. Revathi, C. Jeyalakshmi, "Comparative Analysis on the Use of Features and

[11] Alemu, A.A., Melese, M.D. & Salau, A.O. Towards audio-based identification of Ethio-Semitic languages using recurrent neural network. Sci Rep 13, 19346 (2023).

[12]. "Language Identification Using Bidirectional LSTM Networks with Spectrograms" by Shukla, Harsh et al. (2020)

[13] C. Madhu, A. George And L. Mary, "Automatic Language Identification for Seven Indian Languages Using Higher Level Features," IEEE Rajiv Gandhi Institute Of Technology(RIT), Kottayam, Kerala, India 2017.

[14] Https://En.Wikipedia.Org/Wiki/Deep_Learning

[15] E. Ambikairajah, H. Li, L. Wang, Bo Yin, And V. Sethu. "Language Identification: A Tutorial." Circuits., IEEE Second Quarter 2011.

[16] R. Saikia, S. R. Singh, P. Sarmah, "Effect of Language Independent Transcribers on

Spoken Language Identification for Different Indian Languages," IEEE 2017.

[17] B. Duvenhage, M. Ntini, P. Ramonyai, "Improved Text Language Identification for the South African Languages," IEEE PRASA-Robmech 2017.

[18]. Pinki Roy, Pradip K. Das, "Language Identification of Indian Languages Based on Gaussian,"

[19]. David Martinez, Lukas Burget, Luciana Ferrer and Nicolas Scheffer, "Ivector-Based Prosodic System for Language Identification," IEEE Conference on ICASSP, 2012.

[20]. V. Ramasubramanian, A. K. V. Sai Jayram and T. V. Sreenivas, "Language Identification Using Paralle

[21]. Kinnunen, Tomi et al. (2017) "Language Identification Using Bidirectional LSTM".

[22]. “Spoken Language Identification Using Deep Neural Networks with Long Short-Term Memory" by Zhang, Xinglei et al. (2019)

[23]. Gundeep Singh, Sahil Sharma, Vijay Kumar, Manjit Kaur, Mohammed Baz, Mehedi Masud, "Spoken Language Identification Using Deep Learning", Computational Intelligence and Neuroscience, vol. 2021, Article ID 5123671, 12 pages, 2021

[24] Shukla, Harsh et al. (2020) [45] , "Spoken Language Identification using bidirectional algorithm"

[25] J. D. Markel, And A. H. Gray, "Linear Prediction of Speech," New York Springer- Verlag,

1976.

[26] Simon Haykin, "Neural Networks and Learning Machines," Book 2008.