Afaan Oromo Interactive Virtual Assistant for Coffee Production Enhancement

Milkesa Abdi Kitila

A Thesis Submitted to the Department of Computer Science and Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Degree of

Master of Science in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

September, 2023

Adama, Ethiopia

Afaan Oromo Interactive Virtual Assistant for Coffee Production Enhancement

Milkesa Abdi Kitila

Advisor(s) Sintayehu Hirpassa (Ph.D.)

A Thesis Submitted to the Department of Computer Science and Engineering

School of Electrical Engineering and Computing

Presented in Partial Fulfillment of the Requirement for the Degree of

Master of Science in computer science and Engineering

Office of Graduate Studies

Adama Science and Technology University

September, 2023

Adama, Ethiopia

Declaration

I hereby declare that this Master Thesis entitled "**Afaan Oromo Interactive Virtual Assistant for Coffee Production Enhancement**" is my original work. That is, it has not been submitted for the award of any academic degree, diploma, or certificate in any other university. All sources of materials that are used for this thesis have been duly acknowledged through citation.

_____	_____	_____
Name of student	Signature	Date

Recommendation of Advisors

I/we, the advisor(s) of this thesis, hereby certify that I/we have read the revised version of the thesis entitled "**Afaan Oromo Interactive Virtual Assistant for Coffee Production Enhancement**" prepared under my/our guidance by **Milkesa Abdi Kitila** submitted in partial fulfillment of the requirements for the degree of Masters of Science in computer science and Engineering. Therefore, I/we recommend the submission of the revised version of the thesis to the department following the applicable procedures.

_____ _____ _____

Advisor Signature Date

Approval Page

I/we, the advisors of the thesis entitled "**Afaan Oromo Interactive Virtual Assistant for Coffee Production Enhancement**" developed by **Milkesa Abdi Kitila** hereby certify that the recommendations and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

Advisor	Signature	Date

We, the undersigned, members of the Board of Examiners of the thesis by **Milkesa Abdi Kitila** have read and evaluated the thesis entitled "**Afaan Oromo Interactive Virtual Assistant for Coffee Production Enhancement**" and examined the candidate during the open defense. This is, therefore, to certify that the thesis is accepted for partial fulfillment of the requirement of the degree of Master of Science in Computer Science and Engineering.

Chairperson	Signature	Date
Internal Examiner	Signature	Date
External Examiner	Signature	Date

Final approval and acceptance of the thesis are contingent upon the submission of its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and School Graduate Committee (SGC).

Department Head	Signature	Date
School Dean	Signature	Date
Office of Postgraduate Studies, Dean	Signature	Date

ACKNOWLEDGMENTS

I wish to express my heartfelt appreciation and profound gratitude to God for bestowing upon me the strength, endurance, and guidance that have been instrumental in my academic journey. I am truly thankful for the blessings and opportunities that have played a pivotal role in shaping my work.

Additionally, my heartfelt appreciation goes to Adama Science and Technology University for providing me the facilities necessary for me to undertake this research. The supportive academic environment and available resources have been instrumental in the successful completion of this study.

I am deeply indebted to my advisor, Dr. Sintayo Hirphassa, for his invaluable mentorship, unwavering support, and insightful guidance. His expertise and dedication have significantly influenced the direction and quality of this thesis.

Lastly, my profound gratitude extends to my family for their unwavering belief in me, unconditional love, and constant encouragement. Their steadfast support has served as the cornerstone of my academic journey.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AI: Artificial Intelligence

AIML: Artificial Intelligence Markup Language

AM: Attention Mechanism

ANN: Artificial Neural Network

API: Application Program Interface

BiGRU: Bidirectional Gated Recurrent Unit

BiLSTM: Bidirectional Long Short-Term

BOW: Bag of Words

CBOW: Continuous Bag of Words

CNN: Convolutional Neural Network

DL: Deep Learning

DNN: Deep Neural Network

GDP: Gross Domestic Product

GRU: Gated Recurrent Unit

GPU: Graphics processing unit

JSON: JavaScript Object Notation

LSTM: Long Short-Term Memory

MIT: Massachusetts Institute of Technology

ML: Machine Learning

MLP: Multi-Layer Perceptron

NLP: Natural Language Processing

NLTK: Natural Language Tool Kit

OOV: Out of Vocabulary

RAM: Random Access Memory

RNN: Recurrent Neural Network

SNNPR: Southern Nations, Nationalities, and People's Region

SOV: Subject Object Verb

SVO: Subject Verb Object

TF-IDF: Term Frequency Inverted Document Frequency

TPU: Tensor Processing Unit

Abstract

Coffee is Ethiopia's top export in terms of revenue, accounting for more than 60% of all foreign exchange earned by the nation. In order to produce quality coffee crops, follow-up is necessary particularly during production processes and collection. Traditionally, agricultural professionals help farmers with the best coffee-production's agricultural practices. The development of text-based chatbots for agriculture has made significant strides in assisting farmers with valuable information and solutions. These systems have leveraged various deep learning techniques and data preprocessing methods to achieve high accuracy rates. However, there is room for improvement, particularly in handling new queries, incorporating more diverse datasets, and enhancing the chatbot's natural language understanding capabilities. Further research and development in this field hold great potential for improving agricultural productivity and supporting farmers in their endeavors. This study proposed a chatbot system that addresses this issue by allowing machines to provide virtual assistance for users via messaging platforms in Afaan Oromo Language. The dataset for this study was collected from the agriculture bureau, a book titled "Teeknoloojii oomisha, qulqullinaa fi walitti hidhamiinsa gabaa", and farmers. The study proposed four models: BiLSTM, BiGRU, CNN-BiLSTM and CNN-BiGRU-AM and two feature extraction techniques: word2vec and fasttext were used. Hyperparameter tuning was conducted using grid search to optimize model performance, considering factors like learning rate, batch size, and network architecture. CNN-BiGRU-AM outperformed other models on our dataset achieving accuracy of 91.01% with word2vec and 89.47% with fasttext. Human evaluation played a vital role in understanding user-centric aspects of chatbot performance, including response time, user-friendliness, usability, completeness, relevance, and overall user experience. The proposed model achieved an average of 86.88% acceptance. Flask framework and telegram bot API were used to create a prototype of the model.

***Key words**: Afaan oromo, Artificial Intelligence; chatbot system; Agriculture; coffee production, BiLSTM, BIGRU, CNN-BiLSTM, CNN-BiGRU-AM, word2vec fasttext*

CHAPTER ONE

1. Introduction

1.1 Background of the Study

In Ethiopia, agriculture plays a pivotal role in the nation's economy, contributing to 40% of the Gross Domestic Product (GDP), 80% of total exports, and employing roughly 75% of the country's labor force(Zerssa et al., 2021). So, an increase in agricultural production in quality and amount plays a key role to improve income and earn valuable foreign exchange. Coffee is Ethiopia's primary export commodity, responsible for generating over 60% of the country's foreign currency earnings(Coffee: The Backbone of Ethiopian Economy, 2015). It is expected that the production of coffee will lead to growth and prosperity, both in terms of quantity and quality. However, in order to produce high-quality coffee crops, it is important to closely monitor the production process and collection methods (Sarirahayu & Aprianingsih, 2018). These production processes are like the selection of coffee seeds, weed control, disease management, pruning, choice of shade trees, season to plant coffee seeds and water them, how to treat seedlings, time to transform them, how to prepare fields to plant coffee, when to collect and how to collect and etc. Post-collection treatments are also important in the production of high-quality coffee crops. Easy accessibility of information about these issues helps farmers to increase their coffee crop yields.

The origins of coffee can be traced back to Ethiopia, where the first coffee plant, known as coffee arabica, naturally thrives in the forested highlands. These highlands are situated at altitudes between 1100 to 2200 meters and experience annual rainfall between 1500 to 2500 mm. The coffee plants grow in soil with a slightly acidic pH ranging from 4.5 to 6.5, marking the historical beginnings of coffee. Coffee is grown practically everywhere in Ethiopia. At present, the primary coffee-producing regions include Kaffa, Illubabor, Wollega, Gimbi, Sidamo, Gedeo, YirgaCheffe, and Harrarge (comprising three zones). The majority of coffee production is concentrated in the Oromia and the Southern Nations, Nationalities, and People's Region (SNNPR), with Oromia alone accounting for 63.3% of the total coffee production(*Coffee: The Backbone of Ethiopian Economy*, 2015). The production of coffee is not like sprinkling and hoping it would grow; it requires prompt attention and follow-up from

the farmers. However, farmers need the assistance of agricultural experts on the treatment needed in the production processes of coffee crops. In our country, agricultural experts have been providing assistance to the farmers with the production processes of coffee crops. However, delivering timely assistance and up-to-date information to farmers through human labor has proven challenging. Therefore, there is a need for a conversational system that is to alleviate these difficulties. we proposed a chatbot model, one application of artificial intelligence (AI) technology that enables machines to communicate with users in natural language via messaging applications which will come up with a solution to this problem.

A chatbot system is an AI conversational software program that enables communication to take place between human users and machines (computers) in natural language via the user interface. The first chatbot, Eliza, was created in 1966 by the Artificial Intelligence Laboratory at MIT(ZEMČÍK, 2019). Although its communication abilities were restricted, Eliza served as an inspiration for subsequent chatbot developments. The widespread adoption of chatbot platforms received a significant boost when major industry players such as Facebook, Skype, WeChat, and others introduced their own chatbot offerings. Nowadays thanks to the advancement of machine learning and natural language processing, there are various chatbot systems like Apple's Siri, Amazon's Alexa, Cortana, and Microsoft's XiaoIce. Chatbots offer numerous advantages, with their most prominent attribute being their ability to provide swift and reliable responses and solutions. These qualities have positioned chatbots as a valuable tool in various domains, leading to rapid advancements in their utilization across sectors such as agriculture, business, e-commerce, healthcare, education, and entertainment.

The aim of this study is to create an Afaan Oromo text-based chatbot model that responds to farmers' queries related to the production processes of the coffee crop. It enables conversation in Afaan Oromo Language between farmers and the chatbot model on coffee production's best agricultural practices. Coffee production processes include preparing seeds, planting seeds, watering the coffee plant, preparing the field, transforming to the field, collecting, post-collection treatment, etc. Easily accessibility of coffee production-related information will support farmers in making decisions quickly on their coffee production practice which may result in increased income for the farmers and the country. The farmers send their questions to the chatbot model in the Afaan Oromo Language and the chatbot provides the answer in the Afaan Oromo Language. One major advantage of developing a chatbot is its availability 24/7

to provide service for the users. In this study, we have built Afaan Oromo text-based chatbot using the power of AI applications that provide assistance for the users on the production of coffee.

This paper is divided into six chapters. Chapter 2 analyzes pertinent works on the chatbot model, reviews earlier work on the chatbot, and identifies gaps in the chatbot model particular to Afaan Oromo. Chapter 3 outlines the selection of assessment criteria, concentrates on the procedure for gathering datasets, and discusses NLP preprocessing, the proposed models, and feature extractions. In Chapter 4, the suggested model architecture for Afaan Oromo coffee production assistant chatbot model is explained in depth, along with the model's implementation. Results, evaluation techniques, model comparisons for each model, and discussions are presented in Chapter 5. The work is finally concluded with possible future research directions and makes recommendations for improvement.

1.2 Motivation for the Study

Agriculture supplies essential resources for mankind and currently serves as the foundational material for the process of industrialization (Role of Agriculture in the Economy - QS Study, 2010). Ethiopia has historically been centered around agriculture, with agriculture serving as the fundamental cornerstone of its economy(Ganati, 2014). Within the agricultural sector, the coffee industry takes the forefront in terms of its contribution to the national economy as a whole, especially the export sector. It generates a substantial 60% of export revenue, comprises 10% of the country's overall agricultural output, and consistently contributes an average of 5% to GDP(*Coffee: The Backbone of Ethiopian Economy*, 2015). So, we have been motivated to improve the coffee crop yield, which will result in an increase in the economy of the nation. We have proposed a chatbot model that will assist farmers in the production processes of the coffee crop. Chatbot has been decreasing the effort and time spent on providing services to the customers and doing the given task. Chatbots are one application of AI, that provides replies to user inquiries in any specific domain in which they are employed. Using the advantages of chatbot technology, we are motivated to solve the problem farmers are facing in the production of coffee crops because coffee production needs timely treatment to give expected coffee crop yield and farmers require on-time assistance on the treatment required in the production of coffee. Since coffee is the main income of the country, engaging

technology to improve its output will improve the country’s economy. So, another reason why we were motivated to do this study is, we want to contribute to improving the economic benefits of coffee using AI technological advantage.

1.3 Statement of the Problem

In most developing countries, agriculture has been serving as a consistent source of inputs for major industries. Agriculture currently dominates Ethiopia's economy and employment; however, non-agricultural sectors grow more rapidly than the agricultural sector(Agriculture and Food Security | Ethiopia | U.S. Agency for International Development,2022). One factor in to slow growth of this sector is the lack of engaging technology in this sector. Numerous research efforts have been conducted to address the issue of delivering agricultural information to farmers across various sectors. However, many of these studies have limitations. Most of them can only generate responses when farmers input queries exactly as they appear in the database, making it challenging to provide context-based responses aligned with the farmers' intentions. Additionally, these studies are predominantly carried out in languages that differ morphologically from Afaan Oromo. Only one study has been conducted in Afaan Oromo, focusing on maize production, and it is specifically trained on a maize production dataset. Consequently, it cannot support farmers in the context of coffee production. In Ethiopia, coffee leads the agricultural sector. Increasing the productivity of the coffee crop has great value in increasing the overall economy of the country. To produce quality and quantity coffee crops, farmers need assistance from coffee production agricultural experts on how to enhance their coffee crop yields. The end production yield of coffee is affected by the treatment and follow-up farmers do from seedling to drying and storing. Farmers access information on best agriculture practices for coffee production through agricultural experts who continuously follow up with farmers and give them advice. Usually, government employees (agricultural extensions) and field officers visit the fields, interact with villagers' farmers, and give them training on the finest farming techniques of coffee. Consequently, a significant number of farmers face challenges in accessing information efficiently, leading to time and financial resources being wasted when seeking information or reaching out to authorities. Improved accessibility to information on coffee cultivation in the farmers' native language would empower them to make more informed decisions regarding their coffee production practices. This study proposed a chatbot model for coffee production processes in the Afaan Oromo

Language, which will act as an agricultural expert to give assistance to the farmers in their agricultural practices for coffee production.

1.4 Research Questions

The main focus of this study revolved around the following three research questions.

RQ1: What are the essential agricultural practices to enhance the production of coffee crops?

RQ2: Which deep learning algorithms can be applicable to build model that provide assistance on the coffee production practices?

RQ3: To what extent can we enhance the overall performance of coffee production virtual assistant model?

1.5 General and Specific Objectives

1.5.1 General Objective

The general objective of this study is to build Afaan Oromo interactive model that provides virtual assistance for the farmers on coffee production.

1.5.2 Specific Objectives

The subsequent lists comprise specific objectives aimed at attaining the study's general objective

- ✓ Investigate the literature on the concepts of agricultural practices on coffee production, modern chatbot development, and the implication of chatbot systems for the agriculture sector
- ✓ Gather the dataset associated with coffee production’s agricultural practices and prepare a dataset for model development
- ✓ Identify and select appropriate models for building a chatbot model for the proposed study
- ✓ Build the model and seek the optimal hyperparameters for enhancing the model's performance
- ✓ Compare the deep learning models’ performances on the dataset and select the outperformed model on our dataset

- ✓ Assess the model's performance using machine learning evaluation metrics

1.6 Significance of the Study

A significant portion of the national economy depends on agricultural goods, with most industries relying on these products as essential inputs. This underscores the importance of fostering the growth of the agricultural sector for the advancement of other industries. Utilizing technological advances in agricultural practice will enhance agricultural output, which will enhance other industries that use the agricultural output as an input. The production of coffee, which dominates Ethiopia's agricultural industry, is one major agricultural sector of focus. The purpose of this study is, to create a chatbot model that can converse with farmers about coffee production procedures in Afaan Oromo. The proposed model in this study provides information on coffee production's best agricultural practices and it is more easily accessible so that farmers can receive assistance to increase the yield of their coffee crops.

1.7 Expected Outcome of the Study

This study aimed to support the agriculture sector in the production of coffee by creating an easy way of accessing information on coffee production processes so that farmers are able to improve their coffee crop yield. This chatbot will act as a virtual agricultural expert to answer queries from farmers on coffee production. Farmers give their request to the chatbot as a question and it responds with agricultural experts' assistance as an answer. Farmers get assistance from this chatbot in Afaan Oromo language.

1.8 Scope of the Study

The scope of this study is developing a text-based chatbot that can interact with users specifically on coffee production enhancement. Farming practices to enhance coffee production involve having information on how to prepare seedlings, treatments of coffee sprouts, climate requirement, rainfall, soil preparation, field design and preparation, when and planting, fertilization use, weed control, pest control, disease control, harvest and dry and pruning(Wolde et al., 2017). Accessibility of this information affects the productivity of coffee crops. This study focuses on developing an interactive model that assists farmers by providing information about their requests on the production processes of coffee. Since coffee is the backbone of the Ethiopian country's economy, focusing on the improvement in the production

of coffee both in quality and quantity will result in the overall development of the economy. The chatbot communicates with users in the Afaan Oromo language, which is widely spoken among the Ethiopian population. This study's objective is to create a text-based chatbot model in Afaan Oromo to assist with coffee production best agricultural practices.

CHAPTER TWO

2. Literature Review and Related Work

2.1 Background of Chatbot Technology

The creation and examination of smart hardware and software, referred to as intelligent agents, constitute a pivotal element of artificial intelligence (AI), which is increasingly becoming intertwined with our everyday lives(Khanna et al., 2015). An instance of an AI system is a chatbot, which is a computer program capable of comprehending one or multiple human languages through natural language processing (NLP) and providing text or spoken responses in conversations with humans(Bansal et al., 2018). The term "chatbot" is interchangeable with other names such as "smart bots," "interactive agents," "digital assistants," or "artificial conversation entities."

The concept of chatbots gained prominence in 1950 when Alan Turing introduced the Turing Test as part of his paper "Computing Machinery and Intelligence," which was published in the journal Mind. This test aimed to answer the question, "Can machines think?"(TURING, 1950). As per the Turing Test, a computer successfully passes the test when a group of humans engaging in a conversation with an unidentified entity perceives the entity as human, despite it actually being a computer. However, there are challenges that limit the test from being a perfect measure of intelligence. The main focus of this test is simulations of human beings' conversational skills, but not on the higher goal of AI research, which is to solve problems. Eliza is the first known chatbot developed in 1966 and was designed to function as a virtual therapist, offering responses to user inquiries presented in the form of questions (Adamopoulou & Moussiades, 2020). In ELIZA pattern matching and the template-based mechanism were used to respond to the queries. At the time because people had no experience of talking with computers, it was enough to confuse people however, its ability in conversation was poor. The development of ELIZA laid the basic foundation for the development of later chatbots. In 1972 a chatbot improvement over ELIZA named PARRY was created(Balaji & Yuvaraj, 2019). In contrast to ELIZA, PARRY, often referred to as "ELIZA with Attitude," employed a conversational approach that was notably more sophisticated and earnest. PARRY achieved the distinction of being the first chatbot to successfully navigate a variation of the

Turing test. With increased interest in using chatbots improvements continued on chatbots. The chatbot ALICE was created in 1995 and in 2000, 2001, and 2004 it earned the Loebner Prize, which is awarded for passing the Turing Test every year. In 2001, a chatbot called SmarterChild was created and made accessible through message services(Singh & Thakur, 2020).

These days chatbots have become most popular in many areas because of their success story and productivity. Chatbots are available throughout the day and provide the services they are created to give. Currently, there are virtual assistance chatbots like Apple Siri, Microsoft Cortana, Amazon, Alexa, Google Assistant, and IBM Watson are accessible via messaging apps such as FB Messenger or WeChat. The need for chatbots increased in many companies to support their routine operations.

2.2 Types of Chatbot

Chatbots can be categorized using different criteria, including their knowledge domain, the services they provide, their objectives, and their design approach(Adamopoulou & Moussiades, 2020).

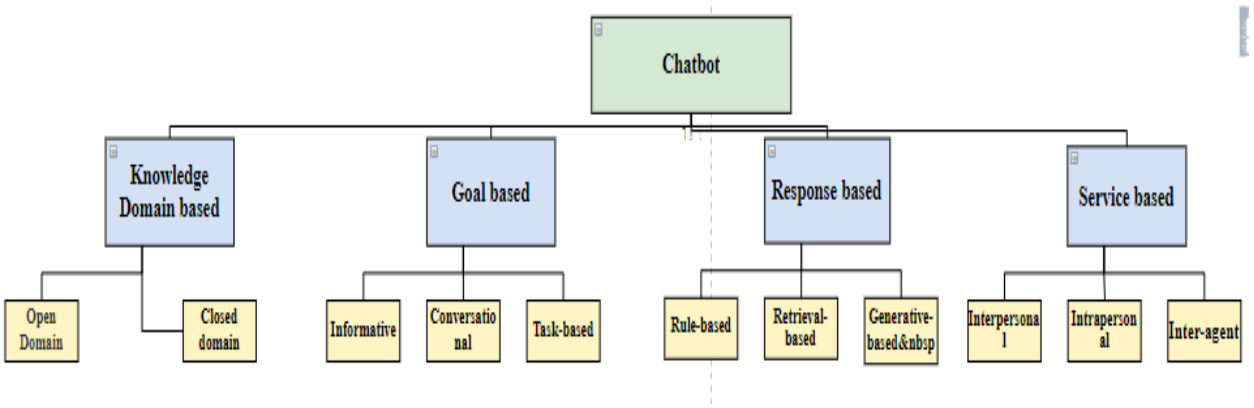

Figure 2.1 Types of Chatbot

When categorizing based on the knowledge domain, the classification considers a chatbot's information accessibility or the extent of its training data. Open-domain chatbots can take part in a conversation on any agenda more like humans do, while closed-domain chatbots provide responses on a particular task(Adiwardana et al., 2020). The amount of training data needed to develop an open-domain chatbot makes the implementation process more difficult and

complex. In this study, we have proposed a chatbot for coffee production assistance, which can be categorized under closed domain chatbot.

The main objective that chatbots try to accomplish is another parameter to categorize chatbots into informative chatbots, conversational chatbots, and task-based chatbots (Adamopoulou & Moussiades, 2020). Informative chatbots are designed to provide users with access to pre-stored information or information from a fixed source. Unlike informative chatbots, conversational chatbots engage in dialogues with users with the aim of providing accurate responses to the phrases they are given. On the other hand, task-based chatbots are designed to perform specific tasks and demonstrate intelligence in seeking information and understanding user input (Adamopoulou & Moussiades, 2020).

The way by which chatbots generate responses defines their model, categorizing them into rule-based, retrieval-based, and generative-based models. In rule-based models, chatbots select responses based on a predefined set of rules, relying on lexical analysis of the input text, and do not generate entirely new textual responses. The latter two models are neural network-based techniques and they became possible with the development of deep learning(Agarwal & Wadhwa, 2020). The response from a neural network-based chatbot can be relevant and grammatically sound because the data used to train the neural network is large. Deep learning enables this type of chatbot to accept inputs in different formats such as text, images, or speech formats. In retrieval-based approaches, responses are created by choosing the most relevant response, which can be achieved through methods like scoring functions using neural networks to compute conditional probabilities or by evaluating the context's connection to potential responses in a co-ranking manner with reinforcement. Conversely, in the Generative Method, responses are generated word by word, with each word corresponding to the input after calculating probabilities across the entire vocabulary. Finally, depending on the service it offers, a chatbot might be classed as interpersonal, intrapersonal, or inter-agent(Adamopoulou & Moussiades, 2020).

2.3 Application Domains of Chatbots

The advancement in the evolution of AI technology attracts companies to include these technologies to support them in providing better products or services. Applications of chatbots can be found in numerous industries, including e-commerce, healthcare, and education. Yet,

the medical industry is one that has benefited from chatbot applications. Early on, a lot of academics concentrated on chatbots used in the healthcare industry to diagnose and treat patients remotely, but in the last four years, e-commerce chatbots and chatbots used in education have also gained popularity alongside medical chatbots. Moreover, chatbots are now a common way for banks, educational websites, and many other businesses to give services to their clients(Eltahir et al., 2023).

2.4 Frameworks of Chatbot Development

AI encompasses frameworks designed for constructing chatbots, providing greater flexibility and deeper insights. Some of the most renowned and advanced frameworks utilized in chatbot development include Facebook Bot Engine, Google's Dialogflow, Microsoft Bot Framework, IBM Watson, and Amazon Lex(Borah et al., 2019).

2.5 Approaches to Chatbot Development

Under this topic, we are going to discuss the approaches used in chatbot design and development from early stages to advanced approaches (ML based approaches).

2.5.1 Early Approaches

The approaches used in designing chatbots selected based on the type of chatbot to develop. The two common methods(Suta et al., 2020) of this approach are pattern-matching and rule-based methos. For instance, the first chatbot, ELIZA used pattern matching approach.

Pattern Matching (Rule based)

The pattern-matching approach involves the creation of predefined templates that serve as rules to select responses for input queries (Suta et al., 2020). This method operates under a set of established rules (Thorat & Jadhav, 2020). In such chatbot systems, the input is transformed into a "keystack" with the most relevant keywords at the top. The responses closely associated with the top-ranking keyword are identified, aiding the chatbot in selecting an appropriate response. This approach is suitable when the chatbot's responses are specific and not meant for broad questions. However, it proves ineffective when the input pattern doesn't match any predefined rules. Another challenge in building a rule-based chatbot is the time-consuming task of creating rules for various scenarios, making it challenging to cover every possible

situation. Commonly used languages for developing rule-based chatbots include Artificial Intelligence Mark-up Language (AIML), RiveScript, and ChatScript

During the period from 1995 to 2000, the development of the Artificial Intelligence Markup Language (AIML) took place, drawing inspiration from the concepts of pattern recognition and pattern matching (Adamopoulou & Moussiades, 2020). AIML is a tag-oriented, XML-based markup language that revolves around essential dialogue components referred to as "categories." These categories are formed through user input patterns (denoted by the <pattern> tag) and chatbot responses (indicated by the <template> tag). The following Table 2.1 shows an example of AIML code.

Table 2.1 Example of AIML Code

```
<category>
<pattern>User input</pattern>
<template>bot response</template>
</category>
```

2.5.2 Techniques in Advanced (Machine Learning) Approaches for Chatbot Development

The adoption of the self-learning (machine learning) approach became feasible as machine learning, deep learning algorithms, and Natural Language Processing (NLP) models advanced.

In contrast to rule-based chatbots, machine learning-based chatbots acquire knowledge from extensive datasets, requiring a sizable dataset to create a highly accurate chatbot model. Within this approach, two models are employed: retrieval-based and generative. Retrieval-based models select the most appropriate response from a set of predefined responses, while generative models generate responses using deep learning techniques (Huang, 2021). Deep learning, a subset of machine learning, involves constructing neural networks with multiple processing layers to enable computers to learn from data at various levels of complexity. This hierarchical learning, also known as hierarchical feature learning, replaces the need for higher-level dependencies on shallow networks in feature engineering. Instead, it utilizes successive

layers of representations to extract features from data, progressing from low-level to high-level features in a hierarchical manner (Krishna Vamsi et al., 2020).

Deep learning employs artificial neural networks to mimic the way in which the human brain perceives and assimilates information. This constitutes a subset of machine learning that operates at a high level of abstraction in order to manipulate data. Typically, deep learning methods employ ANNs to construct intelligent models and resolve intricate issues, utilizing the same information processing techniques as the human brain. A neural network, comprises three layers, each housing numerous neurons as shown in Figure 2.2. Neurons serve as the basic components of neural networks, and those within one layer are interconnected with neurons in neighboring layers to enable the flow of information between these layers (Sairamya et al., 2019).

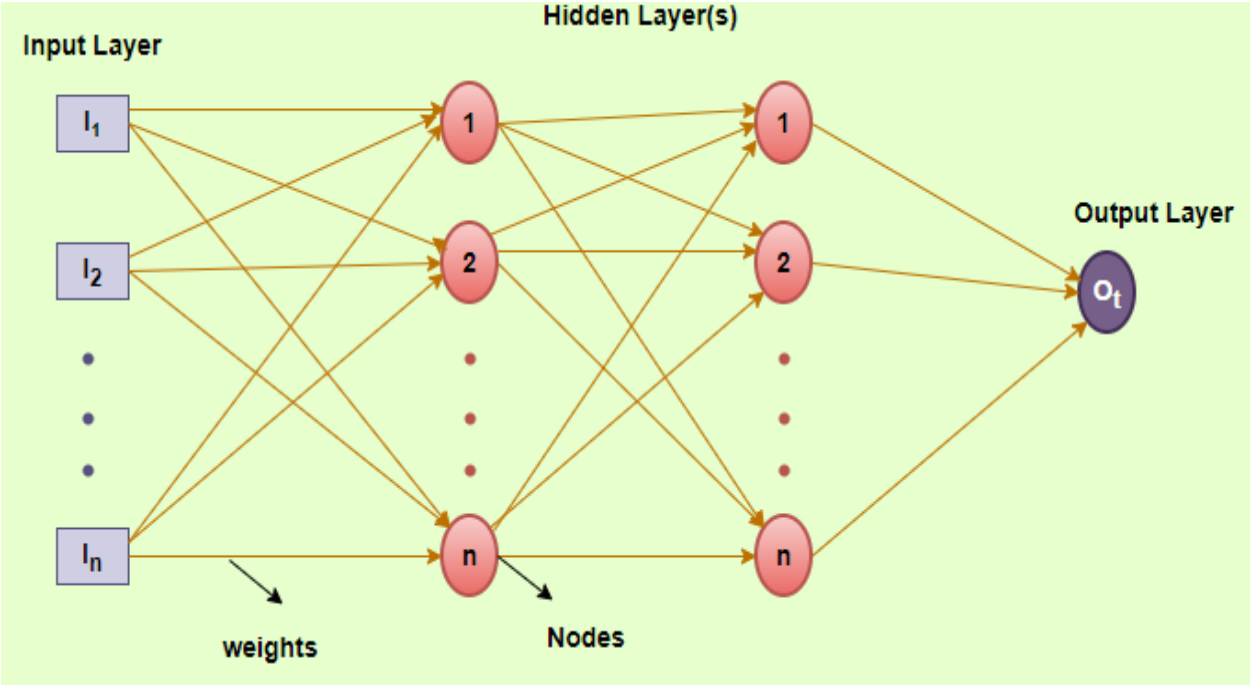

Figure 2.2 General Structure of ANN

Input Layer: Input nodes are responsible for gathering data from external sources. After receiving this information, the input nodes proceed to analyze, categorize, and process it before passing it on to the subsequent layer, known as the hidden layer.

Hidden Layer: Within artificial neural networks, hidden layers have the capacity to accept input from either the input layer or other hidden layers. These networks may incorporate

multiple hidden layers, with each layer examining the output of the preceding layer, conducting further processing, and subsequently transferring it to the subsequent layer in the sequence.

Output Layer: The output layer displays the final result of the data processing carried out by the artificial neural network and can consist of one or multiple nodes.

In ANN, neuros (nodes) receive input from another neuron and process it. And, every neuron has a propensity to fire, but only under specific circumstances. In neurons, activation functions are used to transform combined input.

$$A_f = \sum_{i=1}^{n} xiwi \qquad \textit{Equa. 2.1}$$

Where,

A_f is activation function used in the neurons to transform input

xi is particular input values to the neurons

wi is weights of particular input

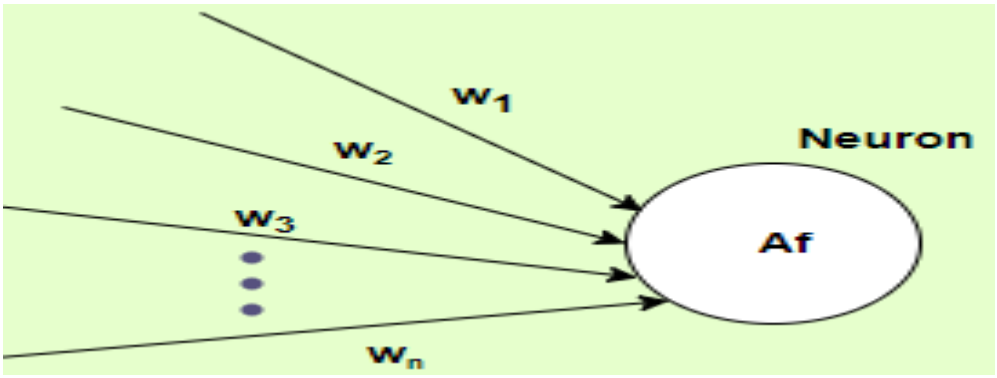

Figure 2.3 How Neurons Process Input

The ML and NLP research communities have been closely scrutinizing the suitability of DL techniques for NLP tasks, and these techniques have proven highly effective across a range of NLP tasks, spanning from character-level to sentence-level analysis(Alshahrani & Kapetanios, 2016). In recent years, there has been notable progress in the field of NLP, with its widespread application in various domains such as information retrieval, intelligent chatbot systems, and machine translation.

RNN and its variants like GRU, and LSTM(Shanghai hai yang da xue & Institute of Electrical and Electronics Engineers, 2020) have emerged as the most popular language modeling technology among all types of language models thanks to their potent capacity for processing sequence data and capacity for self-learning. Transformer is another deep learning that is developed to overcome the downsides of the RNN algorithm when working on sequential data (Vaswani et al., 2017). However, Transformers-based models are huge and complicated, requiring a lot of processing time and training data and, they are quite sensitive to the quantity and quality of the training data(Gillioz et al., 2020). We will discuss details of RNN variant algorithms in the next chapter.

2.6 Coffee Production in Ethiopia

Coffee is a significant crop that holds a prominent position among beverages. It is considered the second most traded commodity in the world after oil and is grown for its numerous health benefits, as well as its economic and socio-political advantages for various nations, such as Ethiopia. Coffee belongs to the Rubiaceae family, which encompasses a variety of species such as Coffea arabica, Coffea canephora, and Coffea liberica. Although both arabica and canephora coffee are cultivated for commercial purposes, the Coffee arabica variety accounts for 70% of the global coffee market due to its high-quality standards(Megersa, 2022). It was initially discovered in the highlands of southwestern Ethiopia and its species have been extensively distributed from their place of origin to many tropical and sub-tropical regions that are suitable for their growth. As a result, they can now be found in more than 100 countries worldwide(DaMatta, 2004).

Ethiopia has a long-standing tradition of cultivating, trading and enjoying coffee, and this tradition continues to be a vital component of both the local and national economies. Roughly 20% of the entire national population depends on coffee as a means of livelihood. About five million small-scale farmers employ a low input-output production system to contribute to 95% of the total coffee production(Adane & Bewket, 2021). Coffee constitutes a substantial portion of Ethiopia's exports, representing up to 50% of the country's exported goods and contributing to over a quarter of its foreign currency earnings.

Despite Ethiopia's leading position as Africa's primary coffee producer, it holds the second rank globally in terms of coffee exports. This can be attributed to several factors, including

inadequate supervision and care in both the pre- and post-harvest phases of coffee production, which are essential for meeting the stringent quality standards of the specialty coffee market. Additionally, shortcomings in marketing systems play a role in this scenario(Adane & Bewket, 2021).

Recently, there has been growing interest among researchers in the production of high-quality coffee, known for its exceptional quality and distinctiveness. Achieving this quality requires farmers to possess extensive knowledge in various aspects of coffee cultivation, including seedling preparation in nurseries, selecting suitable plantation sites, field preparation, transplanting seedlings, managing nutrients and irrigation, controlling weeds and diseases, and implementing shade regulations. These factors collectively contribute to the production of premium coffee beans(Adane & Bewket, 2021). Therefore, many of the methods used in the establishment and management of coffee fields have a beneficial impact on the competitiveness of the coffee production industry and generate foreign revenue for the country. However, the agricultural practices and management systems employed in Ethiopian coffee farming are largely traditional (between 90-95%) and require enhancement. This study aimed to develop an AI conversational model that supports farmers by providing the information they need in the production stages of coffee.

2.7 Afaan Oromo Language

Oromia is a Regional State within the Federal Government of Ethiopia, predominantly inhabited by the Oromo people who communicate in their native language, Afaan Oromo. Afaan Oromo belongs to the Afro-Asiatic language family and is a prominent member of the Cushitic language group. It holds the status of being one of the primary African languages, extensively utilized not only in Ethiopia but also in some neighboring nations, as highlighted by Tegegne in 2016. Apart from Ethiopia, Afan Oromo serves as a means of communication in Kenya, Somalia, Sudan, and Tanzania. This extensive usage positions Afan Oromo among the most widely spoken languages in Africa(Wayessa & Abas, 2020).

The official writing script for Afan Oromo is called Qubee (Latin based alphabet), which employs a total of 33 fundamental characters derived from the Latin alphabet. Among these characters, there are 5 vowels and 24 consonants, including 7 paired consonants that are combined together (such as "ny", “dh”). The Afan Oromo alphabet is similar to the English

alphabet in that it has capital and small letters. Similar to the English language, Afan Oromo has vowels that produce sounds on their own. Afan Oromo vowels are distinguished as having both short and long vowels.

Afan Oromo, like many other languages, utilizes blank spaces to indicate the end of a word. Additionally, word boundaries may also be denoted through the use of parenthesis, brackets, and quotation marks. Moreover, sentence boundaries in Afan Oromo are quite similar to those in English, as they are often marked by punctuation such as periods (.), question marks (?), and exclamation points (!)(Tesema & Tamirat, 2017).

2.7.1 Afaan Oromo Word Separation

In Afan Oromo, the smallest linguistic unit is the word, referred to as "jecha." Methods for isolating individual words vary among languages. In many Latin-based languages, including Afan Oromo, white space characters are commonly utilized to separate words when using the Latin script for written communication. For instance, the sentence "Tolaan Buna Dhaabe" demonstrates this practice, where the words "Tolaan," "Buna," and "Dhaabe" are delimited by white space characters.

2.7.2 Afaan Oromo Sentence Structure

Afan Oromo and English exhibit distinct sentence structures. Afan Oromo follows a subject-object-verb (SOV) pattern, wherein the subject, object, and verb appear in that order. In contrast, English employs a subject-verb-object (SVO) structure, with the subject preceding the verb and object. For instance, the Afan Oromo sentence "Caaltuun gabaa deemte" adheres to the SOV pattern, with "Caaltuu" as the subject, "gabaa" as the object, and "deemte" as the verb. However, when translated to English as "Chaltu went to the market," the sentence takes on an SVO structure.

Furthermore, differences exist in the placement of adjectives between Afan Oromo and English. In Afan Oromo, adjectives typically come after the noun or pronoun they describe, while in English, adjectives typically precede the noun. For example, "Muka dheeraa" (a tall tree) in Afan Oromo has the adjective "dheeraa" following the noun "Muka." In English, it would be expressed as "a tall tree," with the adjective preceding the noun.

Both Afan Oromo and English use punctuation marks for similar purposes, except for the apostrophe. In English, the apostrophe (') indicates possession, whereas in Afan Oromo, it represents the “hudhaa” sound, an essential element in the language's reading and writing systems. This sound is particularly important when writing words containing two consecutive vowels, like "ga’ee" (role). In some instances, the apostrophe in Afan Oromo can be replaced with the letter "h." For instance, "ga’ee" and "danda’a" can also be written as "gahee" and "dandaha," respectively, without altering their meanings.

2.8 Related Work

Under this topic, we are going to discuss related works done on chatbots for various agriculture sectors. Many researchers are focusing on deep learning techniques for chatbot model development.

(Jain et al., 2019) developed “AgriBot (Agriculture-Specific Question Answer System” with the objective of helping Indian farmers access timely farming-related information to increase agricultural output. They used a sentence-to-vector (Sen2Vec) model and divided their dataset into two pars training (80%) and test (20%) data. First, they trained the word2vec model, and then the output of this model (word embedding) was converted to sentence embedding. Cosine similarity was employed to compare embedding vectors of queries with those of training data, resulting in a list of answers. Subsequently, an answer ranking technique was applied to determine the best answer for output. The model gave them an accuracy of 56% without synonym elimination and entity extraction and 86% after using entity extraction. This study makes a significant contribution to enhancing agricultural yields by providing farmers with convenient access to agricultural information. Nonetheless, a drawback of this system is its limitation to addressing only pre-existing questions stored in the database, rendering it incapable of responding to new, unlisted questions.

In the work titled "Text-Based Smart Answering System in Agriculture," conducted by (Mary A & Sukumar, 2021), a chatbot with the objective of providing farmers with answers to fundamental queries and offering agricultural knowledge and solutions was developed. To train the system, they curated the dataset by constructing a Bag of Words (BOW), which is a model for representing unstructured text in a vector space. After all, they gained an accuracy score of 97.83 % for RNN and 96.97 % for MLP. They trained the model, using two

techniques and compared the accuracies of both, and selected the one with a high accuracy value (RNN). Although, there are variants of RNN algorithms that are found to overcome the problem of vanishing gradient with the traditional RNN algorithm like LSTM and GRU they did not train the model with these models and compare them with RNN. Additionally, they have not used any overfitting handling techniques and evaluation metrics to evaluate the model. This model has undergone training on the Indian language dataset, which is morphologically different from Afaan Oromo language. The other limitation of this study is that they used BOW for feature extraction. However, it cannot capture the context of the word in the text, which is important in chatbot models.

(Suebsombut et al., 2022) proposed the study titled "Chatbot Application to Support Smart Agriculture". In this study, they built the chatbot application named LINE that acts as a knowledge and information representation that offers farmers recommendations on crop cultivation. This study aims to respond to crop-related questions from farmers via chat box. The proposed chatbot application is rule-based, requiring farmers to input specific keywords for responses. This constraint could impede the chatbot's capacity to comprehend natural language and potentially obstruct the dissemination of information. The authors acknowledge this limitation and plan to enhance the chatbot's asking function to be more intelligent in the future. Therefore, the gap in the study is deficient in the form of a requirement for an intelligent chatbot capable of comprehending and replying in natural language to improve the user experience and provide more accurate information to farmers.

(Imalka et al., 2022) presented real-time question-and-answer platforms for the agriculture sector, incorporating artificial intelligence technology platforms that enable farmers to receive prompt and accurate information from agrarian officers in response to their queries. In the study, they used the RNN model to train the model on the dataset. The dataset for this study is prepared presuming some questions asked by farmers, and it is in XML format. The study mentions a high training accuracy rate of 95.24%, but the dataset used was assumed, not directly from farmers. This can affect how the system performs in real-life scenarios with a larger and more diverse set of questions from farmers. The research employed RNNs, which are susceptible to vanishing gradient issues. However, alternative deep-learning language models like LSTM and GRU are capable of mitigating this problem.

(Segni, 2021) introduced the "Afaan Oromo Text-based Chatbot for Enhancing Maize Productivity," marking the first instance of an Afaan Oromo text-based chatbot. The primary aim of this study was to improve maize productivity by delivering maize production-related information. They used the Recurrent neural network technique of deep learning with seq2seq. in this study, they achieved an accuracy of 84.4% using DNN and created a chatbot that gives farmers crucial information on how to increase the productivity of their maize crop. This study has made a significant contribution to the field of developing Afaan Oromo text-based chatbots because it was the first Afaan Oromo chatbot in the domain of agriculture. Despite achieving a high level of accuracy, this study may be limited in terms of the number of target classes or the size of the request intents, as only 25 target classes were used. But, while the study shows promising results in terms of accuracy it will be valuable to handle a wider range of agricultural problems and train the model on other agricultural product datasets like the coffee production dataset to add solution ranges to agriculture-related problems.

2.9 Summary of Related Works

Several researchers have explored the development of chatbots and question-answering systems to support the agriculture sector. Jain et al. developed AgriBot, which is a question-answering system that can provide access to farming-related information to Indian farmers. However, it has the capability to respond solely to questions that already exist in its database and lacks the ability to address novel questions. Mary A & Sukumar introduced a Text-Based Smart Answering System in Agriculture that employed RNN and achieved an accuracy rate of 97.83%. Nonetheless, RNNs face difficulties due to the vanishing gradient problem, which hinders the network's ability to grasp long-term patterns in the data. To overcome the vanishing problem of the RNN model, there are RNN variant algorithms such as LSTM and GRU but, they did not train these models. Not only this, they used BOW for feature extraction in the study. The BOW approach can merely indicate the presence of words within the text but falls short in capturing the context of those words, a factor that holds significant importance in determining the chatbot model's performance. The other author, Suebsombut et al. proposed the Chatbot model to Support Smart Agriculture, which aimed to provide responses to farmers' crop environment questions through a chatbot application named LINE. However, the proposed chatbot application is rule-based. The rule-based chatbot can only answer the query that exactly matches the pre-existing question. Imalka et al. proposed real-time question-and-

answer platforms for the agriculture sector, incorporating artificial intelligence technology platforms that enable farmers to receive prompt and accurate information from agrarian officers in response to their queries. The dataset for the models was created assuming it was asked by the farmer, which can affect the system's performance in real-life scenarios with a larger and more diverse set of questions from farmers. The other study named 'Afaan Oromo Text-based Chatbot for Enhancing Maize Productivity' proposed by Segni achieved an accuracy of 84.4% with the DNN model using 25 target classes (tags). This study used Seq2Seq and the DNN model. However, there are widely used language models like LSTM and GRU that have been scoring promising results in NLP tasks. However, they did not train these models on their dataset and check their performance. In this study, we proposed a chatbot that assists farmers in the production processes of coffee crops. The model trained on the conversational dataset on the production processes like how to take care of the coffee plant, how to prepare quality seeds, type of soil for coffee production, watering of the coffee plant, how to control pests and weeds, how to prepare station and fields to plant coffee. This chatbot assists farmers in these coffee production processes to help them increase their coffee crop yield both in quality and as a result it contributes to increasing the country's income from the production of coffee crops.

Table 2.1 Related Work Summary

#	Paper	Model used	Gaps
1	Agriculture-Specific Question Answer System (Jain et al., 2019)	Sen2Vec Model with 86% accuracy	- Only answer pre-existing query and route the new question to human employee - Did not train state of the art language models on their dataset to check the performance of the models
2	Text Based Smart Answering System in Agriculture (Mary A & Sukumar, 2021)	RNN with 96.97% accuracy, MLP with	- RNN model will be subjected to vanishing gradient problem and there are models came to overcome this problem

		97.83% accuracy	- They used BOW for feature extraction and BOW can only represent the existence of the words in the sentence. But there are feature extraction techniques that can capture the context of the word in the sentence.
3	Chatbot Application to Support Smart Agriculture (Suebsombut et al., 2022)	Rule-based	- To receive a response from the chatbot, users must input the specified keywords accurately; otherwise, they will not receive any response.
4	Farming Through Technology Driven Solution for Agriculture Industry(Imalka et al., 2022)	RNN with 95.24% accuracy	- RNN model will be subjected to vanishing gradient problem and there are models came to overcome this problem - The question was created assuming it was asked by farmers which can affect how the system performs in real-life scenarios with a more diverse set of questions from farmers
5	Afaan Oromo Text-based Chatbot for Enhancing Maize Productivity (Segni, 2021)	DNN with 84.4% accuracy, Seq2Seq with 80% accuracy	- Language models like LSTM and GRU have the ability to capture long-term dependencies of data in the sequences of text.

CHAPTER THREE

3. Methodology

This chapter offers an elucidation of the methods, procedures, tools, and steps utilized to accomplish the goals of this study. encompasses various methods, such as collecting and preparing the dataset, preprocessing the data, selecting feature extraction methods, choosing models, employing evaluation techniques to gauge the chatbot model's performance, and utilizing tools to construct both the suggested model and the graphical user interface.

3.1 Research Design procedures

Research design pertains to the comprehensive framework of methodologies and strategies chosen by a researcher to conduct their study. It outlines the plan for gathering and analyzing data, and provides a framework for interpreting the results.

The following procedures were employed to accomplish the study's goal:

- ✓ **Research area identification**: We are motivated to conduct this study to solve the problem of delivering coffee production-related information to the farmers. Then, we reviewed various agricultural research papers that helped us to know how to conduct this study on one area of agricultural products.
- ✓ **Literature review**: After we identified the area, we reviewed different papers done on the chatbot in agriculture and other domains. Then, we identified a gap and conducted the research focused on developing a conversational system to assist farmers in increasing their coffee crop yields.
- ✓ **Formulating research questions**: During this phase, we developed specific objectives and research inquiries that the study's results must address to resolve the identified issue.
- ✓ **Dataset collection and preparations**: The data was gathered, organized, and prepared in a manner that is appropriate for the model to experimentally address the research questions.
- ✓ **Model Evaluation**: The suggested chatbot model's performance is assessed through different model evaluation methods to determine the model that performs the best.

3.2 Procedures of the Model Building

The structure includes several stages such as data preprocessing, feature extraction, training and testing, comparison of patterns derived from the Intent's data, and retrieval of relevant information that matches. The preprocessing phase encompasses the extraction and preparation of data intended for use as input in constructing the model. We selected deep learning models and trained on our dataset. The models selected for this study are BiGRU, BiLSTM, CNN-BiGRU, and CNN-BiGRU-AM which have demonstrated excellent performance in Natural Language Processing tasks involving sequential data.

3.2.1 Datasets

The initial stage in any Machine Learning task involves data preparation, which is crucial to ensure that the machine can accurately interpret the data. The data utilized for this research was gathered from the Oromia agriculture bureau, the book titled “Teekninoloojii Oomisha, Qulqullinaa fi Walitti Hidhamiinsa Gabaa Bunaa” written by Hayile Fufa who is a senior coffee specialist and from individual farmers. The collected dataset from the Oromia Agriculture Bureau and from the book is the science (guideline) that farmers should follow to increase their yields of coffee production in both quality and quantity. Oromia Agriculture Bureau’s data is prepared by the Jimma Research Institute based on farmers' requests for coffee production instructions. After collecting the dataset, we saved the gathered data in JSON (Javascript object notation) format to make it suitable for our study, which comprises sets of question-answer pairs. JSON is a commonly employed format for sharing files and data, and it finds application in diverse domains like web APIs, scientific computing, data management, and configuration management. It is a common tool for transferring information between different systems and applications(Harrand et al., 2021). To create a Coffee production assistant chatbot, the data is organized into categories such as tags, patterns, responses, and context within an intent file.

- ✓ **Tags**: Potential categories representing a user's intent when posing a question.
- ✓ **Patterns**: Common methods that users typically employ when posing questions associated with a specific tag.
- ✓ **Responses**: Predetermined replies for each tag within the dataset that the model can select to answer a specific question.

- ✓ **Context**: Words within the context related to a tag, facilitate more effective and precise classification of the user's request.

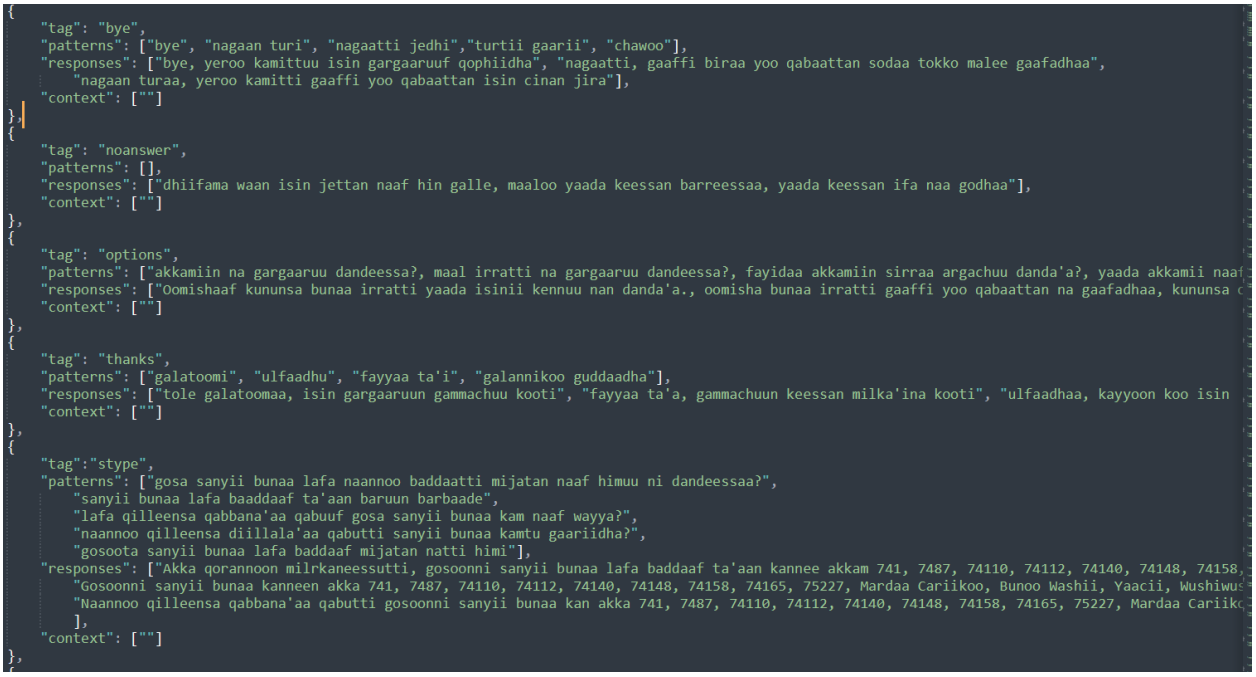

Figure 3.2 Sample of Data Preparation

From the above sample data ‘tag’ are the classes which the user’s intent is categorized under, the ‘patterns’ contain a question that the user ask under a specific tag, and ‘responses’ are the possible answers users may get for their query. This data is prepared in the above format aiming that, the model predicts the class (tag) the user’s question is classified and then chooses the answer randomly from responses. This structured data is used to train the chatbot model to appropriately respond to users’ (farmers’) inquiries and requests regarding the production of Coffee. Table 3.1 shows the number of each dataset structure prepared for the proposed study.

Table 3.1 Dataset Detail

Structure of the dataset	Numbers
Tags	123
Patterns	2,367
Responses	290

3.2.2 Data Preprocessing

Data gathered from the real world cannot be used directly by the model. It contained noise and a disorderly form, which are inappropriate for the model. Therefore, the raw data should be processed in a format that is suitable for the machine to understand. Data preprocessing involves converting unstructured and unusable raw data into a structured and usable format to make it suitable for analysis or further processing. This transformation is necessary to ensure that the data can be effectively utilized according to the specific requirements. The preprocessing task includes tokenization, stop word removal, and normalization. In this study, we process the data so that the machine can understand and manipulate it.

Tokenization: Tokenization is the fundamental procedure of dividing a given sentence into individual tokens, serving as the initial step that prepares the data for subsequent processing. This process entails splitting a sentence or text into distinct words or tokens, which are segregated by spaces, punctuation marks, or specified separators. Tokenization techniques assist in extracting more essential features from the given dataset. Punctuation marks such as periods, commas, and semicolons are used to mark the boundaries between words.

Normalization: Texts written by different users may contain the same words written in different ways, which can create confusion for machines in understanding their meanings. Text normalization is a technique that resolves this issue by converting similar words with different spellings or formats into a single standardized word. Normalization is crucial for languages that utilize Latin characters. The normalization procedure involves converting the character case in the text, which can be in upper case, lower case, or mixed case, to a consistent case across the text. Lowercase conversion is typically preferred as users usually input text in lowercase, rather than dealing with capitalization. Furthermore, a regular expression is employed to standardize words that convey comparable meanings but are expressed in varying formats. For example, in the Afaan Oromo language, certain words such as haga/hanga, erga/ega, baayee/baayy'ee, osoo/otoo, mini/miti, have the same meaning and have been combined into a single normalized word.

Stop word removal: Not all words present in the documents are equally significant in creating a chatbot model. Certain words are insignificant and do not assist in categorizing the users' inquiries into the correct category. These words ought to be eliminated from the dataset to

reduce the model's training duration. In this study, stop words in the Afaan Oromo language have been compiled based on prior studies and customized specifically for this work. These words are the most commonly used terms in any language, such as pronouns, prepositions, conjunctions, articles, and particles. Examples of such commonly occurring stop words in Afaan Oromo include "fi/and", "kana/this", "Akka/like", "kanaaf/so", "siif/for", "irra/on", "sun/that", "isa/he", "kee/yours", and others.

3.3 Feature Extraction Technique

Text feature extraction is a crucial stage in data mining and information retrieval. Its purpose is to identify and quantify significant words or phrases within a given text, transforming it into structured data that computers can interpret and analyze. This process involves reducing the text's complexity and establishing a mathematical model that represents its underlying patterns and relationships(Sichuan Institute of Electronics & Institute of Electrical and Electronics Engineers, 2018). Different methods are utilized in various research to extract meaningful features from textual data, such as BOW, TF-IDF, and word embeddings (word2vec, Glove, and fasttext). These techniques are commonly used for encoding purposes.

The BOW approach only takes into account the existence of the words in the text and whether they appear in the document, without considering the arrangement or order in which they appear(Tishk International University et al., 2018). The TF-IDF method addresses the constraint of the BOW approach by taking into account not only the occurrence of a word but also its frequency within a particular document (term frequency) and the frequency of the word across all documents in the corpus (inverse document frequency). However, like BOW it does not consider the context of the word in the text(Ahuja et al., 2019). Word embeddings are indeed powerful techniques that address some of the limitations of methods such as TF-IDF and BOW. They capture the semantic meaning of words and enable the representation of phrases, sentences, or documents in a continuous vector space.

3.3.1 Word2vec

Word2Vec generates numerical vectors for words, which act as distributed and meaningful representations of various features associated with the words. These features include the contextual relationships that the words share within our language(Di Gennaro et al., 2021).

There are two architectures of the word2vec model: the Continuous Bag of Words (CBOW) & the Skip-Gram model.

CBOW: It aims to predict the present word by considering the context that surrounds it

Skip-Gram: In contrast to the CBOW, it is designed to forecast the words around it, using the current word as input.

Word2Vec constructs its word vectors based on the words it has been trained with. Consequently, any word absent in the training data will be categorized as an out-of-vocabulary (OOV) word. This means that for the words not present in the training dataset word2vec cannot provide meaningful representations. FastText feature extraction technique addresses the limitation of Word2Vec by assigning a vector to even those words not present in the vocabulary. This is achieved by considering the sub-word components of the word(Suhaili et al., 2022).

3.3.2 FastText

FastText serves as an extended version of the word2vec approach for word embeddings. In the FastText model, words are represented as character n-grams rather than directly learning word vectors(Athiwaratkun et al., 2018). It is an enhancement built upon the Skip-Gram model, a word's sub-structure to enhance the vector representations derived from Word2Vec's Skip-Gram method(Suhaili et al., 2022).

FastText utilizes sub-word information instead of directly learning words, whereas the word2vec model constructs word vectors directly. In this study, we employed both FastText and word2vec to compare and assess the performance of both techniques on our dataset.

3.4 Model Selection

Models to train machines depend on the characteristics of the model to be created. Currently, deep learning models are dominating areas of NLP tasks(Sri Shakthi Institute of Engineering and Technology et al., 2020). The choice of deep learning models relies on the characteristics of the problem at hand and the manner in which it needs to be addressed. The proposed coffee production assistant chatbot supports a multi-class classification challenge. The dataset is structured as a JSON file, involving multiple tags. Each tag comprises numerous patterns that represent user inputs, and the goal is to assign the corresponding class to each pattern.

Previously done papers indicated that different chatbots having multi-class classification challenges have been developed using various deep learning techniques such as RNN(Dhyani & Kumar, 2019), LSTM(Chou & Hsueh, 2019), GRU and their model variants hybrid with CNN. LSTM and GRU have the capability to capture long relationships within input sequences, but they lack the capacity to selectively emphasize crucial elements of the input at any given moment. To enhance the performance of these models, integrating an attention mechanism can be beneficial(Zhou et al., 2023).

Since RNN variant models are providing the best performance in sequential and NLP tasks, we decided to train the proposed coffee production assistant chatbot using deep learning models like BiLSTM, BiGRU, as well as CNN-BiLSTM and CNN-BiGRU-AM. It is obvious that it is difficult to say one technique is better than the other without experimenting with them, hence, we selected the outperformed technique from the above deep learning techniques by comparing them based on the result they generated using evaluation metrics.

3.4.1 Recurrent Neural Network (RNN)

RNN is an artificial neural network explicitly developed for handling sequential data or information that is organized in a time series(Xiao & Zhou, 2020). It has gained immense popularity as a language modeling technology due to its impressive capacity for self-learning and analysis of sequential data(Borah et al., 2019). In general, an RNN language model comprises three primary layers, which include an input layer, a hidden layer, and an output layer, as illustrated in Figure 3.3.

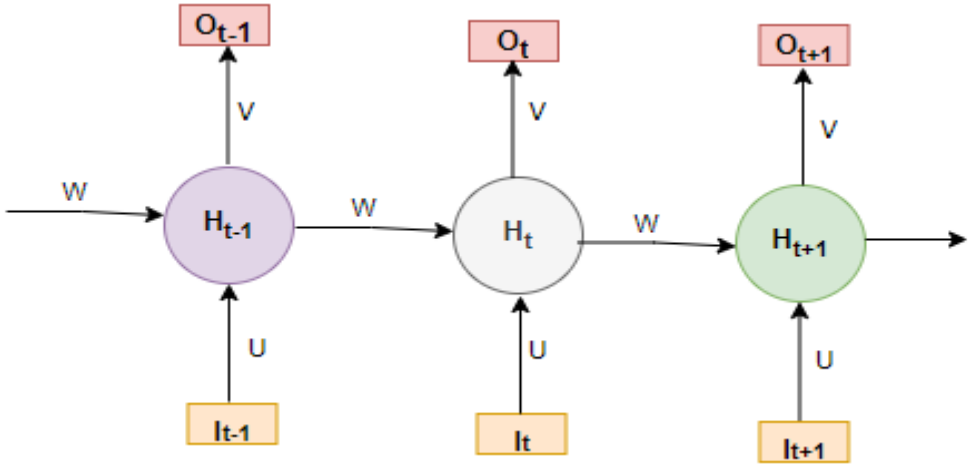

Figure 3.3 Recurrent Neural Network Structure

To calculate hidden and output layers it uses equation 3.1

$$H_t = \delta(U * I_t + W * H_{t-1} + bH) \qquad \textit{Equa. 3.1}$$

$$O_t = softmax(V * Ht + bO)$$

Let us break down the equations and explain the variables:

H_t: The hidden state at a given time step, denoted as t.

I_t: input at time step t.

U: The weight matrix linking the input It with the hidden state Ht.

W: weight matrix connecting the previous hidden state H_{t-1} to the current hidden state H_t.

bh: bias term for the hidden state.

V: The weight matrix that connects the hidden state Ht to the output Ot.

bO: bias term for the output.

The calculation of the hidden state Ht involves performing a dot product ($*$) between the input U and it, the dot product between the weight matrix W and the preceding hidden state H$t-1$, and subsequently adding the bias term bH. The outcome undergoes a delta function δ (typically the sigmoid or hyperbolic tangent function) to introduce non-linearity and capture the hidden layer's output. Meanwhile, the output Ot is determined by conducting a dot product ($*$) involving the weight matrix V and the hidden state Ht, along with the inclusion of the bias term bO. The outcome then undergoes a softmax function (softmax) to transform it into a probability distribution encompassing different classes or categories.

RNN's ability to analyze sequential data makes it a superior algorithm for recognizing relationships within sequences of words. However, the issue of vanishing and exploding gradients encountered in RNNs led to the development of a novel neural network architecture known as LSTM. LSTMs have emerged as a highly effective design and are gaining widespread popularity for various tasks that involve the processing of sequential data(Alom et al., 2019).

3.4.2 Long Short-Term Memory (LSTM)

LSTM represents a category of deep learning models proficient in handling sequential data by facilitating the retention of information over time. They belong to the RNN family but stand

out for their capability to address the vanishing gradient issue commonly encountered by conventional RNNs. LSTM functions in a manner akin to that of an RNN cell, featuring an architectural arrangement comprising three segments known as gates: the input gate, forget gate, and output gate. These gates regulate the flow of data into and out of the memory cell or LSTM cell, as depicted in Figure 3.4.

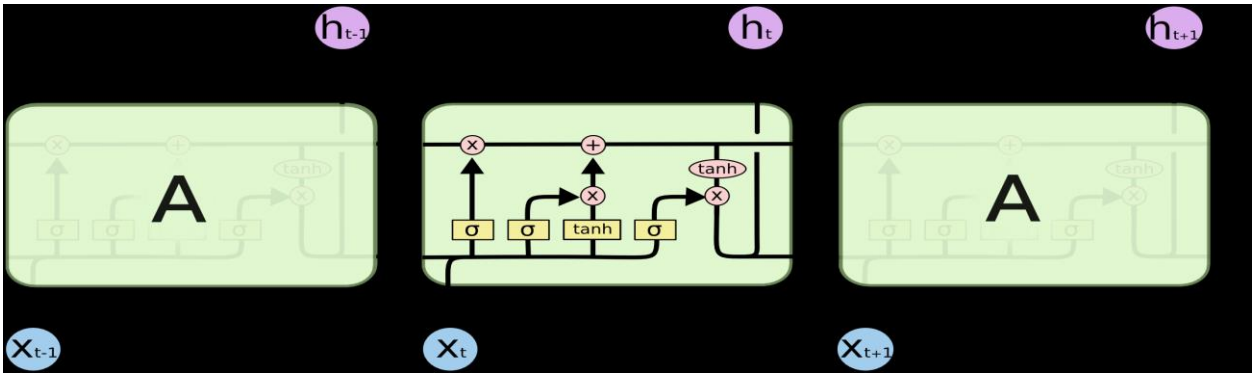

Figure 3.4 Architecture of LSTM(Yu et al., 2019)

Let us discuss the figure:

Forget gate (f_t): decide whether to keep or discard the information from the previous time step.

$$f_t = \delta(U_f * X_t + H_{t-1} * W_f) \qquad \textit{Equa. 3.2}$$

In this context:

Xt represents the input at the current timestamp.

Uf signifies the weight associated with the input.

Ht-1 denotes the hidden state from the previous timestamp.

Wf represents the weight matrix related to the hidden state

Input gate (I_t): assess the significance of fresh information brought in by the input.

$$I_t = \delta(U_i * X_t + H_{t-1} * W_i) \qquad \textit{Equa. 3.3}$$

In this context:

Xt stands for the input at the current timestamp t.

Ui refers to the input weight matrix.

Ht-1 represents the hidden state from the previous timestamp.

Wi denotes the weight matrix for the input relative to the hidden state

The combination of a hidden state from the previous timestamp t-1 and the input x at the current timestamp t is responsible for determining the fresh information (Nt) that should be conveyed to the cell state.

$$N_t = tanh(X_t * U_c + H_{t-1} * W_c) \qquad \textit{Equa. 3.4}$$

Due to the presence of the tanh function, the new information will fall within the range of -1 to 1. The operation involves either incorporating the information into the cell state at the current timestamp when Nt is positive or deducting it from the cell state when the value is negative. Following this, the cell state undergoes an update by adding the new information, and this update process adheres to the provided equation.

$$C_t = f_t * C_{t-1} + It * N_t \qquad \textit{Equa. 3.5}$$

In this instance C_{t-1} in this case denotes the cell state at the present timestamp.

Output gate (O_t):

$$O_t = \delta(U_o * X_t + H_{t-1} * W_o) \qquad \textit{Equa. 3.6}$$

The value from the output gate used to calculate the current state (H_t)

$H_t = O_t * tanh(C_t)$ and the output= softmax(H_t)

LSTM model is one variant of RNN that solves long sequences problems of RNN. According to previous studies done on NLP tasks using LSTM(Santos et al., 2020) model it performed very well for languages models. However, LSTM, lack the capability to utilize information from future data points for learning sequence patterns. Consequently, it may miss out on valuable supplementary information while processing sequential data. BiLSTM improve the performance of LSTM by introducing bidirectional processing, allowing the network to incorporate information from both preceding and succeeding data points when analyzing

sequences. This advancement overcomes the constraint of conventional LSTMs that solely process data in a single direction.

3.4.3 Bidirectional Long Short-Term Memory (BiLSTM)

BiLSTM is an expanded form of LSTM where two LSTM units are utilized for processing the input data(Handhayani et al., 2022). This approach combines the features of LSTMs with bidirectional processing, enabling the network to consider both future and past contexts when examining sequential data. In the LSTM model, data is processed sequentially in a single direction, either from the past to the present or vice versa, leading to limitations in handling sequences. The following Figure 3.5 shows the architecture of BiLSTM.

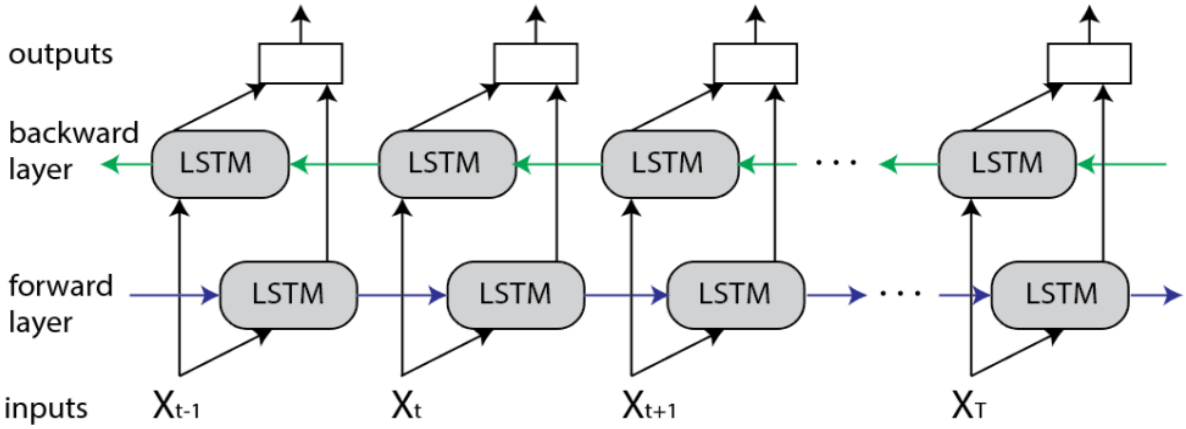

Figure 3.5 Architecture of BiLSTM Model

3.4.4 Bidirectional Gated Recurrent Unit (BiGRU)

GRU and LSTM are comparable however, GRU is a lighter version of RNNs when it is compared to LSTM in terms of topology and computation complexity(Alom et al., 2019). It combines the forget and input gates into a unified update gate and integrates the cell state and hidden state while introducing several other adjustments. GRU needs a smaller number of network parameters, which makes it faster. However, if there is enough data and computational power, LSTM delivers surpasses performance(Yang et al., 2020a). GRU's input and output structures are similar to those of a standard RNN, while its internal structure behaves similarly to an LSTM(Yang et al., 2020b). According to (Nosouhian et al., 2021; Bhuvaneswari et al., 2019) on the small dataset, the GRU model performs better than the LSTM model in a setting with equal parameters, including loss function, number of epochs, optimizer, and batch size. Like RNN and LSTM GRU captures information only in one

direction. BiGRU has the ability to process bidirectionally to enhance the model's understanding of the input sequence, which can lead to more accurate and contextually rich representations. The following Figure 3.6 shows the overview of BiGRU structure.

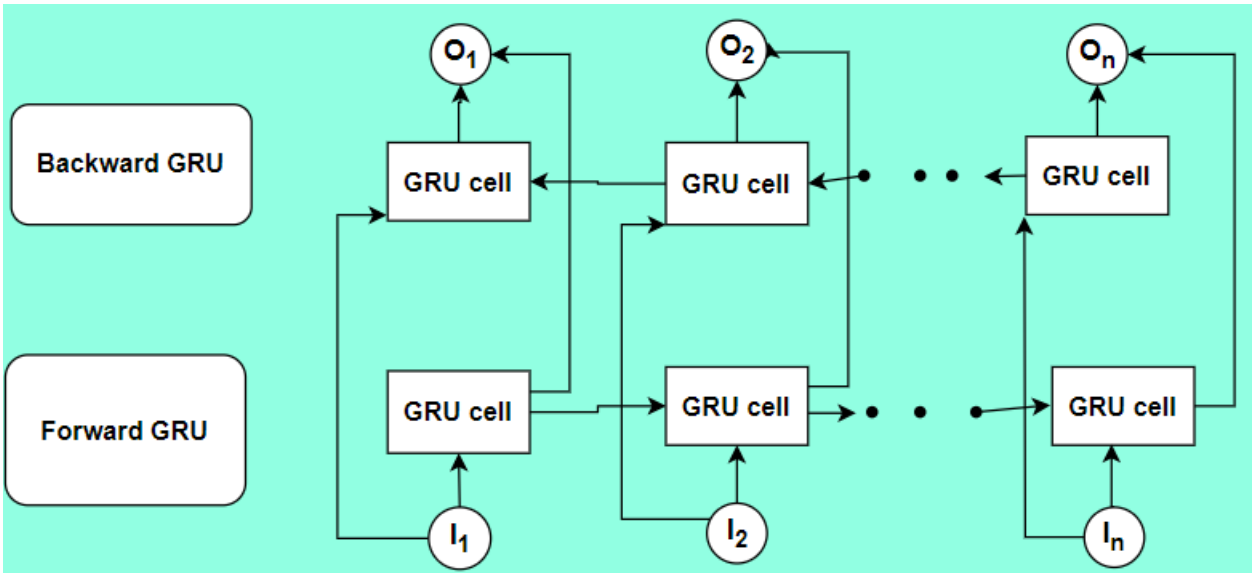

Figure 3.6 Structure of BiGRU

Were I_1, I_2, I_n represents inputs and O_1, O_2, O_n represents outputs

3.4.5 CNN-BiLSTM Model

Convolutional Neural Networks (CNNs), which were first created for image processing tasks, have had great success in computer vision. This neural network functions in a feed-forward fashion and includes a convolutional structure as part of its operation. In the field of NLP, CNNs are frequently employed to detect and understand nearby patterns and features within sequences of words(Yue & Li, 2020). RNN and its various variants have consistently achieved state-of-the-art results across a wide range of NLP tasks(Xiao & Zhou, 2020). The CNN model can process a text using a one-dimensional convolutional layer and word embedding. Although the CNN model is superior at extracting local characteristics from word-embedded data, it is ineffective for learning sequence dependencies(Tam et al., 2021). Since the BiLSTM model can learn sequence relationships, it is conceivable to employ a hybrid technique of CNN and BiLSTM to overcome these limitations of CNN. In NLP, the CNN-BiLSTM architecture combines the advantages of CNNs for local feature detection and BiLSTMs for collecting context and relationships. This enables the model to determine the organization of

the text with good accuracy. The following Figure 3.7 shows the architecture of the CNN-BiLSTM model.

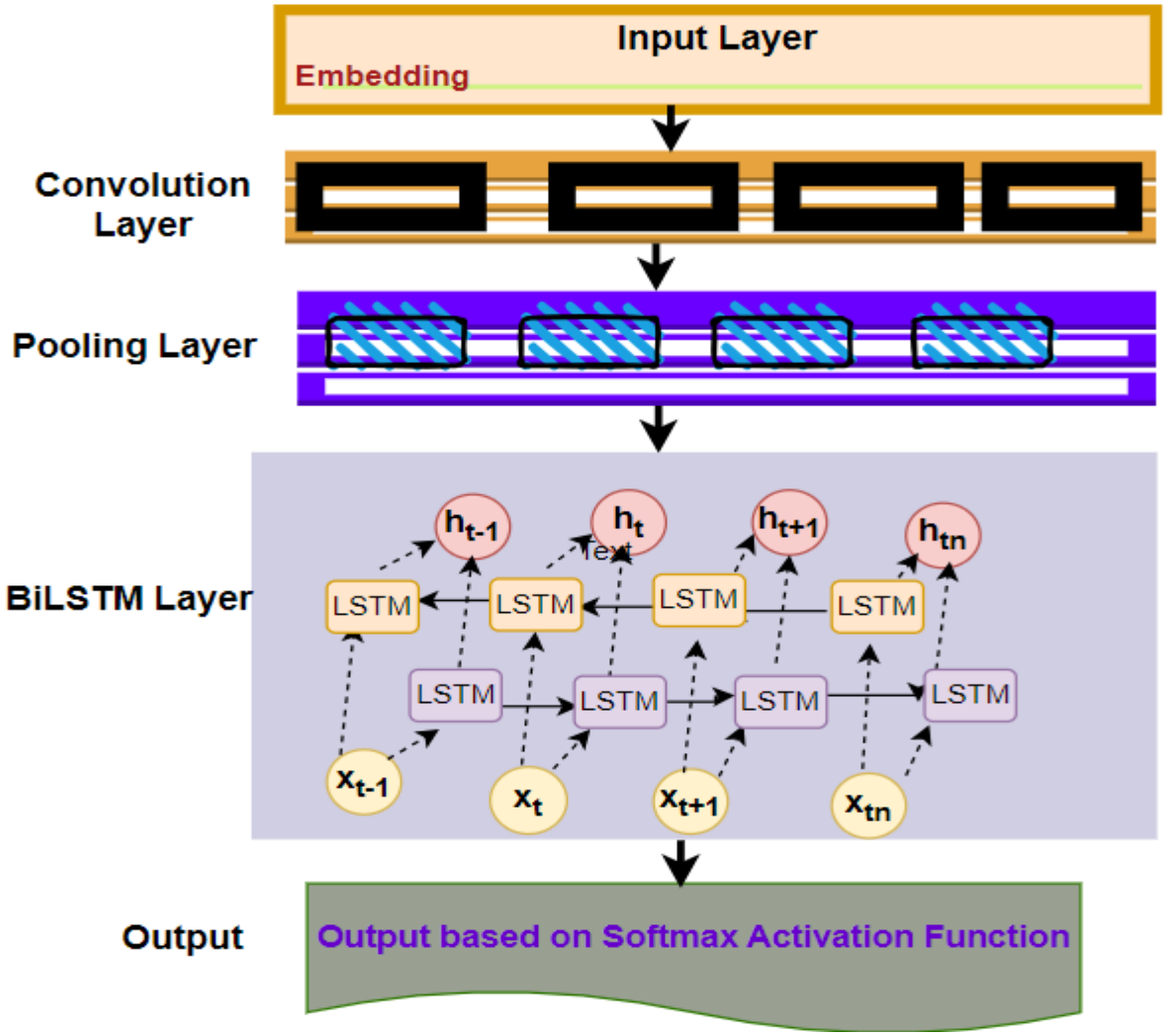

Figure 3.7 Architecture of CNN-BiLSTM Model

3.4.6 CNN-BiGRU-AM Model

The CNN-BiGRU-AM architecture successfully captures patterns and contextual information in sequential data by combining the strengths of convolutional layers, bidirectional recurrent layers, and attention mechanisms(Zhou et al., 2023). This architecture is flexible and can be used for a variety of NLP tasks where comprehending the connections and patterns within sequences is essential for making accurate predictions or decisions. The concept of attention mechanisms (AM) is based on how humans visually focus on things. To improve performance when processing long sequences, models use AM to concentrate on important parts of the

input while disregarding unimportant details (Sun et al., 2022). It allocates varying weights to distinct segments of the input sequence during prediction. This enables the model to concentrate on the most pertinent portions of the input sequence while producing an output. Figure 3.8 shows the structure of the attention mechanism. In this study, we have added a self-attention mechanism to the proposed model. The self-attention mechanism serves as a vital building block in many advanced deep learning models, with a particular emphasis on applications within the field of NLP.

Given a sequence of input vectors X=[x1, x2,…,xn], where each xi represents an input vector of dimension d, the self-attention mechanism computes weighted combinations of these input vectors to produce an output sequence of vectors Y=[y1,y2,…,yn](Shen et al., 2018). It has three main components:

- ✓ **Query (Q):** To derive query vectors, a linear transformation is performed on each input vector x_i using a trained weight matrix W^Q, resulting in a query vector q_i:

$$q_i=W^Q x_i \qquad \textit{Equa. 3.7}$$

- ✓ **Key (K):** Likewise, a linear transformation is applied to each input vector x_i using a trained weight matrix W^K, resulting in a key vector k_i:

$$k_i=W^K x_i \qquad \textit{Equa. 3.8}$$

- ✓ **Value (V):** For the computation of values, a linear transformation is applied to each input vector x_i using a trained weight matrix W^V, resulting in a value vector v_i:

$$v_i=W^V x_i \qquad \textit{Equa. 3.9}$$

The usual way to calculate the similarity between a query vector q_i and a key vector k_j involves taking the dot product of these vectors. In some cases, this dot product is adjusted by dividing it by a factor $\sqrt{d}k$, where d_k denoted as the dimensionality of the key vectors:

$$\text{eij} = \frac{\text{qi. ki}}{\sqrt{d}k} \qquad \textit{Equa. 3.10}$$

In this context, the symbol e_{ij} represents the score that indicates how similar the i^{th} query is to the j^{th} key.

The similarity scores e_{ij} are subsequently subjected to a softmax function to calculate the attention weights a_{ij}. These attention weights indicate the degree to which each key contributes to generating the output for a specific query:

$$\text{aij} = \frac{\exp(eij)}{\sum_{j\prime=1}^{n} \exp(ei\,j')} \qquad \textit{Equa. 3.11}$$

The resulting vector y_i for a given query q_i is determined by taking a weighted sum of the value vectors V_j, where the weights are determined by the attention weights a_{ij}:

$$\text{yi} = \sum_{j=1}^{n} aijVj \qquad \textit{Equa. 3.12}$$

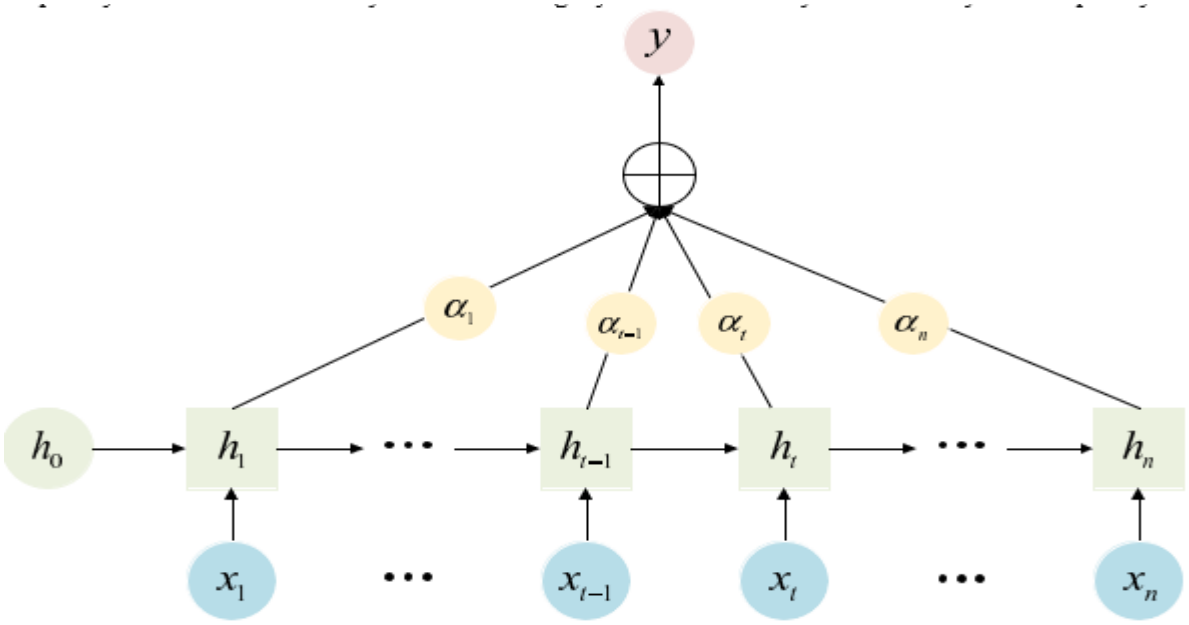

Figure 3.8 Structure of AM(D. Niu et al., 2022)

In the CNN-BiGRU-AM architecture, the input sequence moves through a 1D convolutional layer to capture local patterns. The resultant output of this layer is subsequently directed into a bidirectional GRU, which captures contextual details from both sequence directions. An attention mechanism is then imposed onto the BiGRU's outputs, facilitating the model in concentrating on particular sequence parts crucial for producing prediction. The following Figure 3.9 shows the structure of the CNN-BiGRU-AM model.

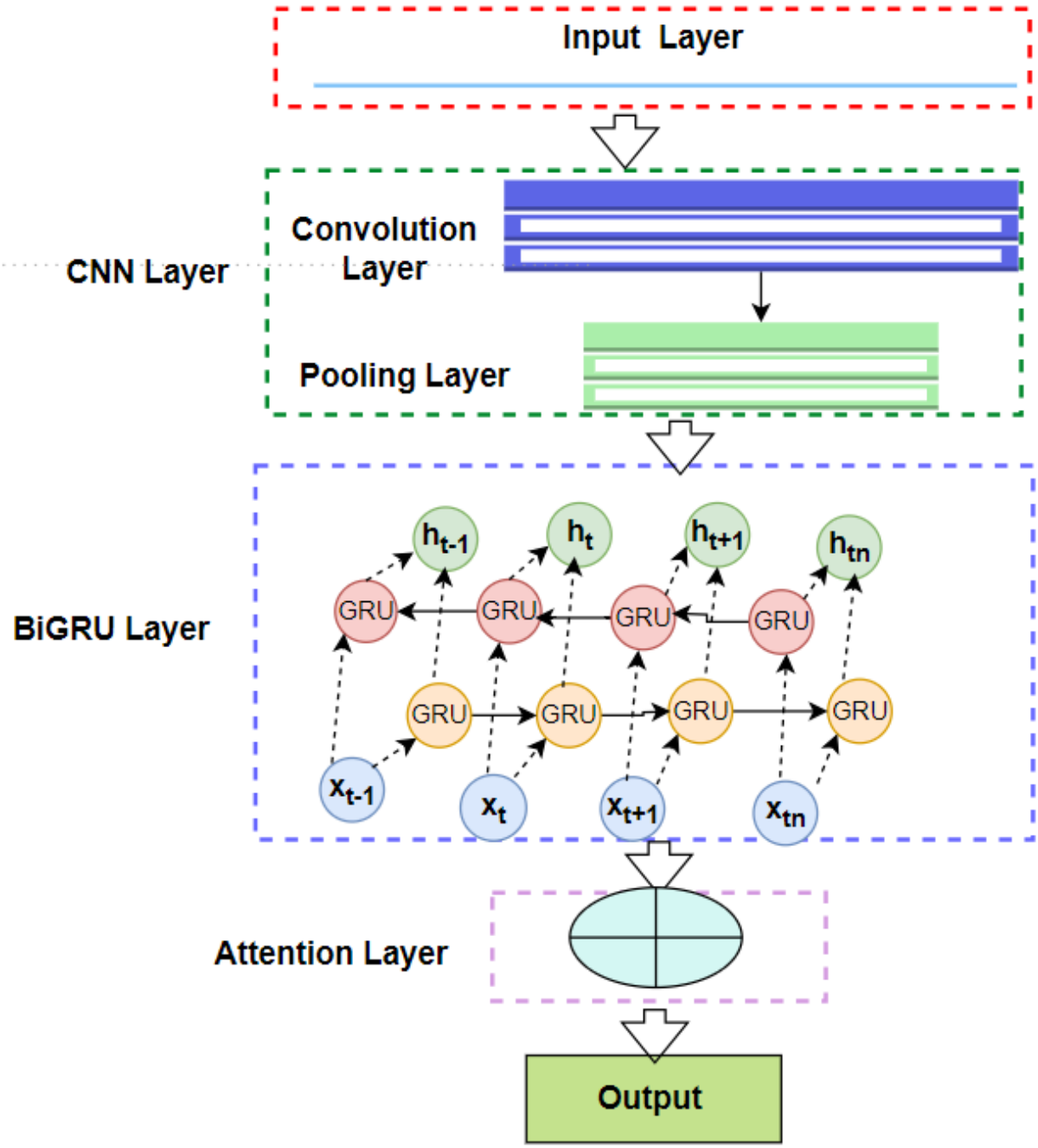

Figure 3.9 Structure of CNN-BiGRU-AM Model

3.5 Model Overfitting Handling

Model overfitting, which occurs when a created model does not generalize well for unknown data but is only well-optimized on the training set, is the most prevalent issue with neural networks. This phenomenon occurs when a model captures excessive noise and intricacies from the training data to such an extent that it adversely affects its performance on new, unseen data. To lessen model overfitting, various approaches are available(Ying, 2019). Those

are L1 and L2 regularization, dropout, data augmentation, early stopping, pruning, and esembling(Ying, 2019). Not only this, controlling the model's complexity can be crucial in preventing overfitting. This can be accomplished by decreasing the model's capacity, for as by lowering the number of layers or units per layer(Yu et al., 2019). Since there is no one size fits all we used combinations of L2 and dropout techniques for this study.

3.6 Finding Optimal Hyperparameters

The three layers of neural network-based models must be tuned with the right hyperparameters in order for them to be accurate. The differences in accuracy across the models are brought about by variations in tuning hyperparameters like number of iterations (epochs), activation function, batch size, learning rate, optimization algorithms, number of neurons in each layer, batch size within the same neural network and setting number of layers; Having too few layers in a model can lead to underfitting, whereas having too many layers can result in overfitting(Hu et al., 2021). Let us see issues that should be considered to tune hyperparameters.

The model learns more quickly when the learning rate is increased, but there is a chance that it will overshoot the minimum loss function and end up somewhere close to it. On the other hand, decreasing the learning rate improves the likelihood of discovering the minimum loss function. In order to achieve a lower learning rate, more epochs must be completed, which requires more time and memory space. When we come to batch size, reducing the batch size speeds up the learning process, but it increases the variance in the validation dataset's accuracy. Using a bigger batch size, on the other hand, slows down the learning process but leads to lesser variance in validation dataset accuracy. The training dataset needs to undergo multiple passes or epochs in order to sufficiently train the model. However, excessive epochs can result in overfitting. This indicated that optimizing hyperparameters is a challenging task that requires a thorough comprehension and intuition about the underlying model process. Furthermore, it can be a lengthy and error-prone procedure, frequently taking numerous iterations of trial and error before identifying an appropriate combination of hyperparameters. For easy hyperparameter optimization, there are two widely used techniques: grid search and random search(Zahran Aufa et al., 2020). Random search is often considered more efficient than grid search because it has the ability to sample important dimensions within the search

space, however, there is no assurance that it will yield optimal outcomes. But, with a small search space grid search gives the best result since it exhaustively trains and tests all the given network parameter combinations(Liashchynskyi & Liashchynskyi, 2019). For this study, we used a grid search optimizer algorithm for hyperparameter tuning.

3.7 Model Evaluation Techniques

To assess the success of our model and determine if our goals have been met, we trained the model on our dataset and tested it with a validation dataset split. In the traditional train-test split assessment method, the dataset is divided into two parts: one for training the model and the other for evaluating its performance. Nonetheless, this method can result in considerable fluctuations in performance estimation, particularly when the dataset is small or unrepresentative. To prevent this issue, k-fold cross-validation is employed to partition the dataset into k-folds, enabling the evaluation of a model's performance and generalization ability in a robust and impartial manner(Anguita et al., n.d.). In this study, we opted for the k-fold cross-validation evaluation approach due to its enhanced and more reliable assessment of the model's performance. By assessing the model across various data subsets, this technique allows us to evaluate its performance with respect to different classes. Furthermore, it mitigates the potential bias introduced by relying on a single division of data into training and testing sets. For evaluating the model based on the dataset, we utilize various evaluation metrics. These metrics include accuracy, recall, precision, and F1 score(Yacouby Amazon Alexa & Axman Amazon Alexa,2020). By analyzing these metrics, we can assess different aspects of the model's performance and determine its effectiveness in achieving the proposed model’s goals.

Accuracy

Accuracy is a metric that quantifies the percentage of correctly predicted data points out of the entire training dataset. It assesses how accurately the model has classified the data into their respective categories. The accuracy is determined using the subsequent formula:

$$Accuracy = \frac{TP + TN}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \qquad \textit{Equa. 3.13}$$

Were,

True Positive (TP): means that the real values and the predicted values are the same.

False Negative (FN): means, the model has inaccurately predicted a negative value for a positive actual value.

False Positive (FP): means, the model has erroneously predicted a positive value for a negative actual value.

True Negative (TN): means, the actual value is the same to the predicted value.

Precision

Precision is a metric that computes the proportion of true positives in relation to the total number of positive predictions. It evaluates the accuracy of the model's identification of positive instances by measuring how many of the predicted positives are genuinely true positives.

Recall

True positive divided by true positive and false negative equals to recall. In other words, recall assesses the model's ability to predict good outcomes.

F1-Score

The harmonic mean of precision and recall is used to get the F1 score. As a reminder, it is not the arithmetic mean. Let us see the formula of F1 score

$$F1\ score = 2 * \frac{Precision * Recall}{Precision + Recall} \qquad \textit{Equa. 3.14}$$

3.8 Human Evaluation Techniques

In our proposed model evaluation, human evaluation holds a pivotal role in understanding the chatbot's performance. While automated metrics offer quantitative data, human evaluation provides a qualitative analysis of the chatbot's interactions from a user-centered viewpoint. By seeking feedback from human reviewers, our intention is to go beyond technical accuracy benchmarks and ensure that the chatbot delivers interactions that are meaningful and beneficial to users.

3.8.1 Evaluation Criteria

We have outlined a comprehensive range of evaluation criteria that cover diverse aspects of the chatbot's performance. They include:

- ✓ **Response Time**: chatbot's responses timeliness.
- ✓ **User-friendly**: refers user's ease of use and understanding of the system in mind.
- ✓ **Usability**: the measure of how well it can be used by the intended users to achieve its goals with effectiveness, efficiency, and satisfaction.
- ✓ **Completeness**: The sufficiency of information offered in responses to meet user requirements.
- ✓ **Relevance**: How well the responses correspond to the user's query and intent.

3.9 Tools for Model Prototype

For the purpose of creating a model prototype within this study, we used the capabilities of the Flask web framework and the Telegram Bot API. These tools were instrumental in facilitating user interactions with our model, enabling dynamic conversations. Consequently, we were able to gather user feedback, providing us with a means to evaluate our model's performance. Users can communicate with the chatbot by using the Telegram app, as well as by accessing it through a web browser on a desktop or mobile device. Flask is a lightweight web framework that's well-suited for creating web interfaces. This web interface can serve as a means of interacting with the chatbot. The Telegram Bot API provides a convenient and user-friendly platform for interacting with our model which allows us to quickly gather user input, test different conversational flows, and receive real-time feedback on our chatbot's performance.

3.10 Implementation Tools

Development Tools

To execute the suggested study, several hardware and software tools are necessary. The following tools were utilized in the development of the proposed model. Here are the tools employed to create the model.

Software tools

During the development of the proposed model, various types of software were employed for different tasks. The following is a list of the software tools utilized in this study:

Development Environment

Anaconda

Anaconda is a comprehensive platform designed for Python and other programming languages used in scientific computing applications such as machine learning, data science, and large data processing. It includes a range of applications like Spyder, JupyterLab, and Jupyter Notebook.

Jupyter Notebook

Jupyter Notebook is a tool that allows the creation and sharing of documents containing live code, equations, visualizations, and text. It operates as an interactive computing environment accessible via the web, making it useful for organizing workflows in machine learning development.

Google Colab

Colab is valuable for tasks such as machine learning, data analysis, and education because it allows individuals to write and execute Python code directly in their web browsers, offering accessibility and convenience. It has the capability of serving different teams to edit their code at the same time. Another advantage of colab is, it includes free resources such as Tensor Processing Unit (TPU) and Graphical Processing Units (GPUs) and does not require any configuration.

Programming Language Tools

Python 3.9.13

Python is a programming language known for its high-level, object-oriented, well-organized, and easily readable code. It is a general-purpose language created for use in a variety of applications, including automation and data science, and has many libraries for machine learning tasks. The following are some of the libraries used in the process of developing the model.

- ✓ NLTK (Natural Language Tool Kit)
- ✓ Keras

- ✓ TFlearn
- ✓ Tensorflow
- ✓ Scikit-learn

Hardware Tools

For this study, laptop computer with 8GB RAM, 1TB internal disk, processor 2.40GHz and Desktop computer with 8GB RAM, 1TB internal disk, processor 3.10GHz are used.

CHAPTER FOUR

4. Proposed Model

4.1 Chapter overview

In this chapter we discuss the architecture of the proposed model, and various steps involved in implementing each procedure during the chatbot development.

4.2 Architecture of the Model

The proposed chatbot architecture comprises several stages, including data preprocessing including data preparation, feature extraction, model training and testing and selection of the best-performing model. The core component of the system is the training model, which utilizes deep learning techniques to process the preprocessed data and generate responses. In this study, we have employed BiLSTM, BiGRU, CNN-BiLSTM and CNN-BiGRU-AM known for their exceptional performance in language-related and sequential tasks. Fasttext and wor2vec word embedding methods are used for feature extraction purpose. The architecture also incorporates user interaction with the system as shown in figure 4.1.

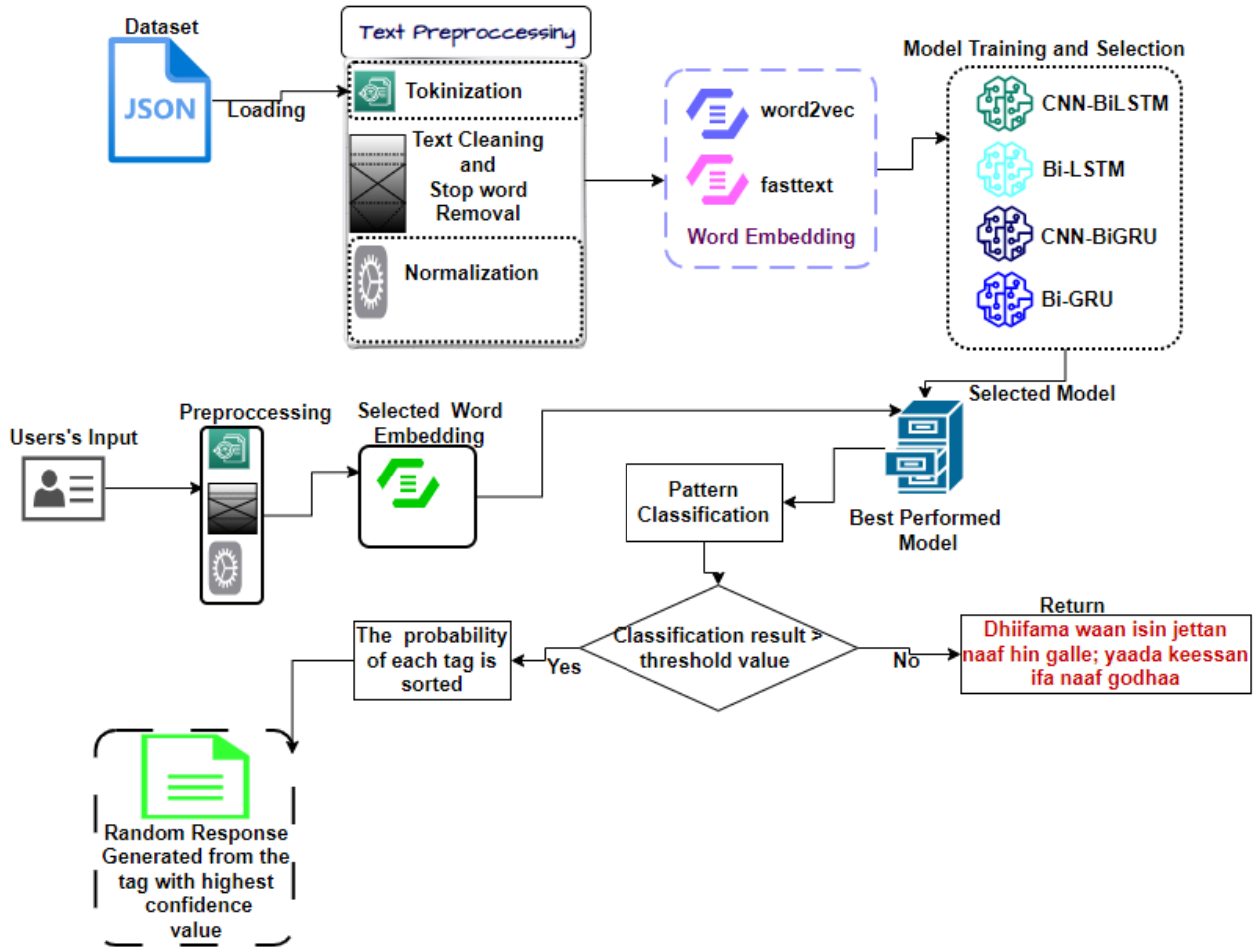

Figure 4.1 Proposed Model Architecture

To build the proposed model, we first collect and prepare data. The gathered dataset is formatted in JSON, with three elements: tag, patterns, and responses. Each tag represents a category that includes user queries or features as patterns and specific context-based answers as responses. After organizing the data in the JSON file, we perform text preprocessing. During the text preprocessing stage, several processes are applied, including tokenization, cleaning, stop word removal, and normalization. Tokenization involves splitting the sentences within the patterns into individual tokens or words. Cleaning punctuation marks and special characters from the resulting word lists. The other thing important to take into account is that, users frequently express the same word in varying ways, and machine learning models also interpret those words differently. Text normalization aids models in recognizing identical words written in various formats. In study, normalization is carried out by converting all words to lowercase to ensure uniformity.

The most common terms found in any language, such as pronouns, prepositions, conjunctions, articles, and particles, need to be separated from the dataset. To accomplish this, the Afaan Oromo stop words were gathered, verified, and stored in the NLTK stop word corpora under the file name ‘afaan_oromo’. Subsequently, these stop words were loaded and help to remove stop words from the dataset. If a word is present in the list of Afaan Oromo stop words, it is removed; otherwise, it proceeds to subsequent processes.

The preprocessed data is utilized for model training, but it cannot be directly entered into deep learning models since they only accept numerical data. In order to make the data compatible with deep learning models and extract contextual information from words, word embedding methods are applied. In this model, two types of word embeddings, namely word2vec and fasttext, are utilized. Word2vec is a word embedding technique that captures the contextual meaning of a word, while fasttext captures sub-word information, specifically n-grams of letters. These embeddings help represent words as numerical vectors, enabling deep learning models to effectively process and understand the data.

In this study, two variants of RNN, namely Bi-LSTM and Bi-GRU, and CNN-BiLSTM and CNN-BiGRU, are employed in order to identify the best-performing model on our dataset. The user's query undergoes text preprocessing and represented using selected feature extraction techniques from word2vec and fasttext. By taking the query, the model performs intent classification, utilizing a threshold value. Threshold is a parameter that is used to determine whether the predicted output from the model should be considered as valid or not. It's used to make a decision based on the confidence of the model's prediction. The proposed model produces an output that represents the model's confidence scores (probability scores) for each class. Then the maximum probability or confidence score produced by the model compared with the predefined threshold value. If the classification result surpasses the threshold value, the intent is classified into a specific tag based on the probability associated with that tag then the response is generated randomly from the responses contained in that tag.

4.3 Loading the Data Implementation

To begin implementing the proposed model, we first load the prepared dataset into the development environment. We have successfully loaded the dataset in JSON format into the

Jupyter environment for further processing. Figure 4.2 provides a code snippet that demonstrates the loading process.

```
#Loading the dataset
intents = json.loads(open('path/coffeedata.json',"r",encoding='utf-8').read())
```

Figure 4.2 Loading Dataset Code Snippet

4.4 Tokenization Implementation

Once the dataset is loaded, we proceed to tokenize the data to make it suitable for subsequent processes. To split the texts in the dataset, we utilized the nltk.tokenize library. Figure 4.3 displays the code snippet that demonstrates the implementation of tokenization on the dataset.

```
#tokenization
sentences = [word_tokenize(text) for text in data["inputs"].tolist()]
```

Figure 4.3 Tokenization Code Snippet

4.5 Stop Word Removal and Normalization

After tokenization, we proceeded with cleaning, stop word removal, and normalization of the dataset. In the cleaning process, we utilized the 're' library to clean special characters and punctuation marks from the text. For stop word removal, we incorporated Afaan Oromo stop words into the nltk directory and removed them from the dataset. And, in the normalization process, we converted all the words in the dataset to lowercase. Figure 4.4 provides the code snippet that demonstrates the implementation of cleaning, stop word removal, and normalization.

```
filtered_sentences = []
for sentence in sentences:
    filtered_sentence = []
    for word in sentence:
        # normalizing the word by converting word to lowercase
        word = word.lower()
        # normalization or removing special characters
        word = re.sub(r'[^a-zA-Z0-9]', '', word)
        # Apply stop word removal
        if word not in Ao_stopwords:
            filtered_sentence.append(word)
    filtered_sentences.append(filtered_sentence)
```

Figure 4.4 Normalization and Stop Word Removal Implementation

4.6 Word2vec Implementation

To train the preprocessed data using word2vec word embedding for converting words into vector representations, we utilized the 'gensim' library, which includes the word2vec module. In our implementation, we specifically trained a skip-gram word2vec model. Figure 4.5 shows the code snippet that demonstrates the implementation of training the skip-gram word2vec model using the 'gensim' library.

```
#loading word2vec module from gensim library
from gensim.models import Word2Vec
```

```
#training skip-gram word2vec model
wv_model = Word2Vec(filtered_sentences, sg=1, window=5, min_count=1, workers=4, vector_size=100)
```

Figure 4.5 Skip-gram Word2vec Implementation

The word2vec model is saved using the 'save' function and loaded using the 'load' function. Once the word2vec model is saved we load and use it in deep learning models as an input. The following pictures shows the line of codes to save and load the word2vec model

```
wv_model.save('path/word2vec_model.bin')
wv_model = Word2Vec.load('path/word2vec_model.bin')
```

Figure 4.6 Save and Load Word2vec

Because our dataset has a high number of dimensions, it's challenging to visualize how data points relate to each other. To address this, we employed t-SNE (t-Distributed Stochastic Neighbor Embedding), a technique designed for visualizing high-dimensional data in a lower-dimensional format. Figure 4.7 below illustrates the data's similarities by condensing it into a 2D representation using t-SNE.

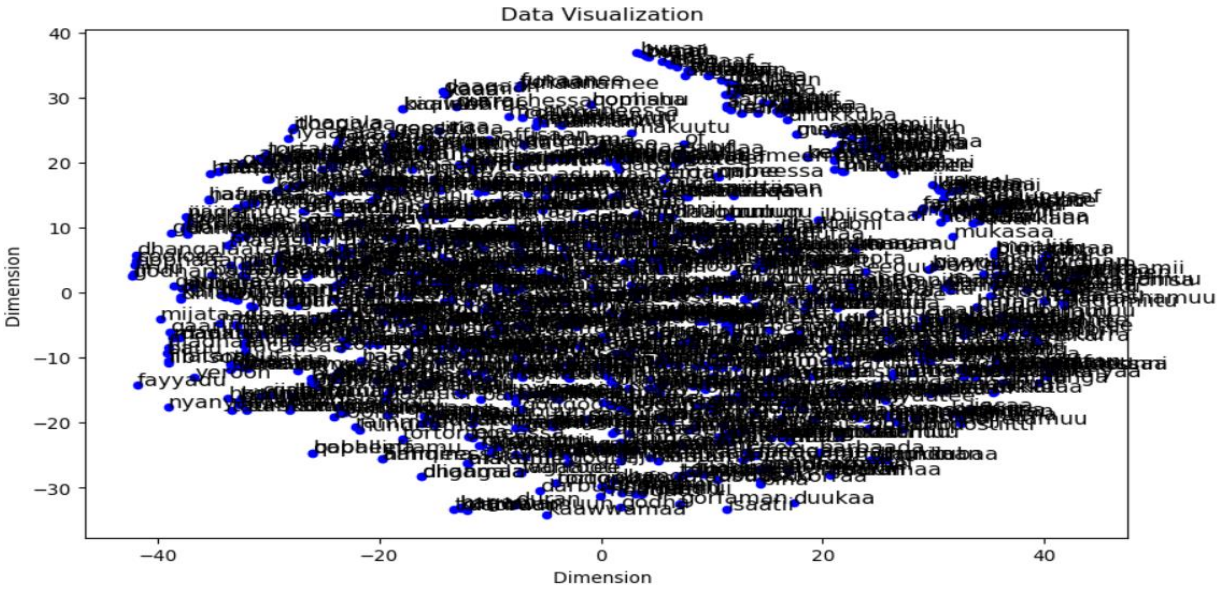

Figure 4.7 Data Visualization Representation from Word2vec Word Embedding

4.7 Fasttext Implementation

To implement fasttext word embeddings, we utilized the gensim library. The implementation process for fasttext is similar to that of word2vec, despite the differences in the information they capture. Figure 4.8 shows how the implementation of fasttext appears.

```
#training fasttext
from gensim.models import FastText
fasttext_model = FastText(sentences=filtered_sentences, vector_size=100, window=5, min_count=1, workers
```

Figure 4.8 Fasttext Implementation

4.8 Proposed Deep Learning Model Implementations

In this study, we implemented two types of recurrent neural network (RNN) variants: Bi-LSTM and Bi-GRU and their hybrid with CNN model: CNN-BiLSTM and CNN-BiGRU-AM which are suitable for sequential data using the Keras API. These models were trained on our dataset to learn its underlying features. Firstly, the models are initialized with the 'sequential' function. Then, the input layer is added to specify the shape of the input data, which represents the length of the input sequence. The embedding layer is included to handle the vocabulary size obtained from the word embedding models. Following the embedding layer, a particular deep learning layer is added to capture long-term dependencies between words in the dataset.

The addition of the flatten layer is intended to transform the output from the preceding layers into a one-dimensional vector by flattening it. The hidden layer is added to perform a mathematical operation that involves multiplying the input values by weights and passing the result through activation function. This process allows the hidden layer to transform the inputs and produce an output based on the learned weights and the activation function. Because of its computational efficiency, we opted for the Rectified Linear Unit (ReLU) activation function in the hidden layer. Because our model is designed for multiclass classification, we implemented the 'softmax' activation function in the output layer and utilized the 'categorical cross-entropy' function as the loss function. For handling overfitting, we configured the model with regularization and dropout techniques.

4.8.1 Bi-LSTM Model Implementation

Like LSTM, bidirectional LSTM also captures the dependencies in the dataset. However, Bi-LSTM enhances LSTM by introducing an additional LSTM layer that operates in the opposite direction, analyzing the input sequence from future to past. Figure 4.9 shows the code to implement the BI-LSTM model.

```
#initializing the model
model = tf.keras.Sequential()
#input shape specification
model.add(Input(shape=(input_shape,)))
#adding Embeedding layer
model.add(Embedding(unique_words+1, 100, input_length=input_shape))
#the Bi-LSTM layer with its units
model.add(Bidirectional(LSTM(200, return_sequences=True)))
```

Figure 4.9 Bi-LSTM Implementation Code

4.8.2 BiGRU model implementation

GRUs have been designed to simplify the gating mechanisms, including the forget gate, input gate, and output gate, which are essential components of RNNs and LSTMs. Like LSTMs, GRUs have the ability to address challenges related to long-term memory and gradient propagation during backpropagation. However, GRU demonstrates superior performance compared to LSTM, leading to enhanced training efficiency and accelerated operational effectiveness(Zhou et al., 2023). Additionally, when working with smaller to medium-sized datasets, GRUs have been proven to outperform LSTMs(Bhuvaneswari et al., 2019).

Therefore, in this study we applied BiGRU on the collected dataset. The following figure 4.10 shows sample code of BiGRU implementation.

```
#initializing the model
model = tf.keras.Sequential()
#input shape specification
model.add(Input(shape=(input_shape,)))
#adding Embeedding layer
model.add(Embedding(unique_words+1, 100, input_length=input_shape))
#the BiGRU layer with its units
model.add(Bidirectional(GRU(300, return_sequences=True)))
```

Figure 4.10 BiGRU Model Code Implementation

4.8.3 CNN-BiLSTM Model Implementation

Combining CNN with BiLSTM creates a hybrid model that leverages the CNN's capability to extract local features and the BiLSTM's capacity to capture long-term dependencies(Thakur & Monga, 2022). CNN deep learning model's capacity to learn sequence correlations and capture word relationship is limited. The BiLSTM model, on the other hand, has difficulty extracting local features in parallel. The hybrid model, however, is intended to overcome the drawbacks of each individual model by combining the convolutional layer with BiLSTM. To implement it, on the top of BiLSTM layer 1D convolutional layer is added with 128 filters and 5 kernels as shown in the figure 4.11

```
#initializing the model
model = tf.keras.Sequential()
#input shape specification
model.add(Input(shape=(input_shape,)))
#adding Embeedding layer
model.add(Embedding(unique_words+1, 100, input_length=input_shape))
#Adding convolutional layer
model.add(Conv1D(128, 7, activation='relu'))
model.add(MaxPooling1D(pool_size=5))
#the BiLSTM layer with its units
model.add(Bidirectional(LSTM(200, return_sequences=True)))
```

Figure 4.11 CNN-BiLSTM Model Implementation

4.8.4 CNN-BiGRU-AM Model Implementation

The combination of CNNs in the CNN-BiGRU architecture allows for the capture of local patterns and spatial relationships, while the inclusion of BiGRUs enables the modeling of sequential dependencies and context. Additionally, using attention mechanisms helps models to efficiently focus on crucial information, comprehend the connections between various components, handle long-distance dependencies, and align sequences(Z. Niu et al., 2021). The CNN-BiGRU model can focus on the most informative characteristics by paying to pertinent elements of the input sequence with the assistance of the attention mechanism. The performance of the model as a whole may be enhanced by this selective emphasis, which helps with improved feature extraction. In this work, we employed the attention mechanism for its state-of-the-art performance in language models(Vaswani et al., 2017). The following figure 4.12 shows the implementation code of CNN-BiGRU custom attention mechanism

```
class SelfAttention(Layer):
    def __init__(self):
        super(SelfAttention, self).__init__()

    def build(self, input_shape):
        super(SelfAttention, self).build(input_shape)

    def call(self, x):
        q = x
        k = x
        e = tf.matmul(q, k, transpose_b=True)
        a = tf.nn.softmax(e, axis=2)
        output = tf.matmul(a, x)  # Weighted sum using attention weights
        return output

# Initializing the model
model = tf.keras.Sequential()

# Input shape specification
model.add(Input(shape=(input_shape,)))

# Adding Embedding layer
model.add(Embedding(unique_words + 1, 100, input_length=input_shape))

model.add(Conv1D(128, 7, activation='relu'))
model.add(MaxPooling1D(pool_size=5))
#Adding Bidirectional LSTM layers
model.add(Bidirectional(GRU(300, return_sequences=True)))
model.add(Bidirectional(GRU(100, return_sequences=True)))

# Adding Self-Attention layer
model.add(SelfAttention())
```

Figure 4.12 CNN-BiGRU with AM Implementation

4.8.5 Hyperparameter Tuning Implementation

As mentioned in section 3.6, finding the optimal hyperparameter combination to enhance model performance is challenging. To address this, we utilized the grid search hyperparameter optimizer algorithm to determine the best hyperparameter combinations. Figure 4.13 illustrates the implementation code of the grid search algorithm aimed at finding these optimal combinations.

```python
# defining model creation function
def create_model(units, epochs, l2_reg, learnig_rate, droup_rate):
    model = Sequential()
    model.add(Input(shape=(input_shape,)))
    mmodel.add(Embedding(unique_words + 1, 100, input_length=input_shape))
    model.add(Conv1D(128, 7, activation='relu'))
    model.add(MaxPooling1D(pool_size=5))
    model.add(Bidirectional(LSTM(units, return_sequences=True, activation ='relu')))
    model.add(Bidirectional(LSTM(100, return_sequences=False, activation ='relu')))
    model.add(Flatten())
    model.add(Dropout(droup_rate))
    model.add(Dense(units=30, activation='relu', kernel_regularizer=l2(l2_reg)))
    model.add(Dense(units=output_length, activation='softmax'))
    optimizer = tf.keras.optimizers.Adam(learning_rate=learning_rate)
    model.compile(loss="sparse_categorical_crossentropy", optimizer=optimizer, metrics=['accuracy'])
    return model

# Create the KerasClassifier wrapper for Scikit-learn
model = KerasClassifier(build_fn=create_model)

# Define the hyperparameters grid
param_grid = {
    'units': [200, 300, 500],
    'epochs': [30, 60, 50, 80, 100],
    'l2_reg': [0.01, 0.001],
    'learning_rate': [0.01, 0.001, 0.0001],
    'droup_rate': [0.3, 0.5],
}
```

Figure 4.13 Grid Search Implementation Sample Code Snippet

4.8.6 K-fold Cross Validation Implementation

In section 3.7, we explored the impact of the traditional train-test splitting approach on model performance. To address this issue, one solution is the utilization of k-fold cross-validation. Figure 4.14 presents the implementation code for 5-fold cross-validation.

```python
# Perform K-fold cross-validation
fold = 1
for train_indices, val_indices in kfold.split(X):
    print("Training on Fold:", fold)

    # Split data into training and validation sets
    x_train, X_val = X[train_indices], X[val_indices]
    y_train, y_val = y[train_indices], y[val_indices]

    # Clear previous model
    tf.keras.backend.clear_session()

    # Compile the model
    model.compile(loss='sparse_categorical_crossentropy', optimizer=optimizer, metrics=['accuracy'])

    # Train the model
    history=model.fit(x_train, y_train, epochs=80, batch_size=64, verbose=1, validation_data=(X_val, y_val))

    # Evaluate the model on the validation set
    y_val_pred_probs = model.predict(X_val)
    y_val_pred = np.argmax(y_val_pred_probs, axis=1)
    accuracy = accuracy_score(y_val, y_val_pred)
    accuracies.append(accuracy)
    precision = precision_score(y_val, y_val_pred, average='macro')
    recall = recall_score(y_val, y_val_pred, average='macro')
    f1 = f1_score(y_val, y_val_pred, average='macro')
    precisions.append(precision)
    recalls.append(recall)
    f1_scores.append(f1)

    print("Validation Accuracy:", accuracy)
    fold += 1

# Calculate and print average accuracy across all folds
avg_accuracy = np.mean(accuracies)
avg_precision=np.mean(precisions)
avg_recalls=np.mean(recalls)
avg_f1_scores=np.mean(f1_scores)
print("Average Accuracy:", avg_accuracy)
print("Average precision:", avg_precision)
print("Average recalls:", avg_recalls)
print("Average f1_scores:", avg_f1_scores)
```

Figure 4.14 K-fold Cross Validation Code Snippet

CHAPTER FIVE

5. RESULT AND DISCUSSIONS

5.1 Overview of the Chapter

This chapter describes the findings of an experiment conducted to evaluate multiple deep learning models proposed for a coffee production assistant. The chapter also covers how the proposed study aligns with the research questions and provides a prototype of the intended model. The dataset for this study consists 2,367 patterns, each containing user intents. These patterns are associated with specific tags, and there are 123 different tags (classes) in total. Each tag comprises patterns and corresponding responses.

5.2 Train, Validation Dataset Split

To evaluate the models' performance, the dataset is split into training and validation sets. However, randomly partitioning the original data in this manner can potentially reduce the number of samples available for model training. The reduction in the amount of training data can influence the model's performance. Additionally, the outcomes of the model can be influenced by using some percent of samples for the training and the other for validation sets(Prusty et al., 2022). As previously explained in section 3.7, the k-fold cross-validation method tackles this problem by dividing the dataset into K segments, employing one segment as the test set, and utilizing the remaining data as the training set in each iteration. In k-fold cross-validation, there is no strict rule for choosing the optimal value of 'k.' some of Previous studies (Guta Tolasa, 2022; Hettiarachchi & Gamini, 2023; Prusty et al., 2022) have compared the model's performance using 10-fold and 5-fold cross-validation and found better results with 5-fold cross-validation. In this study, we also implemented a 5-fold cross-validation approach, where the model is trained on four of the folds, while the remaining one is designated for testing.

5.3 Hyperparameter Tuning

In section 3.6, we explained the approach to finding the optimal hyperparameter combination. For this study, we opted for the grid search algorithm over the random search algorithm due to its capability to exhaustively train and test all combinations of the given hyperparameters, leading to improved results. For this study, the grid search finds optimal hyperparameter combination from Epochs (30, 50, 60, 80, 100), Batch size (32, 64, 128), number of units (200,

300 500) L2 regularization (0.01, 0.001), learning rate (0.01, 0.001, 0.0001), dropout rate (0.3, 0.5). In our model, we utilize the Softmax activation function for the output layer, which is well-suited for multiclass classification problems as noted by Sharma et al. (2020). Additionally, for the hidden layers, we adopt the Relu activation function, which is one of the most commonly used nonlinear activation functions according to Sivri et al. (2022). The following Table 5.1 shows the optimal hyperparameter combination of all proposed model.

Table 5.1 Hyperparameter for Proposed Models

Models	**# of units**	**Epochs**	**L2 regularization**	**Learning rate**	**Batch-size**	**Dropout rate**
BiLSTM	200	80	0.001	0.001	64	0.5
BiGRU	300	80	0.001	0.001	64	0.3
CNN-BiLSTM	200	50	0.01	0.01	32	0.5
CNN-BiGRU-AM	300	50	0.001	0.01	32	0.5

5.4 Proposed Models Results

In this study, we introduced and evaluated two variants of the RNN model, namely BiLSTM and BiGRU. Additionally, the hybrids of these RNN variants with CNN model to create CNN-BiLSTM and CNN-BiGRU with attention mechanism (AM) integrated into it. We employed two feature extraction techniques: word2vec and fasttext. The models were trained on our dataset with both feature extraction techniques. In the subsequent sections, we discuss a detailed analysis of the performance and outcomes of each model individually.

5.4.1 BiLSTM Model Result

After determining the optimal network hyperparameters for BiLSTM, we trained the BiLSTM model on our preprocessed dataset. The model was trained using two feature extraction techniques: word2vec and fasttext. Figure 5.1 shows the plot of training and validation curve of loss, and accuracy for BiLSTM model.

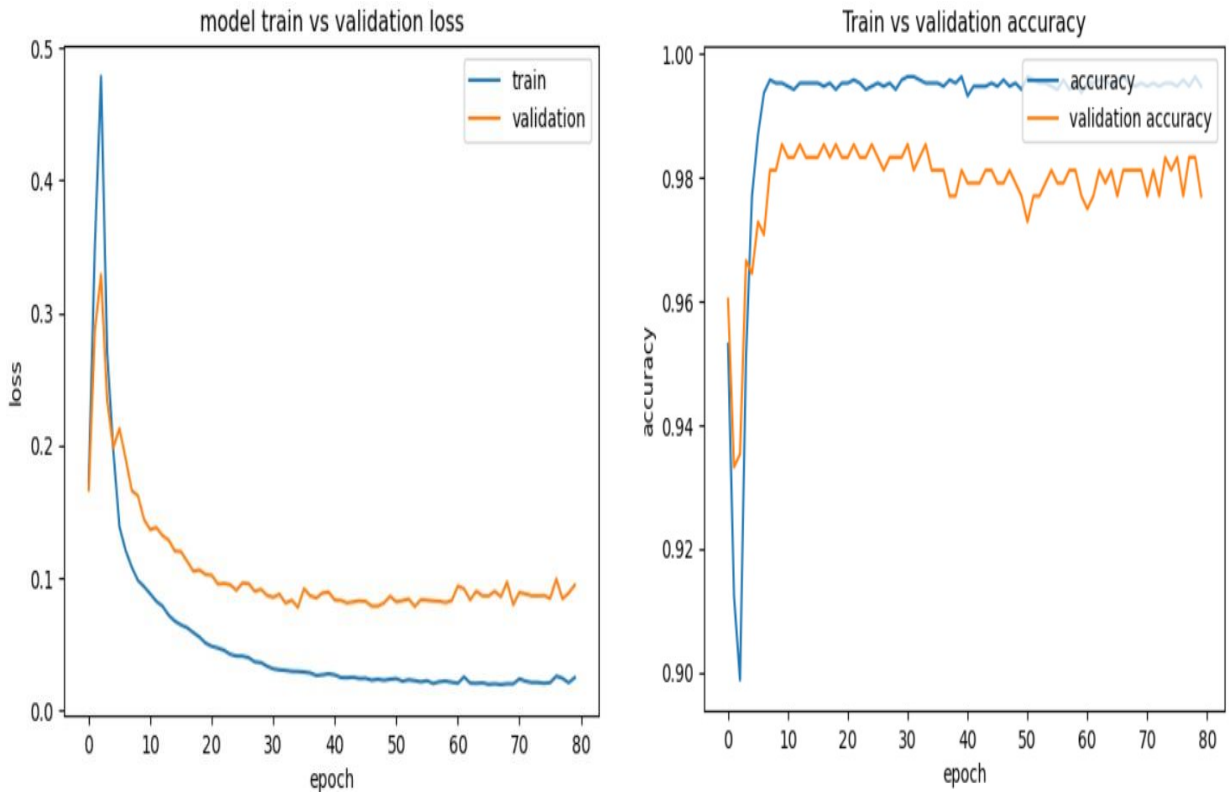

Figure 5.1 Train and Validation Curve for Loss and Accuracy of BiLSTM

The summary of the BiLSTM model's performance with both word2vec and fasttext feature extractions is presented in Table 5.2.

Table 5.2 BiLSTM Model Result

Model	Word2vec				Fasttext			
	Accuracy	**Precision**	**Recall**	**F1 Score**	**Accuracy**	**Precision**	**Recall**	**F1 Score**
BiLSTM	88.15%	85.62%	85.73 %	84.82 %	87.56%	85.88%	85.16 %	84.58 %

5.4.2 Result of CNN-BiLSTM Model

While CNN may extract local features, the BiLSTM model can capture long dependencies in both the forward and backward directions of the sequence. In this work, we combined the two models and train it on our dataset. The model trained using both word2vec and fastetxt feature extractions. The model's accuracy with word2vec is 88.48%and it gave us accuracy of 86.91%

with fastetext. Figure 5.2 illustrates the train and validation curve of loss, and for CNN-BiLSTM model.

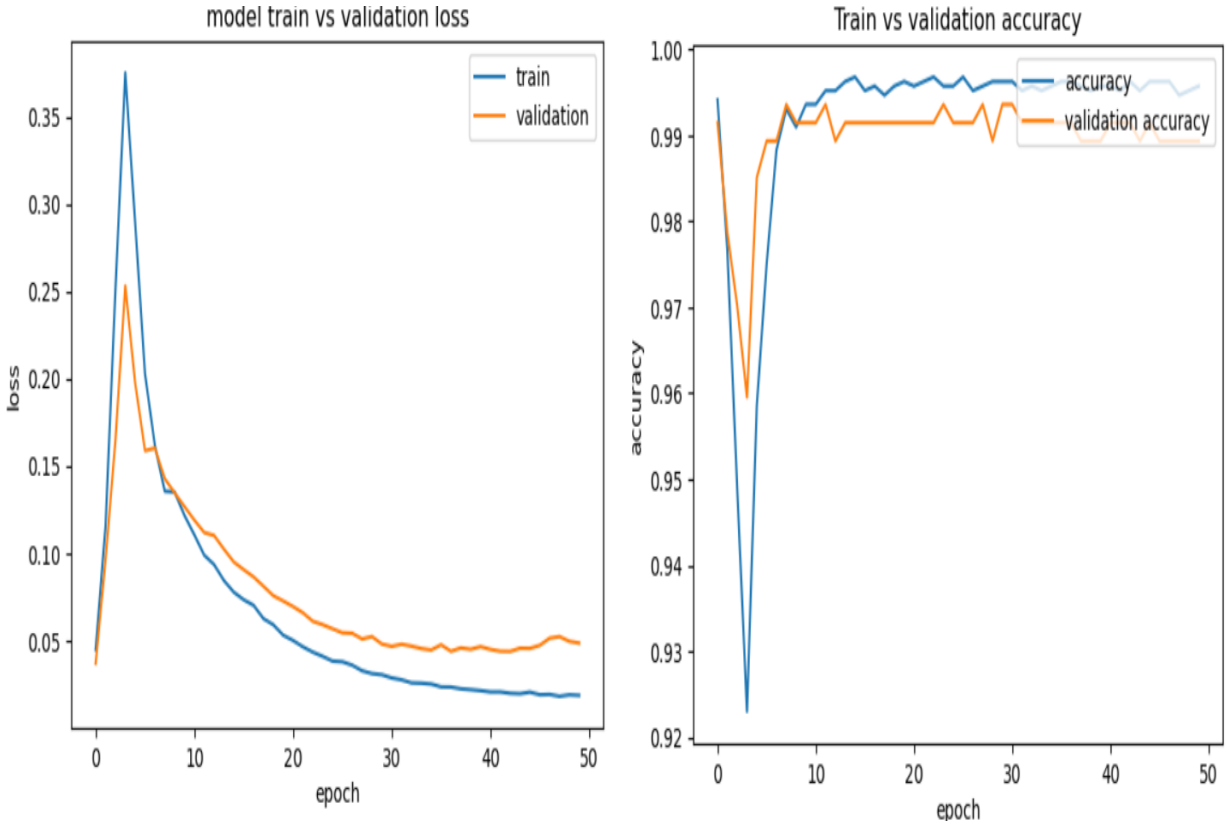

Figure 5.2 Train and Validation Curve of Loss and Accuracy for CNN-BiLSTM

The following Table 5.3 summarize the result we got from CNN-BiLSTM model.

Table 5.3 CNN-BiLSTM Model Result

Model	Word2vec				Fasttext			
	Accuracy	Precision	Recall	F1 Score	Accuracy	Precision	Recall	F1 Score
CNN-BiLSTM	88.48%	86.75%	86.37 %	85.63 %	86.91%	84.37%	84.09 %	83.98 %

5.4.3 BiGRU Model Result

GRU is another variant of RNN which is developed to overcome the draw backs of RNN model. Rather than using a traditional unidirectional GRU, it is more beneficial to use BiGRU,

which can leverage information from both the forward and backward directions. The BiGRU computes the output value by combining insights from both the forward and backward GRU layers. With the implementation of the BiGRU model, the study achieved an accuracy of 88.41% with word2vec and 87.68% with fasttext feature extraction. Figure 5.3 shows train and validation curve of loss and accuracy for BiGRU model.

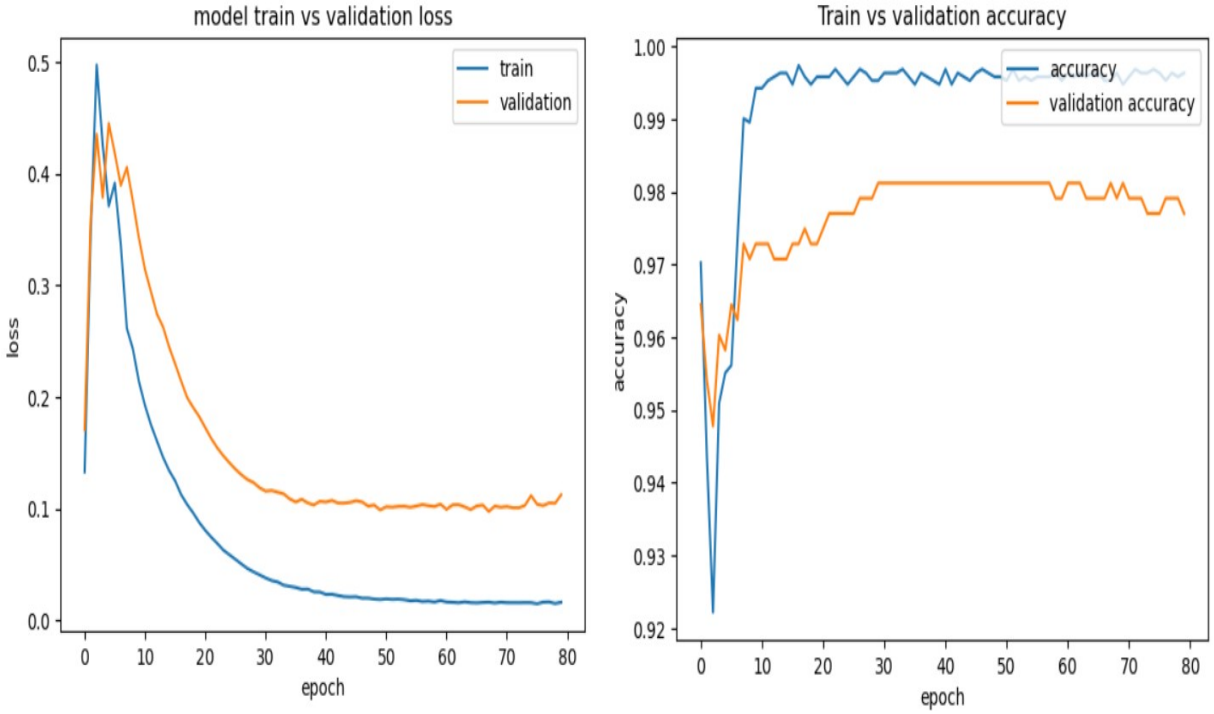

Figure 5.3 Train and Validation Curve of Loss and Accuracy for BiGRU Model

The following Table 5.4 display the result of the study attained using BiGRU model.

Table 5.4 Result of BiGRU Model

Model	Word2vec				Fasttext			
	Accuracy	**Precision**	**Recall**	**F1 Score**	**Accuracy**	**Precision**	**Recall**	**F1 Score**
BiGRU	88.41%	85.87%	84.96 %	84.37 %	87.68%	85.68%	85.59 %	84.45 %

5.4.4 CNN-BiGRU-AM Model Result

In this study, we suggested a hybrid CNN-BiGRU model to take use of the strengths of both models and enhance the model's performance. We also implemented an attention mechanism to the model. By focusing just on the most important input elements, attention mechanisms enhance deep learning models, improving prediction accuracy and computing efficiency. They serve as a spotlight by prioritizing and emphasizing essential information, improving the model's overall performance[1]. Using this model, this study achieved an accuracy of 91.01% with word2vec and 89.47% with fasttext feature extraction methods. Figure 5.4 illustrates train and validation plot of loss and accuracy for CNN-BiGRU-AM model.

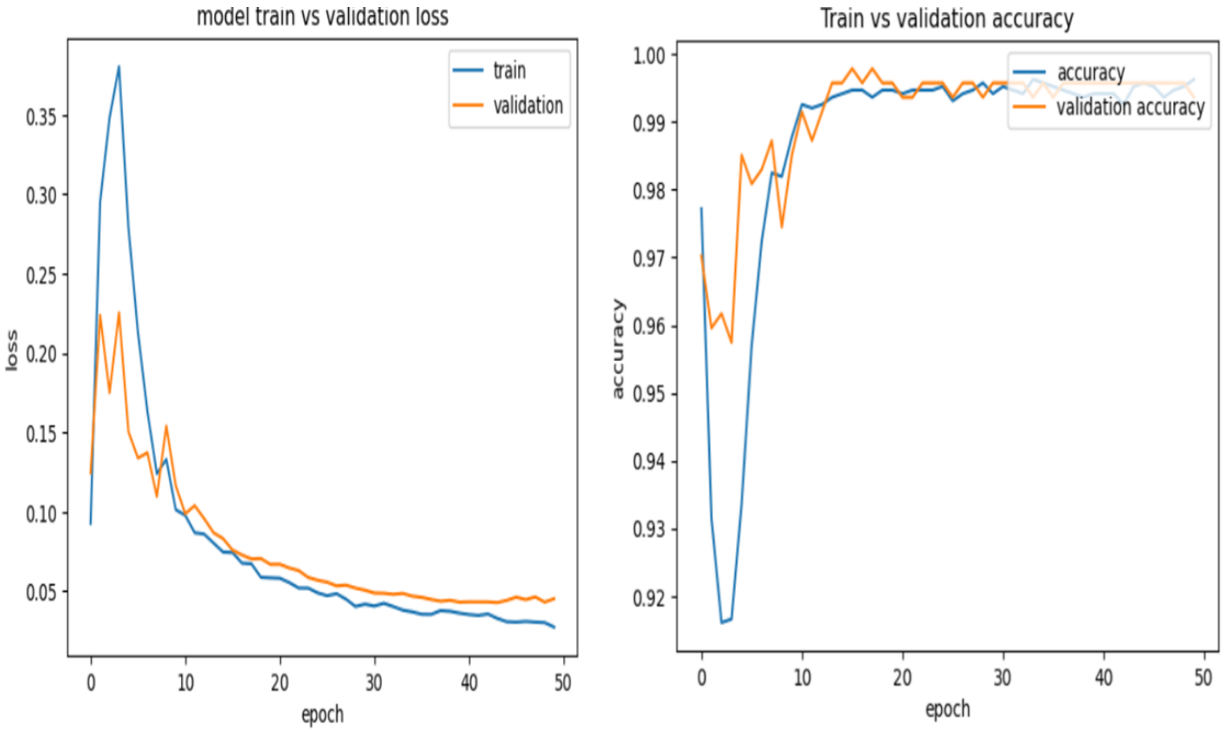

Figure 5.4 Train and Validation Curve of Loss and Accuracy for CNN-BiGRU-AM Model

The following Table 5.5 shows the result of CNN-BiGRU-AM model in this study and the accuracy values at each fold.

[1] https://www.analyticsvidhya.com/

Table 5.5 The Result of CNN-BiGRU-AM Model

Model	Word2vec				Fasttext			
	Accuracy	***Precision***	***Recall***	***F1 Score***	***Accuracy***	***Precision***	***Recall***	***F1 Score***
CNN-BiGRU-AM	*91.01%*	*89.13%*	*89.15%*	*88.49%*	*89.47%*	*87.98%*	*88.09%*	*87.25%*
	5-folds'accuracies with word2vec							
	1st	***2nd***	***3rd***	***4th***	***5th***			
	90.49%	*92.08%*	*91.48%*	*89.71%*	*91.29%*			

To assess how the inclusion of an attention mechanism affects our model's performance on our dataset, we integrated an attention mechanism into all the models we introduced in this study. All of the models performed better after having an attention mechanism added to them. The overview of the models' results with and without an attention mechanism is presented in Table 5.6 below.

Table 5.3 Models' Results Summary for all Models

	Models	Word2vec				Fasttext			
		Accuracy	**Precision**	**Recall**	**F1 Score**	**Accuracy**	**Precision**	**Recall**	**F1 Score**
With AM	***BiLSTM***	89.03%	86.97%	87.59 %	86.36 %	88.79%	86.00%	85.71 %	84.87 %
	CNN-BiLSTM	88.72%	86.85%	85.95 %	85.78 %	86.96%	84.78%	84.37 %	83.49 %
	BiGRU	88.71%	87.51%	86.61 %	85.94 %	88.12%	85.30%	84.81 %	83.92 %
	CNN-BiGRU	**91.01%**	89.13%	89.15 %	88.49	89.47%	87.98%	88.09 %	87.25 %
	BiLST	88.15%	85.62%	85.73	84.82	87.56%	85.88%	85.16	84.58

	M			%	%			%	%
Without AM	***CNN-BiLSTM***	88.48%	86.75%	86.37 %	85.63 %	86.91%	84.37%	84.09 %	83.98 %
	BiGRU	88.41%	85.87%	84.96 %	84.37 %	87.68%	85.68%	85.59 %	84.45 %
	CNN-BiGRU	89.11%	87.34%	87.10 %	86.48 %	88.39%	86.41%	86.40 %	85.40 %

As observed in the outcomes presented in Table 5.6, the models that incorporated an attention mechanism demonstrated superior performance compared to those without attention. In this study all the proposed models trained on our dataset with two feature extraction techniques: word2vec and fasttext. As we have discussed under section 3.3 word2vec has the limitation of OOV problem which is addressed by fasttext. Nonetheless, it's important to note that fasttext doesn't grasp the contextual meaning of a word from its neighboring words. Instead, it focuses on capturing sub-word information, such as letter n-grams. On our dataset the results of the models with word2vec feature extraction better when it is compared to the results the models with fasttext. This shows that in the chatbot model, understanding context of the word is crucial for generating relevant response to the user's query. And on our dataset, CNN-BiGRU-Am model with word2vec feature extraction outperformed the other models with accuracy of 91.01% as we can see from the above Table 5.6. The following chart shows the accuracies of proposed models for this study.

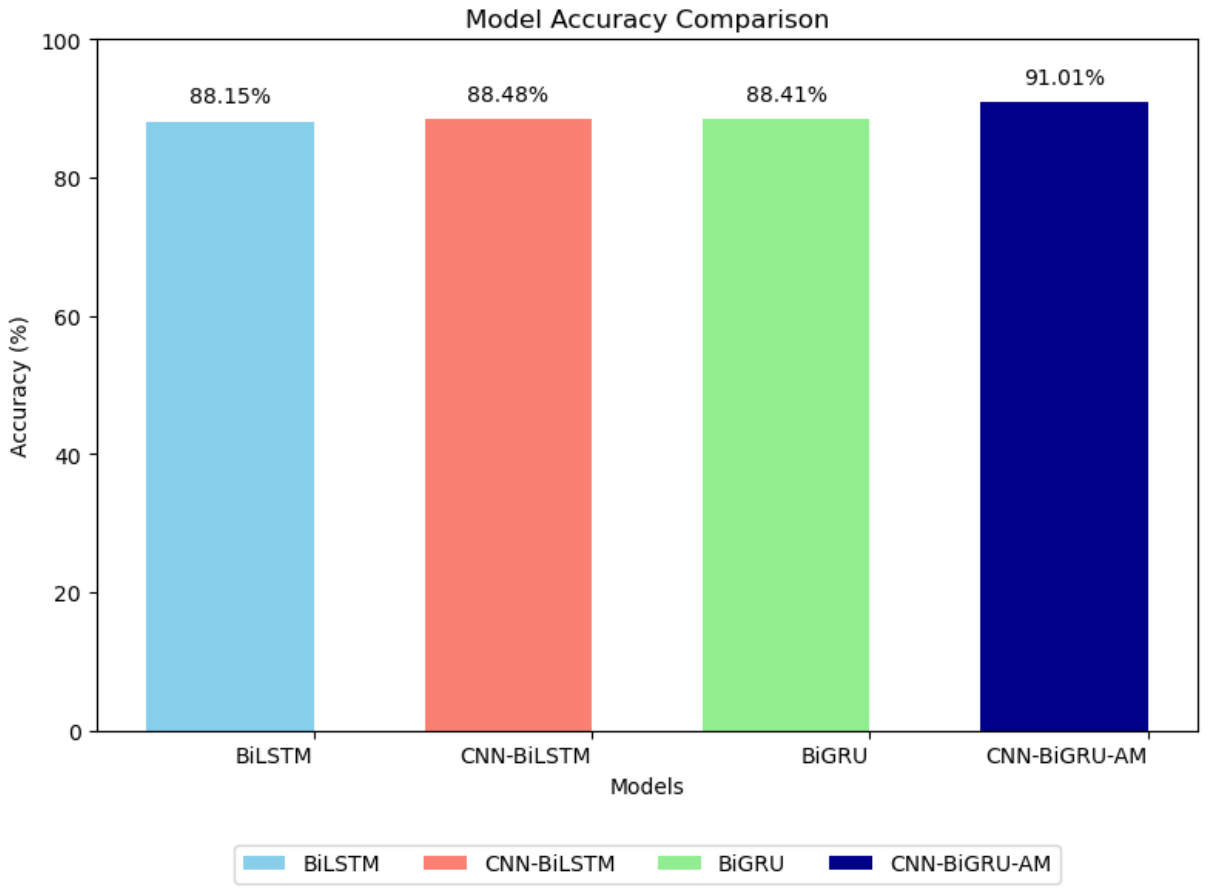

Figure 5.5 Performance Comparison of Proposed Models

5.5 Human Evaluation Results

The proposed model underwent evaluation by users in two methods. The first evaluation centered on the model's performance, evaluated by the accuracy in correctly addressing user queries. The second method, human evaluation, was based on user satisfaction with the proposed model using the human evaluation criteria we defined under section 3.8.1. This involves averaging the queries of ten individuals, selected randomly from Coffee production experts (CPE) and farmers (Farm). We have selected five farmers and five CPE randomly from the individuals who can read and write Afaan Oromo. We can calculate the correctness of the model based on human evaluation using the following Equa. 5.1. And the average correctness model evaluation gave us 84.91%.

$$Correctness = \frac{Total\ \ questions\ answered\ correctly}{Tola\ questions\ requested} * 100 \qquad \textit{Equa. 5.1}$$

The following Table 5.7 shows the correctness of proposed model based on human evaluation.

Table 5.4 Correctness Result of the Model Based on Human Evaluation

Users	**#total questions**	**#correctly answered**	**#wrongly answered**	**Correctness (%)**
Farm-1	18	16	2	88.89
Farm-2	20	17	3	85.00
Farm-3	20	16	4	80.00
Farm-4	16	13	3	81.25
Farm-5	21	19	2	90.48
CPE-1	24	21	3	87.50
CPE-2	27	23	4	85.19
CPE-3	22	18	4	81.82
CPE-4	20	17	3	85.00
CPE-5	25	21	4	84.00
Average				**84.91**

The second human evaluation method is based on users' feedback on the evaluation parameters as we have discussed under section 3.8. we have selected four human evaluation parameters based on(Casas et al., 2020). Each parameter is rated by the rating scale from poor (0) to excellent (5). The following Table 5.8 shows the evaluation of 10 users for the evaluation parameters using the rating scales defined.

Table 5. 5 User Acceptance Model Evaluation

Parameters	**Rating Scales**					**Score (%)**
	Poor (0)	**Average (1)**	**Good (2)**	**V.good (3)**	**Excellent (4)**	
Response Time	-	-	-	3	7	92.5
User Friendly	-	-	1	2	7	90
Usability	-	-	2	4	4	80

Relevance	-	-	2	2	6	85
Average						**86.88**

Since we have created telegram Bot API for the users to interact with our model it is easy for the user to make conversation with model through the telegram app and provide their feedback on the google form created for this purpose. Users interacted with and evaluated the model using the evaluation parameters and provided their feedback by making selections from rating scales corresponding to each parameter. Each user conducted evaluations for all parameters, resulting in a total of 40 evaluations, equivalent to the total number of parameters multiplied by the total number of participating users (10*4). User acceptance can be computed using the subsequent equation. Based this evaluation method we got 86.88% user acceptance result.

$$Acceptance = \frac{\sum_1^n Number\ of\ users\ choose\ rating\ scale * Rating\ valu\ e}{Total\ number\ of\ evaluations} * 100$$ *Equa. 5.2*

5.6 Discussion

In this study, our newly compiled dataset is utilized to perform multiple experiments involving four deep learning models, each paired with two distinct feature extraction techniques. As far as we are aware, three chatbots in Afaan Oromo have been developed in previous research, each catering to different domains. However, due to variations in domains and datasets between those studies and our own, direct performance comparison between our model and theirs is not feasible. We utilized BiLSTM, BiGRU, CNN-BiLSTM, and CNN-BiGRU-AM models and compared the performance of these models.

CNN-BiGRU-AM model with wor2vec outperformed the other models with the accuracy of 91.01%. Word2Vec embeddings can capture semantic relationships between words while fastetxt captures sub-word information. Our dataset is conversational data in which the semantic relationship between words plays a crucial role for the performance of the model. We evaluated the performance of our proposed model in comparison to feature extraction methods like word2vec and fasttext. The model using word2vec achieved better performance on our dataset compared to the model using fasttext. In the CNN-BiGRU-AM model, the

combination of CNN, Word2Vec embeddings, and AM with BiGRU enhances the model's capacity to comprehend and represent textual data. In our study, AM was integrated into all selected models, and their performances were compared with and without AM. AM is designed to enhance the model's ability to emphasize different segments of the input sequence during predictions, consequently aiding the model in effectively grasping long relationships within the text. Our findings on our dataset demonstrate that the inclusion of AM leads to performance improvements on selected models.

In the end, this study has answered the research questions that were posed in section 1.4. As previously explained, the study poses three distinct research questions, all of which are addressed in this specific study.

RQ1: “What are the essential agriculture practices to enhance the production of coffee crop?”. This question deals with agriculture practices that can increase the coffee crop yield. As we have discussed in section 3.2.1 the dataset for this study was collected from Oromia agriculture bureau and from the book intitled “Teeknoolojii oomisha, qulqullina fi walitti hidhamiinsa gabaa bunaa” written by Haile Fufa who is senior coffee specialist. The contents of a dataset from these sources are more about the agricultural practice that the farmers should follow to enhance their coffee crop yields. To enhance the production of coffee crops, several essential agricultural practices can be employed. Firstly, selecting the right variety of coffee seed that are well-suited to the local climate and soil conditions is crucial. Adequate soil management, including appropriate fertilization, helps maintain optimal nutrient levels. Proper spacing of coffee plants allows for adequate sunlight penetration and air circulation, reducing the risk of diseases. Proper watering on coffee plans reduces the risk of fungal diseases and help to keep moisture of soil. Regular pruning not only controls plant height for easier harvesting but also encourages new growth and higher yields. Integrated pest management strategies help minimize the use of harmful chemicals. Timely weed management is another practice that can increase the coffee crop yields. Lastly, adopting shade management techniques through planting trees protects coffee plants from extreme temperatures and supports biodiversity. In integrating these practices, coffee producers can optimize their crop yields.

RQ2: “Which deep learning algorithms can be applicable to build model that provide assistance on the coffee production Practices?”. Applicability of deep learning algorithms depends on the type of dataset. This study’s dataset is text based conversational data which contains pairs of questions and possible answers on the coffee production agricultural practices. Several deep learning algorithms can be harnessed to construct a model that offers guidance on coffee production practices, especially when dealing with a text-based conversational dataset. RNN variants like BiLSTM and BiGRU stand as a suitable choice, as they excel in processing sequential data like conversational exchanges. They hybrid of these models with CNN models also applicable to process sequential data. And AM, often combined with these models, can enhance the model's focus on pertinent parts of the conversation, resulting in more accurate and contextually relevant assistance. For this study, we have selected four deep learning deep learning algorithms: BiLSTM, BiGRU, CNN-BiLSTM and CNN-BiGRU since they show state of the art performance on sequential and NLP tasks.

RQ3: “To what extent can we enhance the overall performance of coffee production virtual assistant model?”. The answer of the question is focus on enhancing the performance of the model using hyperparameter tuning and other methods. To find optimal hyperparameter for the proposed models in this study, we employed grid search algorithm to get optimal combination of hyperparameters for a model. Using this method, we have identified optimal hyperparameter combination for all the proposed models. To improve the performance of the model we have used the hybrid of CNN and RNN variant models. Not only this we have added AM to the models which helps the model to focus on a specific part of a sequence which resulted in overall enhanced performance of the model.

5.7 Model Prototype

In this study, we proposed a flask framework and telegram bot API for developing model prototype. The outperformed model is saved in.h5 format after it has been identified. The saved model then loaded onto the Flask Framework web server or telegram bot so that it could communicate with the user via the Flask Framework user interface or telegram app. Through the offered GU on flask web server or on telegram, the user communicates their request to the model. The user input being preprocessed and the model receive the user's preprocessed input. The model then categorizes the user input under the tag to which it belongs. The model selects

at random the answer to the request after classifying the user intent under the tag to which it belongs. Figure 5.6 and 5.7 below depict sample of a conversation between the model and the user on flask web server. And Figure 5.8 shows sample conversations between users and the model through telegram.

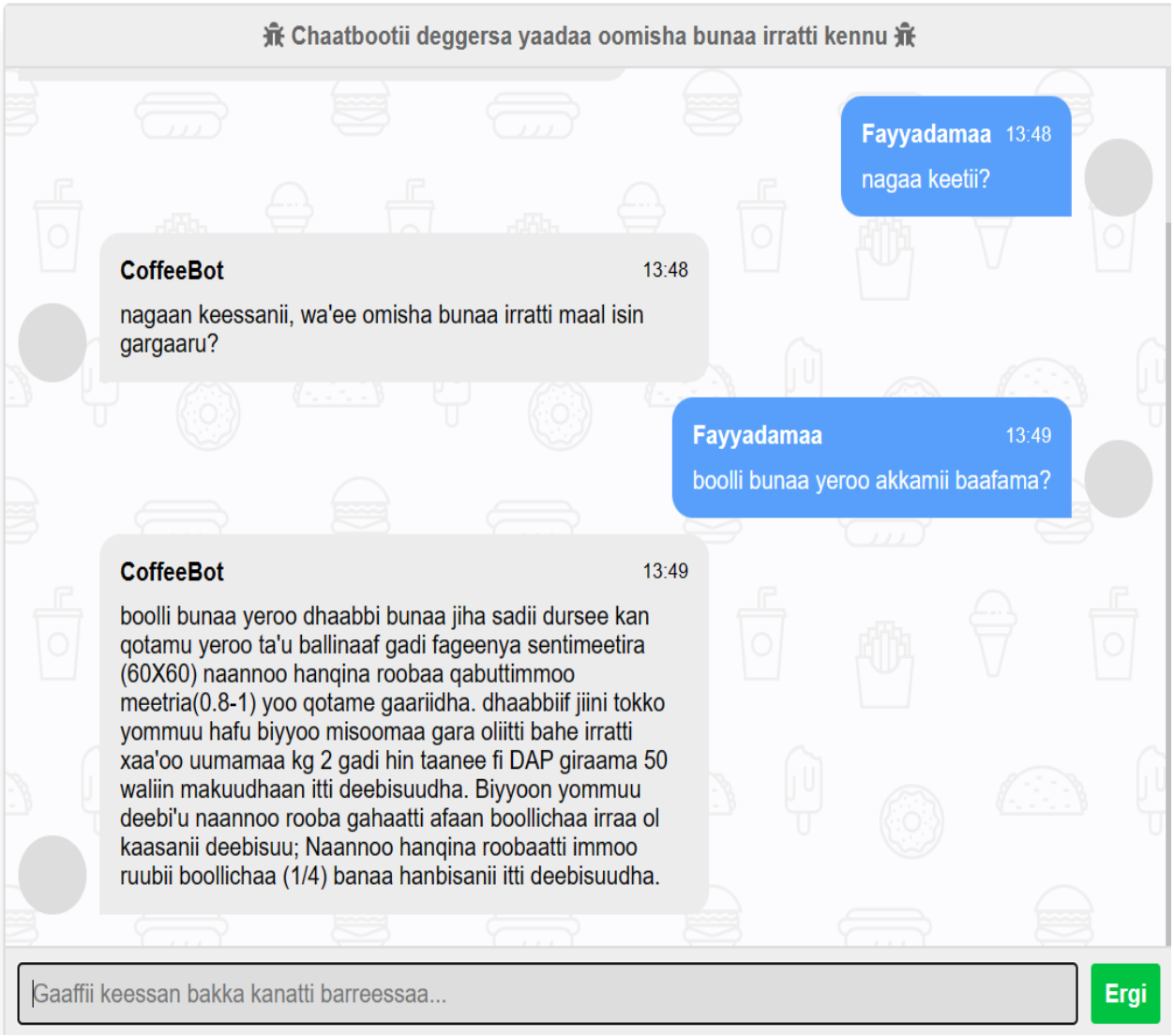

Figure 5.6 Sample Conversation on Flask Framework

Figure 5.7 Sample Conversation on Flask GUI

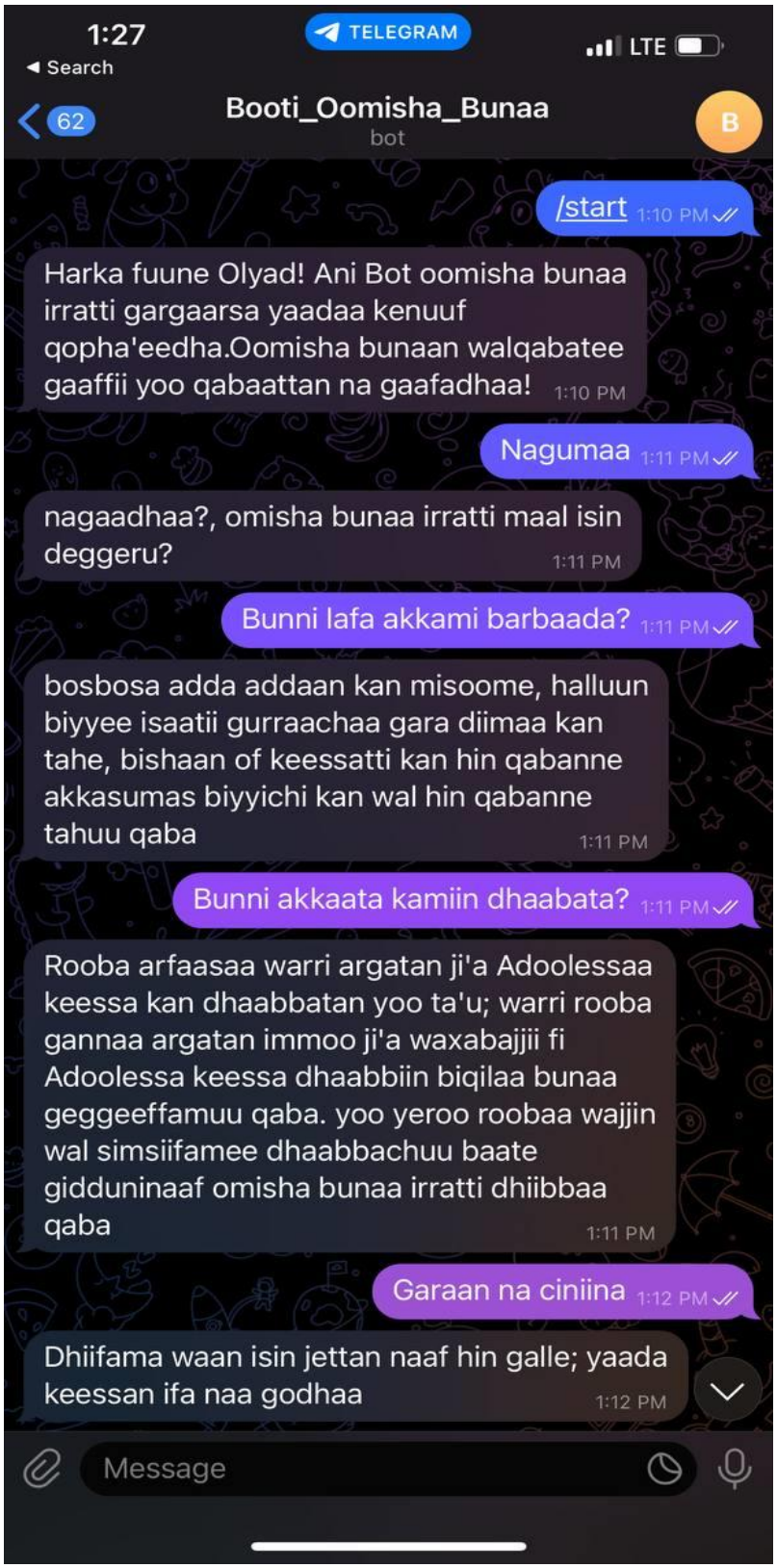

Figure 5.8 Sample Conversation Through Telegram App

CONCLUSION AND RECOMMENDATION

CONCLUSION

This study delved into the realm of enhancing coffee production practices through the development of an Afaan Oromo text-based chatbot. Recognizing the pivotal role of agriculture, particularly the coffee industry, in Ethiopia's economy, the study addressed the challenge of providing timely and accessible information to farmers. The overarching goal was to empower farmers with a conversational AI tool that could bridge the knowledge gap, facilitate informed decisions, and ultimately boost coffee crop yields.

This study proposed text based Afaan Oromo virtual coffee production assistant. To fulfill this study’s goals, a dataset comprising 2363 instances is collected from Oromia agriculture bureau, book and farmers. The dataset contains 123 classes (tags) which holds the patterns and responses. As no prior datasets were available, this dataset is newly prepared. Data preprocessing steps, including tokenization, stop word removal, and normalization, are applied to create a refined dataset suitable for deep learning models. The proposed models were trained on word2vec and fasttext feature extractions. Among the two feature extractions, word2vec outperformed fasttext. The resultant word embeddings from Word2Vec are employed as input for the deep learning models.

we proposed and evaluated BiLSTM, CNN-BiLSTM, BiGRU and CNN-BiGRU-AM model. The outcomes revealed that models with an attention mechanism outperformed those without, reinforcing the importance of context in generating accurate responses. And, despite fasttext's intended solution to the OOV problem, word2vec yielded superior results on our dataset, underscoring the significance of contextual word understanding. On our dataset, CNN-BiGRU-AM model with word2vec outperformed the other models proposed in this study with accuracy of 91.01%. To reduce the bias in the random train test split and ensure reliable evaluations, we employed a 5-fold cross-validation technique. L2 regularization and dropout were used to counteract potential model overfitting. Tunning hyperparameter of the models’ network was performed by using grid search algorithm. Human evaluation further validated the model's success, with an average correctness evaluation score of 84.91%. This score was determined by assessing user satisfaction and the model's accuracy in addressing queries posed by coffee production experts and farmers alike.

Recommendation

To meet the study's goals, we progressed through several phases, including the collection of the dataset. During collecting the data from the farmers, we identified that there is high demand from the farmers to have the system that provide them on the agricultural practice of coffee production information. Additionally, the study's human evaluation results indicated a positive user reception. So, we recommend concerned body apply our model to fulfill the demand.

Feature Work

The primary objective of this study is to enhance coffee crop yields by providing crucial support to farmers in their agricultural practices. The process of delivering a high-quality coffee crop for the global market involves a series of distinct stages. These encompass not only agricultural practices but also extend to encompass post-harvest treatments, which significantly influence the ultimate quality of the coffee crop.

The dataset employed in this study specifically covers essential agricultural practice information that farmers require to cultivate coffee crops of both high quantity and quality. By including a post-harvesting dataset, a more holistic understanding of coffee production can be achieved.

Furthermore, the potential for expansion exists through the addition of datasets in other local languages that pertain to coffee production. By integrating such diverse datasets, the groundwork can be laid for the creation of a multilingual chatbot model. This model could then offer invaluable assistance to farmers in a range of languages, catering to diverse linguistic contexts and enhancing its overall utility.

References

Adamopoulou, E., & Moussiades, L. (2020). An Overview of Chatbot Technology. *IFIP Advances in Information and Communication Technology*, *584 IFIP*, 373–383. https://doi.org/10.1007/978-3-030-49186-4_31

Adane, A., & Bewket, W. (2021). Effects of quality coffee production on smallholders' adaptation to climate change in Yirgacheffe, Southern Ethiopia. *International Journal of Climate Change Strategies and Management*, *13*(4–5), 511–528. https://doi.org/10.1108/IJCCSM-01-2021-0002

Adiwardana, D., Luong, M.-T., So, D. R., Hall, J., Fiedel, N., Thoppilan, R., Yang, Z., Kulshreshtha, A., Nemade, G., Lu, Y., & Le, Q. V. (2020). *Towards a Human-like Open-Domain Chatbot*. http://arxiv.org/abs/2001.09977

Agarwal, R., & Wadhwa, M. (2020). Review of State-of-the-Art Design Techniques for Chatbots. *SN Computer Science 2020 1:5*, *1*(5), 1–12. https://doi.org/10.1007/S42979-020-00255-3

Ahuja, R., Chug, A., Kohli, S., Gupta, S., & Ahuja, P. (2019). The impact of features extraction on the sentiment analysis. *Procedia Computer Science*, *152*, 341–348. https://doi.org/10.1016/j.procs.2019.05.008

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., Hasan, M., Van Essen, B. C., Awwal, A. A. S., & Asari, V. K. (2019). A state-of-the-art survey on deep learning theory and architectures. In *Electronics (Switzerland)* (Vol. 8, Issue 3). MDPI AG. https://doi.org/10.3390/electronics8030292

Alshahrani, S., & Kapetanios, E. (2016). Are deep learning approaches suitable for natural language processing? *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9612*, 343–349. https://doi.org/10.1007/978-3-319-41754-7_33

Anguita, D., Ghelardoni, L., Ghio, A., Oneto, L., & Ridella, S. (n.d.). *The "K" in K-fold Cross Validation*. http://www.i6doc.com/en/livre/?GCOI=28001100967420.

Athiwaratkun, B., Wilson, A. G., & Anandkumar, A. (2018). Probabilistic FastText for Multi-Sense Word Embeddings. *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, *1*, 1–11. https://doi.org/10.18653/v1/p18-1001

Balaji, M., & Yuvaraj, N. (2019). Intelligent chatbot model to enhance the emotion detection in social media using bi-directional recurrent neural network. *Journal of Physics: Conference Series*, *1362*(1). https://doi.org/10.1088/1742-6596/1362/1/012039

Bansal, H., … R. K.-J. of A. R. in C., & 2018, undefined. (2018). A review paper on human computer interaction. *Researchgate.Net*, *8*(8), 2277–128. https://doi.org/10.23956/ijarcsse.v8i4.630

Bhuvaneswari, A., Jones Thomas, J. T., & Kesavan, P. (2019). Embedded Bi-directional GRU and LSTMLearning Models to Predict Disasterson Twitter Data. *Procedia Computer Science*, *165*, 511–516. https://doi.org/10.1016/j.procs.2020.01.020

Borah, B., Pathak, D., Sarmah, P., Som, B., & Nandi, S. (2019a). Survey of Textbased Chatbot in Perspective of Recent Technologies. *Communications in Computer and Information Science*, *1031*, 84–96. https://doi.org/10.1007/978-981-13-8581-0_7

Borah, B., Pathak, D., Sarmah, P., Som, B., & Nandi, S. (2019b). Survey of Textbased Chatbot in Perspective of Recent Technologies. *Communications in Computer and Information Science*, *1031*, 84–96. https://doi.org/10.1007/978-981-13-8581-0_7

Casas, J., Tricot, M. O., Abou Khaled, O., Mugellini, E., & Cudré-Mauroux, P. (2020). Trends & methods in chatbot evaluation. *ICMI 2020 Companion - Companion Publication of the 2020 International*

Chou, T. L., & Hsueh, Y. L. (2019). A task-oriented chatbot based on LSTM and reinforcement learning. *Conference on Multimodal Interaction*, 280–286. https://doi.org/10.1145/3395035.3425319*ACM International Conference Proceeding Series*, 87–91. https://doi.org/10.1145/3342827.3342844

Coffee: The Backbone of Ethiopian Economy. (2015).

DaMatta, F. M. (2004). Ecophysiological constraints on the production of shaded and unshaded coffee: A review. In *Field Crops Research* (Vol. 86, Issues 2–3, pp. 99–114). https://doi.org/10.1016/j.fcr.2003.09.001

Dhyani, M., & Kumar, R. (2019). An intelligent Chatbot using deep learning with Bidirectional RNN and attention model. *Materials Today: Proceedings*, *34*, 817–824. https://doi.org/10.1016/j.matpr.2020.05.450

Di Gennaro, G., Buonanno, A., & Palmieri, F. A. N. (2021). Considerations about learning Word2Vec. *Journal of Supercomputing*, *77*(11), 12320–12335. https://doi.org/10.1007/s11227-021-03743-2

Dilbo, S. B. (n.d.). *Developing Afaan Oromoo Text based Chatbot for Enhancing Maize Productivity*.

Eltahir, A. M., Abdulla, H., Platos, J., & Snasel, V. (2023). *An Overview of Chatbot Structure and Source Algorithms*. 91–98. https://doi.org/10.1109/cscc55931.2022.00026

Ganati, B. A. (n.d.). *Predicting Land Suitability for Wheat and Barley Crops using Machine Learning Techniques*.

Gillioz, A., Casas, J., Mugellini, E., & Khaled, O. A. (2020). Overview of the Transformer-based Models for NLP Tasks. *Proceedings of the 2020 Federated Conference on Computer Science and Information Systems, FedCSIS 2020*, 179–183. https://doi.org/10.15439/2020F20

Handhayani, T., Lewenusa, I., Herwindiati, D. E., & Hendryli, J. (2022). A Comparison of LSTM and BiLSTM for Forecasting the Air Pollution Index and Meteorological Conditions in Jakarta. *2022 5th International Seminar on Research of Information Technology and Intelligent Systems, ISRITI 2022*, 334–339. https://doi.org/10.1109/ISRITI56927.2022.10053078

Harrand, N., Durieux, T., Broman, D., & Baudry, B. (2021). The Behavioral Diversity of Java JSON Libraries. *Proceedings - International Symposium on Software Reliability Engineering, ISSRE*, *2021-October*, 412–422. https://doi.org/10.1109/ISSRE52982.2021.00050

Hu, X., Chu, L., Pei, J., Liu, W., & Bian, J. (2021). Model complexity of deep learning: a survey. *Knowledge and Information Systems*, *63*(10), 2585–2619. https://doi.org/10.1007/s10115-021-01605-0

Imalka, L. A., Gunawardana, K. G. A., Kodithuwakku, K. M. S. K., Arachchi, H. K. E., Harshanath, S. M. B., & Rajapaksha, S. (2022). Farming Through Technology Driven Solutions For Agriculture Industry Ceylon E-Agro mobile application-find technology based solutions for agricultural problems. *IEEE Region 10 Humanitarian Technology Conference, R10-HTC*, *2022-September*, 306–310. https://doi.org/10.1109/R10-HTC54060.2022.9929335

Jain, N., Sahit, P. J., Jain, P., Pachpande, S., Singh, M., Kayal, P., & Choudhari, J. (n.d.). *AgriBot: Agriculture-Specific Question Answer System*. https://data.gov.in[9][10].

Khanna, A., Pandey, B., Vashishta, K., Kalia, K., Pradeepkumar, B., & Das, T. (2015). A Study of Today's A.I. through Chatbots and Rediscovery of Machine Intelligence. *Undefined*, *8*(7), 277–284. https://doi.org/10.14257/IJUNESST.2015.8.7.28

Liashchynskyi, P., & Liashchynskyi, P. (2019). *Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS*. http://arxiv.org/abs/1912.06059

Mary A, R. C., & Sukumar, R. A. (n.d.). *Text Based Smart Answering System in Agriculture using RNN*.

Megersa, H. G. (2022). Habtamu Gudisa Megersa. Coffee (Coffea arabica L.) Field Establishment and Management Practices in Ethiopia. *American Journal of Engineering and Technology Management*, *7*(3), 48–58. https://doi.org/10.11648/j.ajetm.20220703.12

Niu, D., Yu, M., Sun, L., Gao, T., & Wang, K. (2022). Short-term multi-energy load forecasting for integrated energy systems based on CNN-BiGRU optimized by attention mechanism. *Applied Energy*, *313*. https://doi.org/10.1016/j.apenergy.2022.118801

Niu, Z., Zhong, G., & Yu, H. (2021). A review on the attention mechanism of deep learning. *Neurocomputing*, *452*, 48–62. https://doi.org/10.1016/j.neucom.2021.03.091

Nosouhian, S., Nosouhian, F., & Khoshouei, A. K. (2021). *A review of recurrent neural network architecture for sequence learning: Comparison between LSTM and GRU*. https://doi.org/10.20944/preprints202107.0252.v1

Prusty, S., Patnaik, S., & Dash, S. K. (2022). SKCV: Stratified K-fold cross-validation on ML classifiers for predicting cervical cancer. *Frontiers in Nanotechnology*, *4*. https://doi.org/10.3389/fnano.2022.972421

Role of Agriculture in the Economy - QS Study. (n.d.). Retrieved December 3, 2022, from https://qsstudy.com/role-of-agriculture-in-the-economy/

Sairamya, N. J., Susmitha, L., Thomas George, S., & Subathra, M. S. P. (2019). Hybrid approach for classification of electroencephalographic signals using time-frequency images with wavelets and texture features. In *Intelligent Data Analysis for Biomedical Applications: Challenges and Solutions* (pp. 253–273). Elsevier. https://doi.org/10.1016/B978-0-12-815553-0.00013-6

Santos, R. M., Santos, I. M., Rodrigues, M. C., & de Mendonça Neto, M. G. (2020). Long term-short memory neural networks and word2vec for self-admitted technical debt detection. *ICEIS 2020 - Proceedings of the 22nd International Conference on Enterprise Information Systems*, *2*, 157–165. https://doi.org/10.5220/0009796001570165

Shanghai hai yang da xue, & Institute of Electrical and Electronics Engineers. (n.d.). *Proceedings of 2020 IEEE International Conference on Artificial Intelligence and Computer Applications : ICAICA 2020 : Dalian, China, June 27-29, 2020*.

Shen, T., Zhou, T., Long, G., Jiang, J., Wang, S., & Zhang, C. (2018). *Reinforced Self-Attention Network: a Hybrid of Hard and Soft Attention for Sequence Modeling*. http://arxiv.org/abs/1801.10296

Sichuan Institute of Electronics, & Institute of Electrical and Electronics Engineers. (n.d.). *2018 IEEE 4th International Conference on Computer and Communications (ICCC) : December 7-10, 2018, Chengdu, China*.

Singh, S., & Thakur, H. K. (2020). Survey of Various AI Chatbots Based on Technology Used. *ICRITO 2020 - IEEE 8th International Conference on Reliability, Infocom*

Technologies and Optimization (Trends and Future Directions), 1074–1079. https://doi.org/10.1109/ICRITO48877.2020.9197943

Sri Shakthi Institute of Engineering and Technology, Institute of Electrical and Electronics Engineers. Madras Section, All-India Council for Technical Education, & Institute of Electrical and Electronics Engineers. (n.d.). *2020 International Conference on Computer Communication and Informatics : January 22-24, 2020, Coimbatore, India*.

Suebsombut, P., Sureephong, P., Sekhari, A., Chernbumroong, S., & Bouras, A. (2022). Chatbot Application to Support Smart Agriculture in Thailand. *7th International Conference on Digital Arts, Media and Technology, DAMT 2022 and 5th ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering, NCON 2022*, 364–367. https://doi.org/10.1109/ECTIDAMTNCON53731.2022.9720318

Suhaili, S. M., Salim, N., & Jambli, M. N. (n.d.). A Comparative Analysis of Generative Neural Attention-based Service Chatbot. In *IJACSA) International Journal of Advanced Computer Science and Applications* (Vol. 13, Issue 8). www.ijacsa.thesai.org

Suta, P., Lan, X., Wu, B., Mongkolnam, P., & Chan, J. H. (2020). An overview of machine learning in chatbots. *International Journal of Mechanical Engineering and Robotics Research*, *9*(4), 502–510. https://doi.org/10.18178/ijmerr.9.4.502-510

Tam, S., Said, R. Ben, & Tanriöver, Ö. (2021). A ConvBiLSTM Deep Learning Model-Based Approach for Twitter Sentiment Classification. *IEEE Access*, *9*, 41283–41293. https://doi.org/10.1109/ACCESS.2021.3064830

Tesema, W., & Tamirat, D. (n.d.). *Investigating Afan Oromo Language Structure and Developing Effective File Editing Tool as Plug-in into Ms Word to Support Text Entry and Input Methods*. www.pubicon.in

Thakur, A., & Monga, S. (2022). A Survey on Time Series Forecasting Approaches and Applications. *International Journal for Research in Applied Science and Engineering Technology*, *10*(4), 2729–2736. https://doi.org/10.22214/ijraset.2022.41879

Tishk International University, Erbil Polytechnic University, Institute of Electrical and Electronics Engineers, & Institute of Electrical and Electronics Engineers. Iraq Section. (n.d.). *Proceedings of the 5th International Engineering Conference (IEC2019) : June 23-25, 2019, Erbil, Kurdistan Region-Iraq*.

TURING, A. M. (1950). I.—COMPUTING MACHINERY AND INTELLIGENCE. *Mind*, *LIX*(236), 433–460. https://doi.org/10.1093/MIND/LIX.236.433

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017a). *Attention Is All You Need*. http://arxiv.org/abs/1706.03762

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017b). *Attention Is All You Need*. http://arxiv.org/abs/1706.03762

Wayessa, N., & Abas, S. (2020). Multi-Class Sentiment Analysis from Afaan Oromo Text Based On Supervised Machine Learning Approaches. In *International Journal of Research Studies in Science, Engineering and Technology* (Vol. 7, Issue 7).

Wolde, Z., Tefera, A., Yared, S., Gezahagn, T., & Tadesse, T. (n.d.). A Review on Coffee Farming, Production Potential and Constraints in Gedeo Zone, Southern Ethiopia. *Journal of Natural Sciences Research*. www.iiste.org

Xiao, J., & Zhou, Z. (2020). Research Progress of RNN Language Model. *Proceedings of 2020 IEEE International Conference on Artificial Intelligence and Computer Applications, ICAICA 2020*, 1285–1288. https://doi.org/10.1109/ICAICA50127.2020.9182390

Yacouby Amazon Alexa, R., & Axman Amazon Alexa, D. (n.d.). *Probabilistic Extension of Precision, Recall, and F1 Score for More Thorough Evaluation of Classification Models*.

Yang, S., Yu, X., & Zhou, Y. (2020a). LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example. *Proceedings - 2020 International Workshop on Electronic Communication and Artificial Intelligence, IWECAI 2020*, 98–101. https://doi.org/10.1109/IWECAI50956.2020.00027

Yang, S., Yu, X., & Zhou, Y. (2020b). LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example. *Proceedings - 2020*

International Workshop on Electronic Communication and Artificial Intelligence, IWECAI 2020, 98–101. https://doi.org/10.1109/IWECAI50956.2020.00027

Ying, X. (2019). An Overview of Overfitting and its Solutions. *Journal of Physics: Conference Series*, *1168*(2). https://doi.org/10.1088/1742-6596/1168/2/022022

Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: Lstm cells and network architectures. In *Neural Computation* (Vol. 31, Issue 7, pp. 1235–1270). MIT Press Journals. https://doi.org/10.1162/neco_a_01199

Yue, W., & Li, L. (2020, December 14). Sentiment analysis using word2vec-cnn-bilstm classification. *2020 7th International Conference on Social Network Analysis, Management and Security, SNAMS 2020*. https://doi.org/10.1109/SNAMS52053.2020.9336549

Zahran Aufa, B., Suyanto, S., & Arifianto, A. (2020). Hyperparameter Setting of LSTM-based Language Model using Grey Wolf Optimizer. In *2020 International Conference on Data Science and Its Applications (ICoDSA)*.

ZEMČÍK, Mgr. T. (2019). A Brief History of Chatbots. *DEStech Transactions on Computer Science and Engineering*, *aicae*. https://doi.org/10.12783/dtcse/aicae2019/31439

Zhou, G., Guo, Z., Sun, S., & Jin, Q. (2023a). A CNN-BiGRU-AM neural network for AI applications in shale oil production prediction. *Applied Energy*, *344*. https://doi.org/10.1016/j.apenergy.2023.121249

Zhou, G., Guo, Z., Sun, S., & Jin, Q. (2023b). A CNN-BiGRU-AM neural network for AI applications in shale oil production prediction. *Applied Energy*, *344*. https://doi.org/10.1016/j.apenergy.2023.121249

Appendix A: Afaan Oromo sample stopwords list

aanee	Ala	isaaf	utuu	Gama	turuuf
gidduu	illee	isaanif	osoo	silaa	asii
itti	kan	isaan	booddee	kanaafi	kam
narraa	siin	keef	iseen	kana	kamiin
akka	alatti	tanaaf	isheen	alatti	mini
gubbaa	immoo	ani	keessatti	yogguu	miti
ittuu	kana	isaaf	waan	hogguu	duuba
natti	silaa	keenna	waanta	ittuu	siif
jala	amma	tanaafuu	dura	akka	wanti
nu	inni	ati	ishii	kiyya	ofirraa
akkasumas	kanaafi	isaanirraa	kiyya	warra	ofirratti
henna	kanaaf	keessa	warra	wal	sitti
erga	sitti	ta'ullee	duuba	hoo	takkaaf
jara	ammo	bira	ishiif	jirutti	moo
nurraa	irra	isatti	koo	kanneen	ture
akkuma	koo	keessan	duraaf	jira	kootu
hogguu	koof	teenya	yeroo	kami	akkasumatti
sana	kanaafuu	booda	ega	baayyee	maaltu
saniif	sun	tun	ishiirraa	akkamii	gadi
nuti	an	kun	kun	akkan	atuu
ta'us	anaaf	keenya	yommuu	atoo	situ
fakkaata	isin	yoom	irratti	gara	natu
eegasii	na	kun	aansuun	baayee	ta'eef
isii	yookaan	malees	fufuun	nan	ta'uus
malee	gama	kanaafuu	kanaaf	maalif	jelaa
yoo	isin	keessa	keenya	maaf	gubbaan
fi	naaf	dabalataan	yookaan	irraan	jelaaf
jelaan	tureef				

Appendix B: Sample of dataset


```
{
 "tag": "Damping_off",
 "patterns": ["biqilaan bunaa akkuma biqileen, hundeen isaa akka tortoru kan godhu maali?",
        "biqilaatu madaba irratti akkuma biqileen na jelaa coologe",
        "hundeen biqilaa bunaa maaltu gurracheessuu danda'a?",
        "cologuu biqilaa bunaa ittisuun hin danda'amaa?",
        "biqilaan buna akka tortoruuf sababi maalidha?",
        "biqilaan reefuu biqilu maaf hundeen isaa tortoraa",
        "hundeen biqilaa akkuma biqileen tortoruun sababnisaa maali laata?",
        "tortorsaa hundee maxxansee bunaa",
        "tortoruu biqilaa bunaa",
        "buna maxxanseen akkuma lafa keessa ol ba'een hundeen isaa coolage",
        "tortorsaa hundee biqilaa bunaa reefu biqiluu akkamiinan hambisuu danda'a?",
        "colloguu hundeee biqilaa bunaa madaba irraa maaltu fidaa?",
        "sanyiin ani facaase akkuma biqileen coolagee gubate maal fallikoo",
        "biqilaa buna ega lafa keessa ba'ee jirmisaa gurracha'e",
        "sababni tortorsaa hundee biqiltuu bunaa maaluma laata?",
        "waanumti hundee biqilaa bunaa gossitu kuni malii?",
        "wanti hundee biqilaa reefuu laf akeessaa bahuu nyaatu kun maal ta'aa?",
        "waanti buna rrefuu biqilaa jiru ciru maaloodhaa?"],
 "responses": ["sanyiin faca’e biyyoo keessaa ega bahee/biqilee booda biyyoo keessaa akkuma baheen bakka jirmi biqiltuu biyyoo tuqutti tortorsudhaan gurraachessanii boodammoo cologsanii ajjeessan keessaa inni tokko dhukkuba tortosaa fangasii biyyoo keessaati. kanafimmoo gara caalu sababa kan ta'u danda'u madaba biqiltuutti bishaanni baayyachuudha.",
        "maloota dhukkubni kun ittiin ittifamuu danda'aan:\\n bayyina afaa fi bishaanii irraa kan ka'e jidhinni akka madaba irratti hin baayyanne gochuu fi walirraa fageenya biqiltuu gidduu jiru sirreessuu\\n buufata biqiltuu tokko yeroo heedduuf deddeebi'anii fayyadamuu dhiisuu fi sanyii bunaa haaraa ta'e fayyadamuu\\n Gadi fageenya gorfamuutii olitti gadi awaalanii facaasuu dhiisu fi biqiltuuwwan dhukkubaan miidhaman buqqisanii awwaaluu\\n Biyyoo madaba biqiltuu muka qara qabuun kotokotudhaan qilleensa akka argatu gochuu."],
 "context": ["tortorsaa"]
},
{
 "tag": "Brown_Eye_Spot",
 "patterns": ["baala biqilaa bunaa maaltu irraa harcaasa?",
        "biqilaa bunaa baalli isaa waan akka guggubataa qabu",
        "xuqaa baayyeetu baala irratti baayyatee mullata sababni isaa maali?",
```

"waanti akka roga ija xuqaa baalaa bunaa irratti maaf mullataa?",

"waanti akka xifxiffee baala bunaa irratti bahu dhiibbaan isaa maali?",

"waanti akka geengoo gugguraachaa baala bunaa irratti bahu maalirraa ka'aa?",

"baalli bunaa sababa maalitiif irraa harca'aa?",

"baalli bunaa maalif waan akka roga rogaa godhataa?",

"sababni baalli bunaa akka harca'u godhu maal inni?",

"waanti baala bubaa waan guguraacha xixiqqaa irraati baasu dhukkubamoo?",

"akka baalli bunaa gugurracha'ee irraa hin harcaaneef maal gochuun qaba?",

"maal inni kan baalli bunaa irraa harca'uuf?",

"maaloo waanuma baala bunaa irratti waan gugurattii xixiqqaa baasuun irra harcaasu kana natti himimee",

"baalli bunaa maalif akka irra harca'uuf waan beektu yoo jiraate natti himi",

"gugurattii baala bunaa irratti baatu",

"waanti baala bunaa gugurachessee irraa harcaasuu dhukkuba akkamiiti?",

"baalli bunaa akka irraa hin harcaaneef akkamiinan ittisa?",

"dhukkuba baala bunaa ure uree irraa harcaasuu akkamiin to'achuun danda'aa?",

"baalli bunaa hanqinaa maaliitiif darbee darbee tuqaa guguraachaa godhachuun harca'aa?",

"yoon maal godheen baalli bunaa waan akka roga rogaa godhachuu dhiisa?"],

"responses": ["wantoonni akka roga ijaa xuqaa baayyee baala biqilaa bunaa iratti mullatan, yeroo baayyee buna yeroo dheeraaf irraanfatame aramaan nyaatame irratti mul'ata kana malees, biqiltuuwwan gaaddisan dhaban irrattillee hin mullata. dhukkubni kun maloota akka gaaddisa gahaa ykn daasii biqilaaf hojjechuu, bishaan gahaa ta'e biqiltuu obaasuu fi xaa'oo ykn dikee biqilaaf kennuutiin ittifamuu danda'a.",

"gosa dhukkuba baala biqilaa bunaa harcaasu ittisuuf, gaaddisa gahaa ykn daasii ijaaruu, bishaan gahaa ta'e obaasuu fi xaa'oo itti naquudhaan ittifamu danda'a",

"baalli biqilaa bunaa waan akka roga rogaa akka godhatu kan godhu dhukkuba fangasii kanuunsa dhabuu biqilaa bunaa irraan kan dhufuudha "],

"context": [""]

},

Appendix C: Sample user evaluation form for the proposed model

Formii Madaallii Chatbot Oomisha bunaa

Duraan dursa systema kana madaaluudhaaf ayyamaamaa ta'uu keessaniif galanni keenya guddaadha. Erga System kana fayyadamtanii booda gaaffilee armaan gadii sirriitti nuuf guutun nu gargaaraa.

abdimilkesa98@gmail.com Switch account

Not shared

* Indicates required question

Maqaa guutuu *

Your answer

Ga'ee hojii keessanii *

Your answer

Gaaffii meeqa sistema keenya gaafattan? *

Your answer

Gaffii meeqatu sirriitti isiniif deebi'e? *

Your answer

Turtii deebiin keessaan itti isiniif kenname akkamitti madaaltu? *

- ◯ Gadaanaadha
- ◯ Giddu-galeessa
- ◯ Gaaridha
- ◯ Baayyee gaariidha
- ◯ Bayyee baayyee gaariidha

v

sistemichi fayyadamummaaf salphaa ta'usaa akkamitti ilaaltu?

- ◯ Gadaanaadha
- ◯ Giddu-galeessa
- ◯ Gaariidha
- ◯ Baayyee gaariidha
- ◯ Baayyee baayyee gaariidha

Walitti dhufeenyi gaaffii keessaniif deebii isiniif kennama gidduu jiru akkamiin madaaltu? *

- ◯ Gadaanaadha
- ◯ Giddu-galeessa
- ◯ Gaariidha
- ◯ Baayyee gaariidha
- ◯ Baayyee baayyee gaariidha

Yaada dabalataa yoo jiraate nuuf heeraa. Yeroo keessan lattanii nu gargaaruuf hayyamamaa ta'uu keessaniif ulfaadha!

Your answer

Submit Clear form

Appendix D: Sample code

Appendix D.1: Sample code to integrate to flask framework

```
def predict_class(patterns, model, threshold=0.5):
    p = word_emb(patterns, words, show_details=False)
    predic = tokenizer.texts_to_sequences(p)
    predic = np.array(predic).reshape(-1)
    predic = pad_sequences([predic], input_shape)
    output = model.predict(predic)
    threshold=0.75
    if np.max(output) >= threshold:
        output = output.argmax()
        response_tag = le.inverse_transform([output])[0]
        res = random.choice(responses[response_tag])
    else:
        res = "Dhiifama waan isin jettan naaf hin galle; yaada keessan ifa naa godhaa "


    return res
def chatbot_response(msg):
    ints = predict_class(msg, model)
    return ints
@app.route("/")
def home():
    return render_template("index.html")
@app.route("/get")
def get_bot_response():
    userText = request.args.get('msg')
    return chatbot_response(userText)
```

Appendix D.2: sample code to integrate to telegram bot

```
def predict_class(patterns, model, threshold=0.5):
    p = word_emb(patterns, words, show_details=False)
    predic = tokenizer.texts_to_sequences(p)
    predic = np.array(predic).reshape(-1)
    predic = pad_sequences([predic], input_shape)
    output = model.predict(predic)
    threshold = 0.75
    if np.max(output) >= threshold:
        output = output.argmax()
        response_tag = le.inverse_transform([output])[0]
        res = random.choice(responses[response_tag])
    else:
        res = "Dhiifama waan isin jettan naaf hin galle; yaada keessan ifa naa godhaa "
    return res
def chatbot_response(msg):
    ints = predict_class(msg, model)
    return ints
def start(update: Update, context: CallbackContext):
    user = update.effective_user
    update.message.reply_text(f"Harka fuune {user.first_name}! Ani Bot oomisha bunaa irratti gargaarsa yaadaa kenuuf qopha'eedha.Oomisha bunaan walqabatee gaaffii yoo qabaattan na gaafadhaa!")
def echo(update: Update, context: CallbackContext):
    message = update.message.text
    response = chatbot_response(message)
    update.message.reply_text(response)
def main():
BotFather
    updater = Updater(token="My_tegerambot_token")
```

Appendix E: Telegram Bot User name