A Legal Consultant Chatbot for Supreme Court of Oromia Using Deep Learning Techniques

Fiseha Damtew Mokonnen

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's in Computer Science and Engineering

Office of Graduate Studies

Adama Science and Technology University

June, 2023

Adama, Ethiopia

A Legal Consultant Chatbot for Supreme Court of Oromia Using Deep Learning Techniques

Fiseha Damtew Mokonnen

Advisor: Teklu Urgessa (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's in Computer Science and Engineering

Office of Graduate Studies
Adama Science and Technology University

June, 2023
Adama, Ethiopia

Approval Sheet

I/we, the advisors of the thesis entitled "**A Legal Consultant Chatbot for Supreme Court of Oromia Using Deep Learning Techniques**" and developed by **Fiseha Damtew Mokonnen** , hereby certify that the recommendation and suggestions made by the board of examiners are appropriately incorporated into the final version of the thesis.

Teklu Urgessa (Ph.D)

Major Advisor	Signature	Date
Co-advisor	Signature	Date

We, the undersigned, members of the Board of Examiners of the thesis **Fiseha Damtew Mokonnen** have read and evaluated the thesis entitled "**A Legal Consultant Chatbot for Supreme Court of Oromia Using Deep Learning Techniques**" 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 Engineering

Chairperson	Signature	Date
Internal Examiner	Signature	Date
External Examiner	Signature	Date

Finally, approval and acceptance of the thesis are contingent upon 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

Declaration

I hereby declare that this Master Thesis entitled "**A Legal Consultant Chatbot for Supreme Court of Oromia Using Deep Learning Techniques**" 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.

Fiseha Damtew Mokonnen	_____	_____
Name of the Student	Signature	Date

Recommendation

I/we, the advisor(s) of this thesis, hereby certify that I/we have read the revised version of the thesis entitled "**A Legal Consultant Chatbot for Supreme Court of Oromia Using Deep Learning Techniques**" prepared under my/our guidance by **Fiseha Damtew Mokonnen** submitted in partial fulfillment of the requirements for the degree of Mater's of Science in Computer Science Engineering. Therefore, I/we recommend the submission of the revised version of the thesis to the department following the applicable procedures.

Teklu Urgessa (Ph.D)	_____	_____
Major Advisor	Signature	Date

Acknowledgment

First and foremost, I would like to express my gratitude to God for providing me with the strength and support to overcome the challenges I encountered during my studies and for helping me attain this significant achievement.

My next and most grateful step is to express my heartfelt appreciation to my advisor, Dr. Teklu Urgessa (Ph.D.), for the continuous support and guidance provided throughout this stage of my work. Without his involvement and intervention during challenging moments, completing this thesis according to the initial plan would have been extremely arduous. Dr. Teklu Urgessa has offered valuable feedback and suggestions, enabling me to approach research problems tactfully and systematically. I am truly thankful to him for his unwavering assistance.

Another person I am very grateful to is Dr. Mesfin. Dr. Mesfin's opinion has a very significant contribution to the success of my thesis.

Also, I would like to express my deep gratitude to Mr. Girma Asefa for dedicating his time and effort from the beginning till the end, and for going above and beyond to assist me in every way possible. He has exhibited the qualities of an exceptional teacher, providing me with unwavering support and willingly offering his professional guidance. I am truly grateful for his invaluable help.

Finally, I would like to express my heartfelt gratitude to my entire family, mentioning my mother, Aregash Mokonnen, and my wife, Enway Demsie. Their unwavering love, encouragement and support have been instrumental in my personal and academic journey. Thank you so much for always being there and helping me in every way and going above and beyond. Their faith in me has been a source of inspiration, and I am truly lucky to have them by my side.

Table of Contents

List of Table

List of Figures

Acronyms And Abbreviation

AI- -Artificial Intelligence

ANN- -Artificial Neural Network

API - -Application Program Interface

CUI- -Chatbot User Interface

CNN- -Convolutional Neural Network

CSOs--Customer Service Officers

DL- -Deep Learning

DNN - -Deep Neural Networks

ELMo -- Embedding’s from language Models Representation

FAQ- - Frequently Asked Questions

FE- -Feature Extraction

GPT--Generative Pre-training Transformer

IPA- -International Phonetic Alphabet

JSON- -JavaScript Notation

KB- - Kilo Byte

LSTM- -Long Short-Term Memory

MB- - Mega Byte

ML- -Machine Learning

NLP- -Natural Processing Language

OSC- -Oromia Supreme Cort

RNN- -Recurrent Neural Network

SDK- -Software Development

SNS-- social network service

UML- -Unified Modeling Language

Abstract

Legal awareness is crucial in promoting the rule of law and fostering legal cultural awareness. The legal domain is characterized by complex and intricate language structures, making it challenging for non-experts to navigate. The introduction of generative chatbots for court services has the potential to revolutionize the way legal information is accessed and understood by the general public. Many court clients miss their right due to lack of awareness about legal. In this study, we present the development of a generative legal consultant chatbot for the Oromia Supreme Court, aimed at providing accurate, timely, and accessible legal information to users. We collected 2MB of data from primary and secondary (Facebook, Telegram, and websites) sources, to train our models. The collected dataset was prepared in the form of plaintext and preprocessed using various NLP techniques. An experimental approach was performed in this study to determine best model. Deep learning models based on GPT (Generative Pre-trained Transformer) and LSTM (Long Short-Term Memory) with ELMO (Embedding's from Language Models Representation), have been experimented for legal consultant services using 10-fold stratified cross-validation on fine-tuned hyperparameters. Different classification metrics are used for evaluating models. The GPT model outperformed the LSTM with ELMO model, achieving a remarkable accuracy of 90.8%. Finally, the best-performed model was evaluated by legal experts and achieved 88.5% accuracy. Eventually, our work revealed that using language model with fine-tuned hyperparameters significantly enhances the quality of legal consultant chatbot services. The developed legal consultant chatbot demonstrates its potential to support the Oromia Supreme Court in providing efficient and accessible legal services. Future work may involve expanding the dataset, fine-tuning the models, and integrating additional features to further enhance the chatbot's performance and accuracy.

***Keywords**: Generative chatbot, Legal Domain, Generative Pre-trained Transformer, Long Short-Term Memory, and Embedding's from Language Models Representation.*

CHAPTER ONE

1. INTRODUCTION

1.1. Background of the Study

The discipline and profession of law refer to the norms of community culture, practice, and conduct recognized as binding by society. The enforcement of the law is carried out by the supervisory authority, and one of these authorities is the court. A court is any person or institution, usually a government agency, which has the power to decide legal disputes between parties and to administer justice in civil, criminal, and administrative matters in accordance with the rule of law. Before modern law was enacted, elders, kings, and other systems played and still play an important role in resolving disputes between claimants and defendants. The Oromia Supreme Court (OSC) is one of the state courts in Ethiopia and is responsible for providing accessible and addressable justice services in the region. The judicial decision and legal consultancy services are among the main services offered by the OSC. This research focuses only on legal consultancy services, as provided by the OSC.

Legal consultancy services are the services provided by legal experts or lawyers about legal in order to give awareness for clients. Many people are not even aware of their fundamentals rights as provided by the Constitution. In legal procedures, it is important for the defendant or claimants to know about his rights one by one and to understand in advance what will happen in court, as well as for the defendant to be aware of the law in all decisions. According to (Peter C. Brown, Henry L. Roediger III, 2012), (Blumstein, A., & Rosenfeld, 2008) description, knowing the law can minimize crime in a country. In OSC, to give legal consultancy each court has responsible body that called advocate. It is the role of these advocates (Abukaato Ittisa) to advise and represent the defendant. However there are a limited number of government advocate in. In addition to the advocate of OSC, there are private legal consultancy firms that were established under proclamation No. 86/2004 under provisions of license to advocate and code of conduct of advocate in Oromia region to provide advocate services (Magalata Oro).

In many instances, clients are required to physically go to court or private legal firms to obtain information such as the date of their appointment, the status of their record, the type of decision

made, and obtain copies of relevant documents. These processes often result in unnecessary expenses for the clients. The Supreme Court has identified a range of problems associated with these processes, including the high level of labor, time, and resources required by the courts during proceedings. Additionally, there is a challenge in identifying the whereabouts of a participant in the legal process and effectively handling court documents.

Due to this fact, it is necessary to efficiently use the technological support that provides legal consulting services. The way people access information today has changed dramatically through technology across different industries. Legal understanding has a crucial benefit for court clients, to give awareness about court procedures, clients' rights & obligations for the defendants and Plaintiff. Artificial intelligence and modern computing solutions have made this information more efficient, simple, and user-friendly. Advances in AI, machine learning, and deep learning have allowed machines to impersonate humans (Nuruzzaman, M., & Hussain, 2018)**.**

Artificial intelligence (AI) is revolutionizing the way humans interact with machines. AI builds technology that enables users to engage with machines in a manner similar to human-to-human interactions. One of the most prominent examples of such technology is the chatbot, which utilizes natural language processing (NLP) to simulate human-like conversations. By combining NLP with machine learning, deep learning, and statistical models, computers can communicate with humans in natural languages such as Amharic, Spanish, Chinese, Afan Oromo, among others. AI startups in legal consulting have significant potential to improve and assist the justice system. Natural language processing is a method of interacting with intelligent systems using natural language (TutorialsPoint, 2020)**.** The current state of AI technology allows for the interaction between humans and machines to occur in the form of natural human-to-human conversation. The role of natural language processing in this process is vital (Joseph et al., 2016).

Chatbots have become an integral part of our daily lives, and they play a crucial role in various sectors. In the legal sector, chatbots are helping litigants save money, energy, and time during legal disputes. Through text messages and apps, chatbots offer counseling services to users to help them resolve disputes peacefully, avoiding costly legal battles that may harm their life and property. Essentially, chatbots act as advanced search engines, allowing users to search computer databases in a user-friendly way. One of the significant advantages of chatbots is automation,

particularly in customer service. According to (Chung, K., & Park, 2019), chatbots are intelligent interactive platforms that interact with users through a chat interface. Because chatbots can connect to the main social network service (SNS) messengers, they can easily provide access to various services to general users. Chatbots are becoming increasingly important in law, healthcare, insurance, banking, financial services, and other sectors.

However, to our knowledge, there is currently no chatbot model based on Afan Oromo text specifically designed for use in courts. Therefore, our main objective is to develop a generative chatbot based on Afan Oromo text that can provide more effective legal counseling services to court users. Machine learning techniques are integrated with word embeddings for much better, faster results and generate accurate response. Nowadays Researchers are using different Machine and deep learning algorithms include empirical rule-based methods, support vector machines (SVMs), artificial neural networks (ANNs), Gaussian mixture models (GMMs), k-nearest neighbors (k-Nns), Convolutional Neural Network (CNN) (B. M. Rocha et al., 2019), and logistic regression models and more advanced machine learning techniques. In this paper the potential of GPT and LSTM are exploited for the exact purpose of developing generative legal consultant service chatbot.

1.2. Motivation of the Study

Legal awareness is crucial in promoting the rule of law and fostering legal cultural awareness. It is important to provide education and awareness about legal proceedings to ensure that individuals have a basic understanding of their legal rights and responsibilities. However, legal proceedings can be complex and require the expertise of legal professionals. Unfortunately, many court clients may not have the necessary knowledge or experience to effectively advocate for their rights, leading to a lack of dispute resolution. Due to this, they didn't dispute for their rights. This causes losses; here is a real scenario:

> *In the area where I was born and raised, there was an elderly farmer named Gammachu Ol'aanaa who had a court dispute over land. The court where the case was held was far away from where they lived…. This father told me that the main reasons for his decision were: They did not make a proper argument for their rights due to lack of information in the debate process, and they did not freely express their thoughts due to their fear of the*

court…. Also, on the day of the appointment and when they gave the appropriate response, they were found to be a little late because there was no transportation inconvenience, and the above-mentioned problems combined, the court found them guilty… They lost their rights.

Therefore, minimizing such problems and decreasing the number of people losing their rights from lack of awareness and free legal counseling services is an ideal solution. It is highly motivating to make a solution for court clients in providing better advice.

1.3. Statement of the Problem

When the government brings the accused to court, they are given legal and constitutional protection just because they are human beings. Because the accused who cannot stand an advocate, does not have the necessary legal skills and awareness to argue with the government without an advocate. In addition to this, due to the fear of the court, the litigants find it difficult to freely submit their requests to the court. Another and crucial problem are that any person goes to the court repeatedly in search of court consulting, which again leads to unnecessary expenses. When private lawyers ask clients to give advice, they charge more money than they can afford. When clients consult attorneys for legal advice, the attorney may mislead them by engaging with the opposing party.

The community often goes to the court in search of court consulting services and this again leads to unnecessary waste of time, morale and effort. When people go to the lawyer's place to get information about what they need, they may be exposed to thieves. Other legal consultancy service providers are not able to provide appropriate consultancy because of the number of people seeking advice.

1.4. Research Question

This study addresses the above problem by answering the following research questions:

RQ1: How to design & develop naturalness conversations maintained during Afaan Oromoo converse of the model?

RQ2: Which are the deep learning models that may improve the performance of chatbot for consultancy services?

RQ3: How can we improve the quality of legal consultant Chatbot model?

1.5. Objectives of the Study

1.5.1. General Objective

The General Objective of this study is to build Afaan Oromo text-based generative chatbot that supports legal consulting service to Oromia Supreme Court using deep learning techniques.

1.5.2. Specific Objectives

To achieve the above general objective, the following specific objectives are set.

- To review and assess literatures related to legal consulting service.
- To collect, analyze and prepare legal data
- To use an appropriate model and hyper-parameters for legal consultancy services.
- To evaluate the performance of the model using a proper functionality with different scenarios.
- To design, develop and deploy a prototype of legal consultancy services chatbot.

1.6. Scope and Limitation

1.6.1. Scope of the Study

The scope of this study is to develop a legal consultant service chatbot using deep learning techniques to improve access to legal services and provide personalized legal advice to individuals. The chatbot will be designed to provide legal guidance, advice, and assistance on various legal issues such as family law, criminal law, employment law, and contract law. The chatbot will be trained on a new dataset constructed by gathering legal concepts and information from various sources such as courts, lawyers, and online websites, and will be programmed to respond in Afan Oromo scripts.

The chatbot will provide an efficient and cost-effective means of consultation for court patrons, reducing costs and improving efficiency. Moreover, the chatbot will be able to provide customized and accurate advice based on individual needs by continually learning and improving over time through user interactions and feedback. The development of this chatbot has the potential to revolutionize the legal industry by making legal services accessible to a broader population and enhancing the legal system's efficiency.

1.6.2. Limitations of the Study

The purpose of this study is to develop a chatbot for legal consultancy services using appropriate deep learning techniques to yield better results. The chatbot will be designed to understand user requests and provide relevant legal advice and related services. The dataset used in the study is constructed by collecting legal concepts and terminology from various sources, including courts, advocates, lawyers, and online websites, as there is no pre-existing dataset on this topic.

The chatbot will be specifically designed to provide consultation services at the Oromia Supreme Court using the Afan Oromo language. Users will interact with the chatbot using written queries in this language, and the chatbot will provide responses in Afan Oromo scripts. However, it is important to note that the chatbot will not support speech conversations and will not consider conversations outside the scope of legal consultancy services. The OSC uses Afan Oromo language to provide judicial services in the region. As a result, the chatbot's work is limited to the Supreme Court of Oromia and users can only interact with the system using written queries in Afan Oromo language. Additionally, the chatbot does not have access to the Oromia Supreme Court database and cannot provide access to documents or other information.

1.7. Significance of the Study

When the legal consultancy service and prototype are fully operational, they will help judges, lawyers and the court community deal with cases more efficiently. Oromo societies primarily benefits from this and individuals in need of court legal consultancy services, while also extending its support to non-natives who can submit their questions in writing using the Oromic language.

Through language accessibility, reduction of judicial burden, elimination of juror bias, provision of timely consult services, and the promotion of judicial uniformity, the OSC aims to enhance access to justice and improve the overall efficiency and fairness of the legal system. The development of a chatbot for the Oromia Supreme Court using deep learning techniques is significant as it has the potential to increase access to justice and speed up court proceedings. By providing an accessible source of legal information, the chatbot could assist individuals in navigating court processes and procedures, and provide guidance on filing a case or finding a lawyer. This is particularly important given that ignorance of the law does not save one from

punishment according to the Ethiopian constitution, and the chatbot could help to minimize the problems caused by a lack of legal awareness.

Moreover, the findings of this study could be a milestone for further research and investigation, particularly for scientists engaged in the adoption of AI in the field of law. The use of deep learning in the development of a chatbot allows for increasingly accurate responses as more data and information are accumulated. As such, the chatbot could become an effective and reliable source of legal information for citizens without the need to wait for an attorney or court representation. Despite some limitations, such as the chatbot being limited to the Afan Oromo language and the costs associated with developing the deep learning model, the benefits of this technology in the legal system cannot be overstated. Overall, the use of deep learning to develop a chatbot for the Oromia Supreme Court can help to make the legal system more accessible and efficient.

1.8. Organization of the Thesis

The whole thesis is organized into six chapters. Chapter one presented background of the study, the motivation behind the study, a description of the problems with the research question to be raised and addressed, the scope and limitations, application of the study, the objectives of the study, and its explanation also included. The second chapter discusses literature review and related works on a court consultant service chatbot. Under this section, we tried to address an overview of a chatbot, history of a chatbot, types of a chatbot, application area of a chatbot, frameworks that used to develop a chatbot, overview of Deep learning in chatbot development, an overview of a chatbot, Afaan Oromo language, and related works with their comparison greatly discussed and briefly seated. Chapter Three states different related works which are done by other researchers about chatbot systems tools and methodologies used with their respective finding. This part has a different section. Those are data collection methodology, Deep learning classification algorithms, dataset preparation methodology, data preprocessing techniques, model selection techniques, system evaluation techniques, techniques used to design the user interface and simulation tools that we used in our study. Chapter Four presents design of Chatbot based legal consultant service system and contains different sections. Those are proposed system prototype architecture, implementation flow chart, proposed system model architecture, data preprocessing techniques implementation, feature extraction implementation, model

implementation, and response selection implementation. The implementation and evaluation of Chatbot based legal consultant service system are discussed in Chapter Five. We discussed the result that we get from the proposed system development. The first phase deals with the evaluation result that we got from the proposed model. These evaluation results have based on our dataset and based on human evaluation. The second phase is the discussion of about how the proposed system answers the research question and discusses what makes it different from the previous work on chatbot development. In chapter six, the obtained results from experimentation and implementation, dataset class distribution result, feature extraction result, models evaluation results, hyper-parameter tuning results were discussed in detail including Deep learning models comparison with and without remove stop-words to figure out the most performing model. It includes also the recommendation and future research directions of this study are clearly stated.

CHAPTER TWO

2. LITERATURE REVIEW AND RELATED WORKS

2.1. Introduction

Nowadays, AI is increasingly being utilized to solve problems of various magnitudes. AI chatbots employ NLP technologies to comprehend the purpose behind a query and provide solutions to customer issues without the need for human intervention. This chapter delves into the various types of chatbots, their evolution, and implementation strategies. It also examines the issues and challenges faced by chatbots, as well as the current state of the art in chatbot systems. Furthermore, to gain a better understanding of the concept and investigate problems, relevant literature has been reviewed which provides a technical evaluation of existing techniques used in legal consultation services. This evaluation begins by providing an overview of the OSC's Afaan Oromo language and then continues with a review of legal consultation modeling, chatbot history, and the available frameworks used for developing chatbots. It also discusses the different types of chatbots, deep learning techniques for legal text classification, algorithms, and techniques used by previous researchers in legal consultation, and related work that has been previously studied. Finally, the evaluation metrics for chatbots are presented.

2.2. Background of Chatbot Technology

2.2.1. Definition of a chatbot

A chatbot is an AI-powered software application that uses natural language processing (NLP) techniques to understand user inputs and provide relevant responses or actions (Joseph et al., 2016). Chatbots are designed to emulate human-like communication and interaction, providing users with a personalized experience through a chat interface. They can be used in various industries and applications, including customer service, marketing, healthcare, and legal consultation. Chatbots can be programmed with specific workflows, responses, and actions based on user input, allowing them to perform tasks such as answering questions, making reservations, providing recommendations, and even completing transactions. With the advancements in deep learning and NLP, chatbots are becoming increasingly sophisticated and can offer more personalized and intelligent interactions with users (OpenAI, 2021). By

leveraging the benefits of chatbots in legal consultation services, it is possible to provide a more efficient and effective experience to clients (J. Khare, 2019)**.** However, creating effective chatbots requires intricate interactions between multiple systems, making it a complex task. Nevertheless, with the potential to reduce the number of customer messages, average handling time, and customer care budget, chatbots present a promising solution for enhancing customer assistance in legal consultation services (OpenAI, 2021)**.**

In recent times, chatbots have expanded beyond their commercial applications and have become increasingly useful in various sectors such as education, public administration, legal institutions, business and e-commerce, healthcare, and entertainment (Shawar, B. A., & Atwell, 2007)**.** Chatbots are also employed to aid customers with enterprise products and services. In this section, we will examine various types of chatbots and their applications based on research conducted by (Mohammad Nuruzzaman, 2018),as discussed below.

2.2.1.1. *Alan Turing*

The main idea behind chatbots is to replicate human-like interactions in a convincing manner. In the early 1950s, Alan Turing, who is considered the father of computer science, introduced a basic test known as the Turing test. This test was designed to assess a machine's capacity to exhibit intelligent behavior. Turing suggested that a computer could be said to possess artificial intelligence and successfully passed the Turing test when a human evaluator cannot reliably differentiate the machine's responses from those of a human. Turing's 1950 article titled "Computer Machinery and Intelligence" formed the foundation for the development of intelligent computer programs. In this article, he posed the question "Can Machines think?" and introduced the Turing Test as a way to measure a computer program's ability to simulate human-like behavior in a real-time conversation (Turning, A. M, 1950)**,** (Turning, 1950)**.**

2.2.1.2. *Elizabot*

Elizabot, created in 1966 at MIT Lab (Weizenbaum, J., 1966) (Barker, S., 2017), is one of the earliest popular chatbots. It simulates human conversations using pre-programmed questions and answers. Elizabot functions as a psychotherapist, guiding users through further conversation and understanding. However, it faces limitations. Unlike humans, Elizabot cannot

learn new topics or respond to unfamiliar questions, leading to repetitive and monotonous interactions. Additionally, it struggles to recognize humor and often misinterprets user queries.

2.2.1.3. *Parry*

Parry**,** created in 1972 by psychiatrist Kenneth Colby, tries to simulate the conversational model of person with paranoid schizophrenia (*PARRY*, 2023.) (Colby, 1981). It implemented a crude model of the behavior of a paranoid, schizophrenic person. This behavior was based on judgments and beliefs about conceptualizations on reactions of accept, reject or neutral. It was a much more complicated and advanced program than ELIZA, embodying a conversational strategy. PARRY was evaluated using a version of the Turing Test in the 1970s. Experienced psychiatrists analyzed a mix of actual patients and computers by showing PARRY's response on tele-printers. Parry bot has its limitations in processing the input and generating appropriate responses and often repeats the same answer.

2.2.1.4. Artificial Linguistic Internet Computer Entity (ALICE)

ALICE, created by Richard Wallace in 1995, is a chatbot based on an updated version of Eliza's architecture. It utilizes pattern matching and a depth-first search approach for user interactions. ALICE uses AIML (Artificial Intelligence Markup Language) templates, which encode rules for generating responses based on dialogue history and user input (Weizenbaum, J., 1966).The user's sentence is processed and matched against decision tree nodes, triggering an answer or action from the chatbot. However, the recursive nature of AIML templates sometimes leads to responses that lack meaningfulness (Jurafsky, D. and J.H. Martin, 2017)**.**

One drawback of ALICE is the limited modeling of personality traits, attitudes, mood, emotions, and physical states, which affects its ability to exhibit realistic behavior (Lemaitre, C., C. A. Reyes, 2004). Furthermore, ALICE lacks reasoning capabilities and struggles to generate human-like responses, making it unable to pass the Turing test (the test for human-like intelligence).

2.2.1.5. Jabberwocky

Jabberwocky, developed by Rollo Carpenter in 1982 (Carpenter, 2022.), is one of the earliest attempts at creating an artificial intelligence for human interaction. It was designed to operate as a text-to-voice system (Jabberwacky, 1982) And aimed to mimic natural human conversation for

entertainment purposes (Jabberwacky, 1982)**.** Jabberwocky was among the first chatbots to utilize AI techniques to simulate meaningful human-like conversations. It had the ability to continuously learn during interactions and was based on heuristics-based algorithms instead of artificial life models like neural networks or fuzzy pattern matching. The chatbot stored user inputs and employed contextual pattern matching techniques to generate appropriate responses (Jabberwacky, 1982).

2.2.1.6. *Watson*

Watson, a rule-based AI chatbot developed by IBM's Deep QA project (Nay, C., 2011), is designed for information retrieval and question-answering. It utilizes natural language processing and hierarchical machine-learning methods. Watson employs various mechanisms to assign feature values and generate responses. However, it has limitations, including the inability to process structured data directly, higher maintenance costs, and a focus on larger organizations. Teaching Watson to maximize its potential requires significant time and effort. Watson operates as an open-domain operating system (Nay, C., 2011).

2.2.1.7. Mitsuku

Mitsuku is a widely used human-like chatbot created by Steve Worswick using AIML (Shawar, Bayan and Atwell, 2002). It engages in general typed conversations and utilizes heuristic patterns for natural language processing. Mitsuku can integrate with platforms like Twitter and Telegram, and it learns from conversations and retains user details. It supports multiple languages and reasoning with objects. However, Mitsuku lacks dialogue management components and requires extensive training data (Worswick, 2010).

2.2.1.8. *Alexa*

In 2014, Amazon introduced Alexa, an intelligent personal assistant (Hard, 2016). According to the source, Alexa is capable of understanding and applying around 50,000 words in various contexts (Hard, 2016). It can provide responses and retrieve relevant content based on user queries, such as information about Whirlpool products (Hard, 2016) (Amazon Alexa, 2023.). However, while Alexa works with a limited number of services, it has yet to make a significant impact on the connected home market.

2.2.1.9. Amazon Lex

Amazon Lex is an AWS service that enables conversational interfaces using voice and text (Amazon Web Services, 2018). It offers natural language understanding and speech recognition, creating engaging user experiences. Integration with AWS Lambda allows for backend logic execution. However, Lex has limited language support and requires a structured integration process with complex dataset preparation (Amazon Web Services, 2018).

2.2.1.10. *Cortana*

Cortana is a virtual assistant developed by Microsoft that utilizes the Bing search engine to assist customers with tasks and provide answers (Risley, 2016) (*Framework Definition*, 2021a)**.** It was initiated by the Microsoft Speech products team in 2009, led by General Manager Zig Serafin and Chief Scientist Larry He. Cortana is capable of recognizing natural voice commands and retrieving information from Bing to provide responses.

Table 2.1: Comparison of chatbot developed

Chatbot and	***Approach***	***Architectural model***	***Application***	***Drawback***
ALAN TURING (1950)	Turing Test. artificial intelligence (AI)	Natural language processing (NLP) architecture. -concept of a Turing Machine.	to provide customer service, to answer questions, or to provide advice, for virtual assistants, and conversational agents.	it is limited by the Turing Test. Is limited by the number of instructions that it can process
ELIZA (1966)	NLP, pattern matching technique	-Retrieval based model -a client-server architecture	- natural language processing, artificial intelligence, and psychotherapy. - Tongue in cheek simulation of Rogerian psychotherapist	- is limited and can only recognize a limited number of keywords and phrases. -No logical reasoning capabilities, inappropriate responses.
PARRY (1972)	-Conceptual ontologies, Turing test. - use a psychoanalytic approach to simulate a conversation.	- a rule-based system - Retrieval based model	Mimic the behavior of a person with paranoid schizophrenia. -behavior of paranoid schizophrenic patients and to help psychiatrists better understand their patients.	- was limited by its rule-based system. -Slower to respond and unable to learn from the dialog. - its responses were often too literal.

jabberwocky (1988)	-natural language processing. -Contextual pattern matching	-knowledge base of facts and rules. -Generative model	-chat rooms, customer service applications, and educational programs. -To simulate natural human chat for entertainment.	- its limited understanding of natural language. -limited in its ability to respond to user input in a meaningful way.
A.L.I.C.E. (1995)	NLP, AIML heuristic pattern matching.	-front-end user interface, -a back-end knowledge base, -a middleware layer -Retrieval based model	-customer service, online education, and entertainment. Pattern matching heuristics are used for user input for the human communication interface	-limited by its lack of understanding of context and its inability to learn from user input. - limited by its reliance on AIML, which can be difficult to understand and modify.
Watson (2006)	NLP, IBMs Deep-QA software and Apache UIMA	-Hybrid which combine rule-based system with a machine learning system. -Retrieval based model	-customer service, virtual assistants, and natural language processing. -QA system that won the 2011 Jeopardy competition	-It can be difficult to train the machine learning system to accurately interpret user input. -Does not process structured data. -Supported on English

Mitsuku	NLP,DNN, Java, JS, Objective C, TTS, STT	Retrieval-based system	-Customer service support, customer engagement, business process automation. -The computer software serves as an intelligent personal assistant	-It relies heavily on manually created templates and rule sets in order to accurately process user input. -Difficult to hear the interlocutor.
Google (2020)	-NLP techniques and rule-based algorithms - machine learning technologies	-Google Cloud Natural Language API -A speech recognition module, NLP, and an interface. Generative model cute.	-Customer service, teaching, entertainment, -Virtual assistant, helping users with daily tasks and providing information when needed.	-difficulty dealing with complex queries.-It has no personality -may not provide accurate answers. -Its question may violate the users' secrecy.
ALEXA (2015)	NLP techniques and rule-based algorithms DL technologies, AIML and deep learning algorithms, TTS, STT, Python, Java, node JS	-The conversation design layer -(NLP) layer -ML layer -AI engine, a database, NLP, and an interface. -Generative model	To automation, to teaching, to facilitate customer service, provide product recommendations, and even provide entertainment.	-May not be able to accurately interpret user input or response to user input in the way desired. -It requires a lot of data to be trained and is not able to handle complex queries.

Cortana (2015)	Python, Java, JS, NLP and machine learning technologies.	-Speech recognition, NLU and deep learning algorithms. -knowledge base -Generative model	-Customer service, teaching, entertainment. -The intelligent assistant makes reminders, sends emails, etc.	It requires a lot of data to be trained and is not able to handle complex queries. It is not able to understand context or the user's feelings.
Amazon Lex	DL,NLP	-Speech recognition, NLU and DL algorithms.	-Customer service, teaching and entertainment.	-It requires a lot of data to be trained and is not able to handle complex queries. -it is not able to understand context or the user's feelings.

2.3. Types of Chatbot

Chatbot applications can be classified into four groups: service, commercial, entertainment, and advisory chatbots (G. Cox, 2019). Service chatbots assist customers with logistics, deliveries, and communication. Commercial chatbots streamline purchases and promotions, while entertainment chatbots engage users with sports, events, and ticket deals. Advisory chatbots offer suggestions, recommendations, and support. Additionally, chatbot applications can be categorized as task-oriented and non-task-oriented (K. D. Ashley, 2017)**.** Task-oriented chatbots help customers complete specific tasks, while non-task-oriented chatbots focus on answering questions and providing entertainment. According to the literature (Nuruzzaman M, 2018)**,** they categorized chatbot applications into four groups such as goal-based, knowledge-based, Service-based and response generated-based as shown in Figure.1.

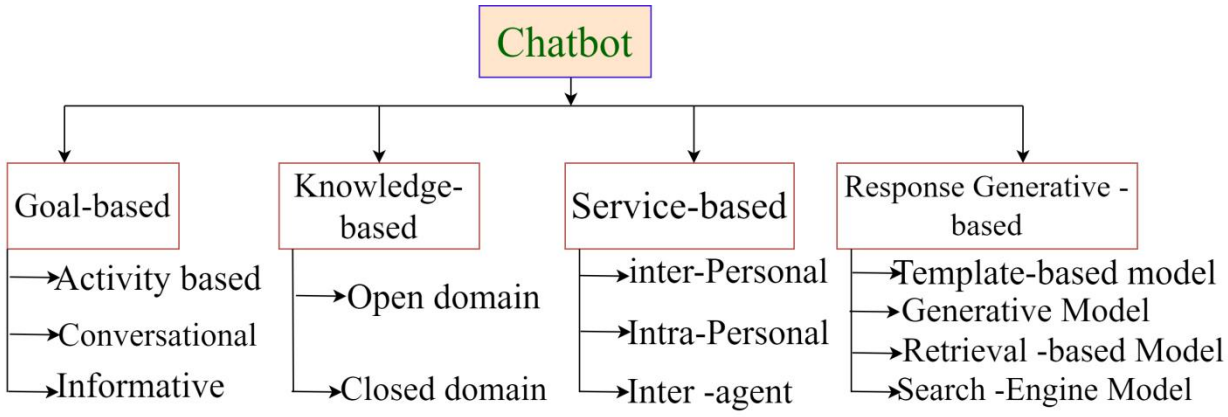


Figure 2.1 ***Taxonomy of Chatbot Application***

Goal-based chatbots are categorized as per the primary goal aim to achieve. They are intended for a specific task and setup to have small talks to get information from the user to finish the task. For example, a company set up chatbot on their websites to assist the customer to answer their question or to solve their problems.

a) Knowledge-based Chatbot

Knowledge-based chatbots are categorized according to the knowledge they retrieve from the underlying data sources or the volume of data they are trained on. The two key data sources are open-domain and closed-domain. Open-domain data sources response depends on general

topics and respond applicably. For example, open domain are Allen AI Science and Quiz Bowl. Closed-domain data sources emphasis on a specific knowledge domain. All information needed for answering the question is provided in the dataset itself.

b) Service-based Chatbot

Service-based chatbots are categorized according to facilities delivers to the customer. It might be personal or commercial purpose. For example, Logistics Company could deliver copies of message documents through chatbot rather than phone calls or customer can make a meal order from MacDonald.

c) Response Generated-based Chatbot.

Response Generated-based chatbots are categorized based on what tasks they accomplish in response generation. The answer models take input and output in natural language text. The dialogue manager is responsible for joining response models together. To produce a response, dialogue manager follows three steps. First, it practices all response models to generate a set of responses. Second, get back a response based on priority. Third, if no priority response, the response is chosen by the model selection policy. The emphasis of the literature is on response generated-based chatbot. In this category, there are numerous response models that are based on four groups as shown in Figure 2.2

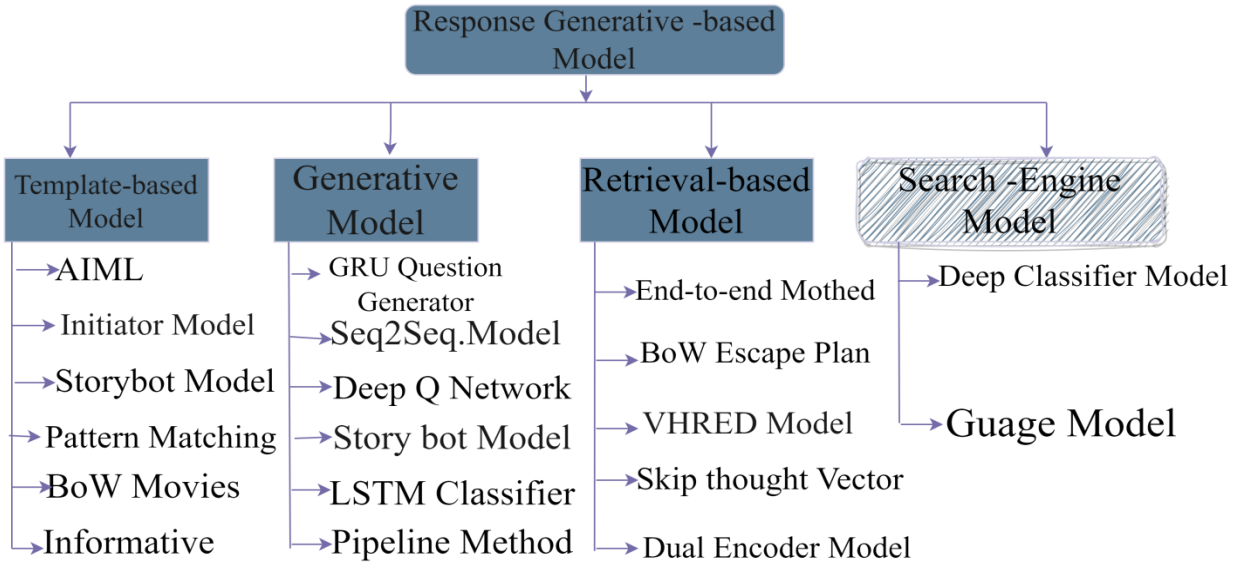


Figure 2.2 ***Classification of Generated-based Models***

2.4. Application Domain of Chatbot Technology

2.4.1. Chatbot in Court/legal Domain/

AI chatbots is important, because it can perform more difficult tasks than humans. Artificial intelligence tools often complete tasks. There are rapid advances in Artificial Intelligence (AI), which will have significant implications for both the legal profession and certain areas of law (K. D. Ashley, 2017). The chatbot's has been applied to the legal or law field to improve the various problems that the legal profession has faced over the years and also it is now being implemented. AI application in law (M. Partner et al., 2022)**.**

***Legal Conversational agents as customer support*:** Legal conversational agents, such as customer support chatbots, offer several advantages in the legal industry (Dale, 2016)**.** They provide instant responses to customer inquiries, eliminating the need for waiting and ensuring timely communication. These bots can deliver real-time answers, helping businesses retain prospects and provide efficient customer service. Alira's conversational agents specifically address legal questions, offering comprehensive and valid information tailored to the customer's needs.

Automating Processes: Chatbots can automate tedious and time-consuming tasks in court proceedings, including filing paperwork, scheduling hearings, archiving documents, and more.

Connecting with Lawyers: Chatbots can connect potential litigants with lawyers who can provide advice and represent them in court if necessary, giving greater access to services for those on limited budgets or with disabilities preventing them from attending a physical court hearing.

Analyzing Legal Documents: The chatbot could be used to analyze legal documents associated with each case in the Oromia Supreme Court and make recommendations accordingly that could help improve the efficiency of court proceedings.

Predicting Outcomes: By leveraging deep learning technologies, such as neural networks and natural language processing, the chatbot could also predict outcomes for certain cases based on past data available within the court system. This would help judges make informed decisions by providing them with insights not readily available through manual data analysis.

Appointment scheduling: those initial consultations can be scheduled through the chatbot, eliminating the involvement of lawyers or receptionists. This is an excellent and well-established use of chatbots.

Legal Adviser Support: Legal Adviser Support: In this case, lawyers or other professionals may request a system, then a system will review the relevant law stored in its system, gathers and give a response highly relevant to the answer. The US law firm Baker is using ROSS, which a legal adviser system is built by IBM Watson.

2.4.2. Chatbot in Health

Healthcare conversational bots provide advice based on customer symptoms and help locate nearby clinics or hospitals. They use medical data to offer reliable information and can schedule appointments with physicians. An example is Dr. Dean chatbot, which recommends remedies based on health status and follows up with additional suggestions if needed (Gentsch, 2019).

2.4.3. Chatbot in Customer Service

Efficient customer service is vital for organizations (L. Cui, S. Huang, F. Wei, C. Tan, C. Duan, 2017). Chatbots offer direct and personalized support, reducing wait times and improving customer relations. They monitor the customer's journey, identify issues, and provide automated help. Chatbots save companies money by bypassing slow support channels. (X. Zhang et al, 2020) developed a personalized chatbot using neural networks for consumer banking call centers. The chatbot assists customers by analyzing bank dialogue transcripts between the customers and Customer Service Officers (CSOs) to identify their reasons for calling.

2.4.4. Chatbot in Agricultures

A Chatbot in agriculture is designed to assist the farmers on how to enhance their productivity. Since agriculture is the key to human life, it is critical to support farmers and allow agriculture industries to increase production, customer service, etc., and then farmers will receive information on how to improve their productivity. The chatbot can offer information that helps farmers to improve their yield based on weather, rainfall, and soil type of the environment and suggest which crop is more suitable to grow (K et al., 2019). As discussed in the cited

study, the chatbot can help users by answering their basic questions on farming, and they understand the type of crop to be grown based on different parameters.

2.4.5. Chatbot in Education

Chatbots have emerged as valuable tools in the education sector, offering various benefits and enhancing the learning experience (Okonkwo & Ade-ibijola, 2021).They provide support to students by offering immediate assistance, material integration, and easy access to information. Chatbots can address the challenges students face in asking questions, such as fear of approaching their teachers. For example, (F. Colace, M. De Santo, M. Lombardi, F. Pascale, A. Pietrosanto, and S. Lemma, 2018) developed a chatbot system focused on courses like Fundamentals of Computer Science and Computer Networks. This chatbot allows students to ask questions and receive answers, facilitating their understanding and providing a user-friendly interface for interaction and navigation within the course content.

2.5. The Available Framework

A chatbot development framework is a set of pre-defined functions and classes that help developers build chatbot applications more efficiently. The Bot Framework is a platform that enables the creation, connection, testing, and deployment of intelligent bots. These frameworks provide tools to streamline code development, address common challenges, and facilitate language processing, dialog management, and user connectivity across different conversation experiences and languages (“Framework Definition,” 2021b)**.** By using these frameworks, developers can leverage powerful features to build robust chatbot applications. Modern chatbots make use of NLP and DL concepts (Kandpal, M., Goyal, R., & Gupta, 2020). Some popular chatbot development frameworks are:

Microsoft Bot Framework: - Microsoft Bot Framework is a comprehensive offering to build and deploy good quality of chatbots for users to enjoy their favorite conversation experiences. It provides a set of services, tools, and SDKs that helps developers in building a rich conversational chatbot (S. Machiraju and R. Modi, 2018). The Microsoft chatbot Framework’s integration component is impressive. It can be integrated with Slack, Facebook Messenger, Telegram, Web chat, Group Me, SMS, email and Skype.

Dialog Flow: -Dialogflow, developed by Google, is a customizable chatbot solution. It offers a free standard edition, but paid plans are available for higher query volumes. With support for voice and text-based assistants, it is beginner-friendly and provides 33 pre-built agents trained in various domains. Supporting multiple languages, sentiment analysis, small talk functionality, and live analytics, Dialogflow also offers Internet of Things (IoT) integration for home automation (Google, 2023).

IBM Watson: - is the most popular chatbot framework as it offers advanced functionality and features for the developers like automated predictive analysis, tone analysis of the user query, visual recognition security, multiple language support etc. It also offers Watson Assistant GUI for non-technical business users for handling queries. One more interesting feature of Watson bot is that it doesn't collect the data for building bot instead allows developers to use private cloud for storing data, hence maintaining confidentiality and data security. As far as pricing is concerned it comes with wide options like Standard, Plus, Premium and Deploy Anywhere (IBM, 2021) .

Amazon Lex:-is the most advanced chatbot framework as it provides advanced deep learning functionalities of automatic speech recognition (ASR) for converting speech to text, and natural language understanding (NLU) to recognize the intent of the text. It is highly scalable due to integration with **AWS Lambda**. AWS Lamda is the most advanced server less computing platform as it offers full stack backend infrastructure and code. This gives developers time to focus solely on building the application instead of deployment part. It is the most cost effective chatbot solution as it provides a **pay-as-you-go** pricing model (Amazon Lex Documentation, 2020)

Botsify: - is the drag and drop chatbot building tool which makes it extremely easy for developers or even non-technical business users to build a bot for their customized needs. It also offers human agent handover feature means it allow you to transfer conversations to a human agent at any point in the chatbot-to-human conversation which is not available in any other bot framework. It also offers live chat support for any queries and doubts. It provides the facility of conversational forms means it can collect handful of information like name, address, email,

mobile number of the user in the form of good looking forms. It also offers integration with multiple channels like slack, RSS feed, Google sheets, website, google search etc (Botsify, 2021)

Pandorabots: - is a chatbot platform that enables users to create and publish chatbots on the internet, smartphone applications, and messaging apps such as LINE, Slack, WhatsApp, and Telegram AI-powered chatbots.The Pandorabots chatbot use AIML (artificial intelligence markup language) scripting language for building conversational bots. The major drawback is that it uses AIML scripting which is only used by Pandarobots and apart from that it is quite less accurate in comparison to other competitors in the market (Pandorabots, 2019) .

Chatterbot: is a Python library that makes it easy to generate automated responses to user input. Chatterbot uses a selection of machine learning algorithms to produce different types of responses (Chatterbot GitHub Repository:, 2022). Before using this library, we have to install the library by using pip install chatterbot commands. This library contains different classes and methods that are uses for performing different tasks. To use this library we import those classes and methods in our project and we develop chatbot easily. We used this library like the following sample code.

API.ai:- is another web-based bot development framework acquired by Google in 2016. API.ai seems to have found out the flaw of letting the users define entities and intents by entering multiple utterances and hence provides a huge set of domains. It can be integrated with many popular messaging, IoT and virtual assistant's platforms. Some of them are Actions on Google, Slack, Facebook Messenger, Skype, Kik, Line, Telegram, Amazon Alexa, Twilio SMS, Twitter, etc. Some of the SDKs and libraries that API.ai provides for bot development are Android, iOS, Webkit HTML5, JavaScript, Node.js, Python, etc. API.ai is built on few concepts as follows:

1. **Agents**: Agents corresponds to applications. Once we train and test an agent, we can integrate it with our app or device.
2. **Entities**: Entities represent concepts that are often specific to a domain as a way of mapping NLP (Natural Language Processing) phrases to approved phrases that catch their meaning.

3. **Intents**: Intents represents a mapping between what a user says and what action should your software takes.
4. **Actions**: Actions correspond to the steps your application will take when specific intents are triggered by user inputs.
5. **Contexts**: Contexts are strings that represent the current context of the user expression. This is useful for differentiating phrases which might be ambiguous and have different meaning depending on what was spoken previously.

2.6. Approaches of Chatbot

2.6.1. Early Approaches in Chatbot Development

some of the early approaches to chatbot development that do not use ML. This includes rule-based and pattern-matching approaches.

Rule Based approaches; involve setting up rules for how the chatbot should respond to given requests. Using rules, programmers can create comprehensive databases with hundreds or thousands of possible conversations that the bot can engage in. Rule-based bots are highly customizable and can be programmed to handle complex conversations far better than pattern matching bots. They are also more accurate because implicit meaning, grammar, syntax and context can be accounted for when defining rules. ELIZA and PARRY chatbot are the main chatbot that was used these techniques. (Colby, K. M., Weber, S. G., & Hilf, 1971).

Pattern Matching is a technique used by chatbots to identify patterns in user input and match them to a predefined set of patterns. This allows the chatbot to respond to incoming queries in an intelligent manner. This approach involves detecting keywords and phrases in the user's input, which then trigger the response from the bot. In chatbot development, the pattern matching technique is used to identify user input as `<pattern>` and provide an appropriate answer stored in `<template>`. The `<pattern>` `<template>` pair used in pattern matching approach is constructed manually. The approach is used in both early and advanced chatbots. The sophistication of the procedures utilized in them varies. For example, according to the description of (Hussain, M., Abbas, H., Hussain, S., & Kang, 2019), ALICE Chabot uses more complex pattern matching rules. The approach has a scaling problem because all possible patterns are built manually.

Generally, there are three most common languages such as AIML (Artificial Intelligence Mark-up Language), Rive script, and Chat script for the implementation of chatbots with pattern matching. AIML is an open-source language developed from XML and used to build a conversation flow in chatbots (Wallace, 2003). The AIML was created from 1995 to 2000, and the first chatbot that used AIML is ALICE. AIML is divided into three categories: chatbot rules, patterns to describe user input, and templates to describe the chatbot's answer.

```
<Category>
   <Pattern>Manni Murtii Tajaajila Akkami Kenna?* </pattern>
   <Template> Haqaa Eega< template>
</category>
```

```
<Category>
   <Pattern>What Services Does the Court Provide?* </pattern>
   <Template>It Maintains Justices< template>
</category>
```

Figure 2.3 ***Example of AIML for Chatbot Development***

Chat script is a rule-based chatbot creation expert system developed in 2011. It can work by providing a set of templates and responses that enable it to answer questions, process commands, and understand requests from users. It generally work by taking input from the user, analyzing it in relation to the known data set, and then responding with an appropriate response. Different chatbot such as Mistsuku, Chip vivant, Rosette, and Suzette are implemented using Chat script language (Wallace, 2003).

```
Concept: ~ greeting [Akkam nagaadha]
Concept: ~ good_day [ Nagaan oolaa]
Topic: ~ chatbot [ ~greeting]
t: ~greeting
#! Hello
u:(* ~greeting *) [Hi][Hello], [maqaan kee eenyu? /What is your name?]
#! Maqaan Koo / My name is
u: (*name* is_) Bagan si baree_0! / Nice to meet you_0!
$username='_0
#! Guyyaa Gaariin siif hawwaa / I wish you a good_day
U:(<<!not* good day*>> good_day $usernames
```

Figure 2.4 ***Examples of Chat Script Language***

The Rive script is an open-source and line-based scripting language that was created in 2009 (Adamopoulou & Moussiades, 2020). In the Rive scripting language, the user input and response are represented by the symbol + and - respectively. It consists of a few simple rules that are combined in powerful ways to create a successful chatbot identity. Generally, this chatbot is easy to learn, quickly to type, and easy to read and maintain. Fig.2.5. shows how the Rive script has been written to design a chatbot.

```
+Hello

-Hi! Maqaan kee eenyu? / What is your name?

+Maqaan koo* / My name is*

-<set name=<formal>>Baga wal baranne / Nice to meet you, <get name>!

+Good day/ Nagaan oolaa<get name>
```

Figure 2.5 ***Example of Rivescript Language***

2.6.2. Machine Learning Approaches Based Chatbot

In today's era, the rise of intelligent machines has revolutionized task completion. Advancements in artificial intelligence (AI), machine learning (ML), and deep learning have enabled machines to mimic human-like interactions. AI, particularly ML, utilizes algorithms and data to perform various tasks, permeating our daily lives. Within ML, there exists a branch called conversational models, driven by natural language processing (NLP), which has given rise to chatbots. These modern chatbots, leveraging ML, do not rely on predefined rules or pattern matching approaches, allowing for more sophisticated and dynamic interactions with users.

Machine learning approaches in chatbots can be classified into two categories: retrieval models and generative models (Mohammad Nuruzzaman, 2018)**.** These chatbots utilize machine learning techniques to learn from data and improve their performance. Deep learning algorithms such as CNN, RNN, and DNN have been widely employed in the development of modern chatbots (Mohammad Nuruzzaman, 2018)**.** Deep learning, a subfield of machine learning, leverages multiple layers of data processing to enable machines to learn through various levels

of abstraction. It has gained significant attention in the ML and NLP research communities, particularly for NLP tasks (Kapetanios, 2016)**.** Modern chatbots incorporate natural language understanding and employ deep learning algorithms to enhance their capabilities (Kandpal et al., 2020)**.**

Deep learning models use artificial neural networks to imitate how the human mind interprets and learns from data. It is one part of machine learning that uses a high level of abstraction to process data. Most DL method uses an ANN to build intelligent models and solve complex problems. ANN uses the approach of how the human brain computes information. A deep neural network is composed that has multiple hidden layers. Neural networks have three layers as depicted in Fig 2.6, and every layer consists of multiple nodes (neurons). Neurons are the building components of neural networks. Neurons in one layer are interconnected to neurons in two surrounding layers, which help to flow data between layers. Lines used to connect every neuron are called weights, denoted as wij.

➢ **Input layer*:*** the layer consists of the list of input features and often it is called a visible layer. It accepts input data and introduces it to the rest of the network. It has a responsibility to initialize the first weights.

➢ **Hidden layer*:*** hidden layers are the hidden pulp of the neural network that is responsible for the better performance and complexity of the model. Tuning the network with the correct hidden layer and neurons in each hidden layer needs more attention because hidden layers are where the operation is done in every single neuron, as shown in Fig.2.6. Where w denotes weight, x denotes input value, b denotes bias, and δ represents sigmoid activation function.

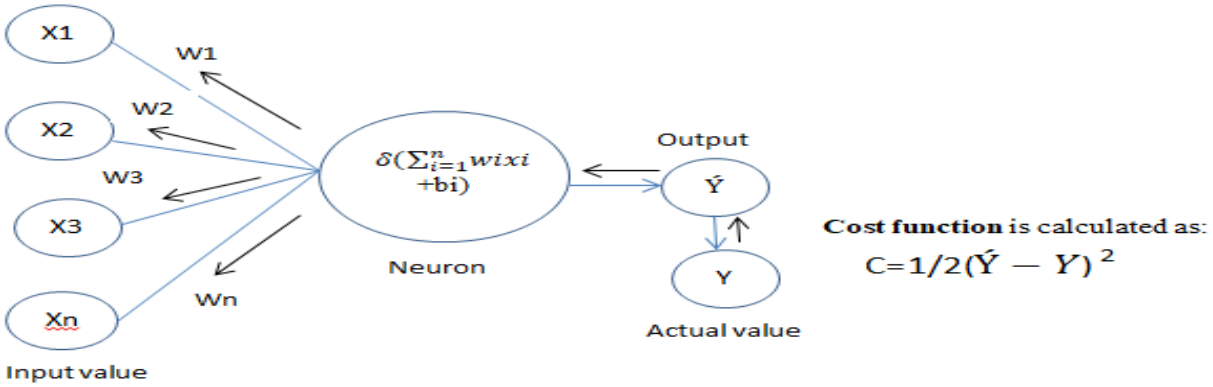


Figure 2.6 ***The Neurons of Hidden Layers Operation*** **(Patil et al., 2020)**

➢ **Output layer:** the last layer and where the data is generated out from the model. This layer has the same number of neurons as it does outputs. If the designed model is a regressor, the output layer has only one node.

In this study, the dataset was prepared with multiple patterns and responses that belong to different classes (tags). And then the patterns were preprocessed, embedded into vectors, and then trained with the proposed deep learning model. Then the model classifies them into appropriate classes to answer the response that belongs to that tag for the users. Assume that, if the user input is "Mala walhimachuun itti xiqaachuu danda'uu naaf ibsuu dandeessaa?" Then the model processes this input question through different steps to generate a response, as shown in Fig.2.6. Lastly, the model classifies the input pattern into the appropriate tag with the highest confidence value to fetch the random response that belongs to that tag. At the output layer the SoftMax activation function is used because the problem multi class classification.

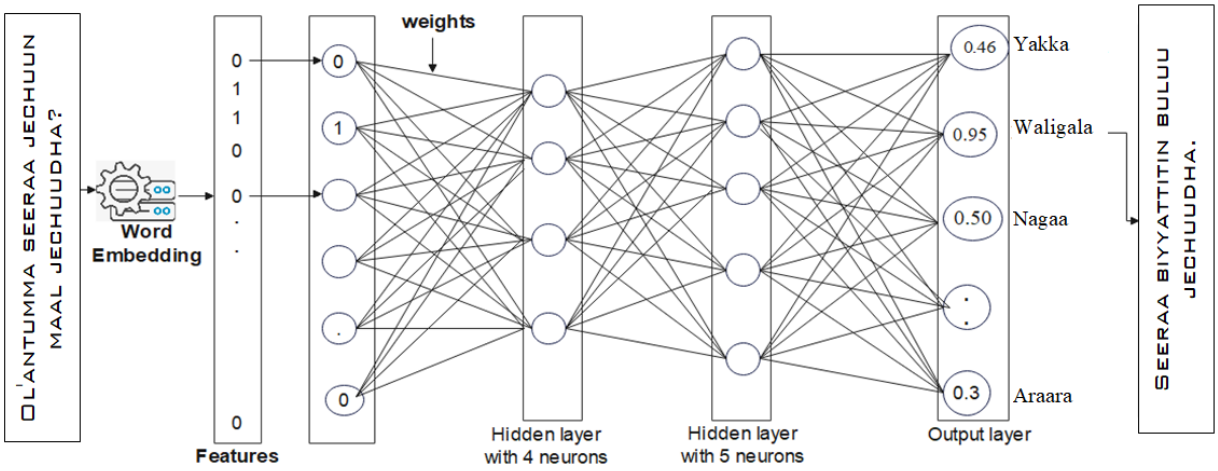


Figure 2.7. ***How Neural Networks Operate Textual Data for Chatbot Development***

The input layer receives the vector-represented data, and all values are computed sequentially using the network layers. The output of one layer is utilized as input for the subsequent layer. The output values are computed by multiplying the input value and the weight for every unit in a current layer and adding them together. After that, a non-linear activation function like rectified linear unit (ReLU), sigmoid, etc., is applied to buffer the data before being sent to the succeeding layer. As the number of hidden layers rises, the output data obtained in the final layer becomes more abstract than the raw input data. As a result, setting the number of layers and neurons in the hidden layers needs more consideration to optimize model performance.

2.7. An Overview of Afaan Oromo Language

This study focuses on the Afan Oromo language, which is spoken by the Oromo people in Ethiopia and neighboring countries. Afaan Oromo is an important tool for communication, enabling people to interact, express themselves, and share knowledge through written symbols, signs, and spoken language. It is a member of the Cushitic group, a branch of the Afro-Asian language family, and is the third most widely spoken language in Africa, with over 36 million speakers (Tolessa, 2013)**.**

The Oromo people have a rich oral culture, as they have not developed an independent writing system for their language. However, in the 19th century, scholars began using the Latin alphabet to write in Oromo, and in 1991, the language was officially written in the form of the Latin-based QUBEE alphabet (Negesse, 2015)**.** Afaan Oromo is widely used in both written and

spoken form in Ethiopia, and it is the working language of the Oromia regional government, primary and secondary schools throughout the region, and public and private offices (Asafa, 2010)**.**

In particular, the Oromia Supreme Court has been using Afan Oromo as a working language since 1991, including the High Court and the First Instance Court. All legal proceedings, including lawsuits, answers of the accused, witness testimonies, and judicial procedures, are conducted in Afaan Oromo (Asafa, 2010)**.** Overall, Afaan Oromo plays a significant role in the daily lives of the Oromo people and in the legal and administrative affairs of Ethiopia's largest region.

2.7.1. Afaan Oromo Alphabets and Orthography

This study focuses on the Afan Oromo language, which uses the Latin alphabet or characters (Roman orthography) for writing and has several vowels and consonants similar to Amharic, Omotic, Tigragn, English, and other languages. Afan Oromo has five vowels, namely A, E, I, O, and U, and 27 consonants, including combined consonants (CH, DH, NY, SH, and PH) known as “Qubee Dachaa.” The Afan Oromo alphabet consists of thirty-two (32) letters, as shown in Table 2.2, and the letters "Z", "P", and "V" are not part of the Afan Oromo script (Wikipedia., 2021)**.**

Table 2.2: Afaan Oromo Alphabets [1]

Aa	Bb	Cc	CHch	Dd	DHdh	Ee	Ff	Gg	Hh	Ii
Jj	Kk	Ll	Mm	Nn	NYny	Oo	Pp	PHph	Qq	Rr
Ss	SHsh	Tt	TSts	Uu	Vv	Ww	Xx	Yy	Zz	

2.7.2. The writing system of the Afaan Oromo language

The writing system, technically known as script or orthography, consists of a set of visual elements, shapes, or characters, or graphs that contain certain structures in a language system.

[1] https://www.africa.upenn.edu/Hornet/Afaan_Oromo_19777.html

We use different writing systems with symbols representing different things: logographic, syllabic, and alphabetic. In alphabetical order, a symbol connects to a sound in the language, rather than to an object, an idea, or a word.

Each language has its own rules and syntax. In the English alphabet, twenty-six letters are used, including vowels and consonants. However, the Oromo alphabet is thirty-two in number and includes five vowels (A, E, I, O, and U) and 27 consonants, including combined consonants (CH, DH, NY, SH, and PH) called “Qubee Dachaa” (Tolessa, 2013)**.**The main difference between the Oromo and English spelling systems is how a word is spelled. While there are no specific spelling rules for English words, Oromo words follow a set of rules called "**Serluga**" that anyone can follow to write and read Oromo words (Tolessa, 2013) (D. Duresa, 2016)**.**

a) Spelling Rules

"Qubee" replaced the various Oromo alphabets with Latin alphabets, helping to standardize the Oromo alphabet. However, spelling variations can still occur due to personal preferences and regional differences in pronunciation. In Afaan Oromo, a word cannot begin or end with two consonants, and most words end with a vowel. For example, the English word "defendant" would be written as "himatama" in Afaan Oromo. Misplacing a single word in a sentence can completely change its meaning and spelling in Afaan Oromo.

In Afan Oromo, there are four rules for writing words: "Sagalee Laafaa" or "Sagalee Jabaa" and "Sagalee Dheeraa" or "Sagalee Gabaabaa". "Laafaa" and "Jabaa" are based on the number of consonants. When the same consonants are written next to each other, the sound is emphasized ("Sagalee Jabaa" in Afan Oromo). For example, "Gubbaa" means "on top". If a single consonant is used, it is "Sagalee Laafaa". For example, "Gubaa" means "burn". "Dheeraa" and "Gabaabaa" are conjunctions with repeated vowels. When vowels are repeated or sounds are stretched or lengthened, it is called "Sagalee Dheeraa". For example, "Boonaa" means "proud man". Otherwise, the sound is short, which is called "Sagalee Gabaabaa". For example, "lafa" means "ground" (Mohammed Abdurahman, 2021)**.**

In Afan Oromo, when two different non-digraph consonants appear in a series, it is called “Irrabuta”, which is a phoneme that is uttered for a very short time. For example, in words

like "Harmaa", "Gamnaa", "Torba", Garbaa, etc., there is a phone [‘I’ or 'i'] in between them that is pronounced for a short time but is not written. Vowels cannot be changed without a break, and there is a consonant called "Hudha" among them. Spelling preferences and accents may vary depending on the breaks used. For example, “very” can be written as baa'yee, baayee, baa'ee, or baay'ee, and “to hear” can be written as dhagahuu or dhaga'uu. The diacritical marker indicates that the vowels are produced separately, not as a diphthong.

b) Grammar

According to (Thompson, 2021)**,** the Oromo grammatical system is very complex and exhibits many characteristics common to other Cushitic languages, namely, it is a distorted language that uses postpositions rather than prepositions.

Nouns and adjectives: Most Afaan Oromo adjectives and nouns are marked for female or male. When biological sex is associated with a particular suffix, names have a natural masculine or feminine gender that cannot be determined by the form of the noun, such as, **-eessa** for masculine and **-eetti** for feminine nouns, for example, **obboleessa** “brother” and **obboleetti** “sister”. All adjectives and nouns are marked to number plural and singular. For example: for female nouns ‘haroo’ (lake) – ‘harittii’ (the lake) and for male nouns ‘nama’ (man) – ‘namticha’ (the man).

Pronouns: Afana Oromo pronouns have a person (3rd, 2nd, 1st), case (locative, nominative, ablative, genitive, accusative, dative, instrumental), number (plural and singular), and so on. There is a difference between near and far display pronouns, for example, ‘kana’ (this) and ‘san’ (that).

Verbs: Oromo verbs consist of suffixes plus the stem, which represents person, gender, number, tense-aspect, mood, and voice. Verbs agree with their subject of person and number. Verbs, except the verb "be", agree with their subject in gender, when the subject is the third person singular pronoun “he” or “she”. In Afaan Oromo verbs there are four moods (indicative, interrogative, imperative, and jussive) and three sounds (active, passive, and the so-called auto benefactive (semi-passive/middle)).

2.8. Related Works

In Recent years, there has been an increasing interest in improving chatbot performance to provide assistance to users across various industries and businesses. Although several models have been introduced by researchers, deep learning models are still in the early stages of chatbot development (Kandpal et al., 2020). Several researchers have conducted studies on different chatbot services. In our thesis, we have thoroughly reviewed relevant papers that are closely related to our research topic.

Shubhashri G et al, (Shubhashri G et al., 2018)**,** worked on *"LAWBO: A Smart Lawyer Chatbot"*. *LAWBO* is a task-oriented chatbot that is specialized in providing legal assistance. The accuracy of the chatbot in providing legal advice is heavily dependent on the quality and size of the training dataset used. Therefore, a comprehensive training dataset that captures a diverse range of legal concepts and user queries is crucial to improve the chatbot's accuracy. The authors have provided a novel solution to improve access to legal assistance for users, while showcasing their technical expertise in data preprocessing, feature extraction, and model selection. However, the paper could have been more informative about the specific techniques and models used in the implementation of the chatbot.

Habitamu Asimare (Habtamu, 2020) proposes the development of a chatbot to assist individuals with Ethiopian law-related queries. The chatbot is built using the Amharic language and utilizes an RNN model with ensemble architecture and rules integration, achieving an accuracy of 73.12%. However, the study acknowledges that this accuracy may be compromised by model overfitting, which cannot be solely resolved by changing data-splitting ratios. The article highlights several gaps that need to be addressed in order to successfully implement an adaptive bot model for legal consultation. The first gap pertains to accuracy and ensuring that the chatbot is able to provide precise legal advice in the Ethiopian legal context. Secondly, ethical and data security considerations need to be taken into account while using the chatbot. Thirdly, scalability is crucial in ensuring that the chatbot can provide accurate legal advice on a large scale. Finally, integrating the chatbot into existing legal service delivery systems will maximize its potential benefits.

Vorada socatiyanurak et al. (Vorada et al., 2021) presents an innovative and socially responsible chatbot called LAW-U, designed to provide confidential legal guidance and support to sexual violence victims and survivors. The authors collected data from multiple sources, including legal cases, legal consultation services, and academic studies, to train the chatbot. The authors used a hybrid approach, combining rule-based and machine learning-based techniques, to extract relevant features from the preprocessed text. They used pre-defined rules to extract legal concepts, case law, and statutes, and machine learning models to extract sentiment and emotional context from the text. The authors used the GloVe algorithm to represent words as vectors for further processing. They used a decision tree model to classify user input into different categories and a machine learning approach to enable the chatbot to learn from user interactions and improve its responses over time. The authors also used the cosine similarity metric to match user input with the appropriate pre-defined responses. The authors reported an average accuracy of 94.7%, which is impressive and demonstrates the effectiveness of the chatbot's design and implementation. While the article does an excellent job of discussing the design and implementation of the chatbot, there are some gaps in the evaluation of the model. The article does not provide sufficient information on how the model was tested, and whether it was evaluated in any way. Additionally, the article does not discuss how the model could be improved or extended in the future.

Segni Bedasa (Segni, 2021) presented a chatbot model aimed at improving maize productivity by providing farmers with relevant information and advice that entitled: *"Developing Afaan Oromo Text-based Chatbot for Enhancing Maize Productivity"*. The chatbot was designed and implemented based on various techniques and approaches, including data preprocessing, feature extraction, model selection, and reinforcement learning. The author also utilized the TF-IDF approach to extract relevant features from the preprocessed text. A support vector machine (SVM) was selected as the machine learning model, and reinforcement learning was used to improve the model's accuracy. The proposed chatbot achieved an accuracy of 93.3%, which is impressive given the complexity of the topic and the challenges associated with natural language processing in Afaan Oromo. The author collected data from various sources, including the Ethiopian Institute of Agricultural Research and the International Maize and Wheat Improvement Center. In this study, the researcher used retrieval-based approaches.

Fekadu Eshetu (Fekedu, 2021) presented a chatbot model that helps Ethio-Telecom customers with customer services in Afaan Oromo and Amharic languages. The author employed Bi-LSTM algorithms and reported 82.6% accuracy using L2 regularization technique to handle model over fitting. While this presented model is better suited to assist Ethio-Telecom customers with customer service, the researcher did not compare different optimizer algorithms to improve model performance. Although there are several optimizer algorithms, sometimes Stochastic Gradient Descent (SGD) outperforms Adam as demonstrated in a study by Vamsi et al. (2020). Additionally, the author did not utilize weight initializers in their model.

Summary of Related work

In general, many researchers have attempted to improve the performance of chatbots using the state-of-an-art approach to introduce a chatbot solution for all industries. The above-reviewed studies have been conducted on developing a chatbot for many different industries, including legal domain. However, the previously presented studies for legal domain are a kind of reterivel based which is not advanced when compared to generative chatbots. Consequently, this study aimed to develop a legal consultant chatbot for legal domain support using deep learning models. In addition to this, the proposed study intended to improve the performance of the existing chatbot model using different techniques such as selecting riched library deep learning models, using weight initialization techniques, and searching for the right network hyperparameter for the proposed model using hyperparameter tuning, etc.

Table 2.3: Summary of Related Work

No	***Titles of the papers***	***Authors and publish year***	***Algorithms with accuracy***	***Gaps***
1	Developing Afaan Oromo Text-based Chatbot for Enhancing Maize Productivity	Segni Bedasa	DNN (84.4%) and Seq2Seq (80.0%)	The author did not specify the threshold probability for candidate response. The researcher has not used the model overfit handling techniques. They used train test split ratios.
2	Designing and Developing Bilingual Chatbot for Assisting Ethio-Telecom Customers on Customer Services	Fekadu Eshetu	Bi-LSTM (82.6%)	The author did not compare the algorithms of different optimizers to achieve better results. The author did not use weight initializer techniques, which are used to adjust the initial weights correctly to ensure the model with better accuracy
3	Designing and Implementing Adaptive Bot Model to Consult Ethiopian Published Laws Using Ensemble Architecture with Rules Integrated	Habtamu Asimarie (2020)	RNN (73.12%)	Handling model overfitting by changing the number of data splitting ratios. Retrieval based
4	LAWBO: A Smart Lawyer Chatbot	Shubhashri G et al., (2018)	Dynamic Memory Network	Heavily dependent on the quality and size of the training dataset They used 90-10 train-test split Retrieval based

5	LAW-U: Legal Guidance Through Artificial Intelligence Chatbot for Sexual Violence Victims and Survivors	Vorada et al., (2021)	Decision Tree (94.7%)	• Combining rule-based and machine learning-based techniques (hybrid approach) used. • Retrieval based

CHAPTER THREE

3. RESEARCH METHODOLOGIES

3.1. Introduction

This chapter provides the methods, techniques, and procedures used in the design and development of a consultant chatbot, to achieve the main objective of this thesis work. The first part of the chapter describes the research design and approaches. Secondly, the data collection and the datasets used are described in detail beforehand along with their pre-processing. In the third part, the algorithms used to develop learning models are explained. Finally, the tools and techniques used with the evaluation methods used are presented.

3.2. Research Design and Procedure

Considering the nature of the problem and the research questions, this study falls under quantitative research, which involves conducting experiments to obtain results. The research design approaches used include data collection, data analysis, quantifying data, establishing relationships, and developing mathematical models to address phenomena or support theories (Chu, M., Jones, A., Clancy, A., & Donnelly, 2014)**.**

To achieve the research goals, various procedures have been employed. These procedures consist of a series of steps aimed at attaining the desired final output of the planned work. The procedures encompass identifying the research area, reviewing existing knowledge from different researchers, conducting a gap investigation, formulating research questions, selecting tools and methodologies, collecting and preparing the dataset, and ultimately designing and implementing the optimal proposed solution. The figure below provides an overview of the procedural flow in this thesis.

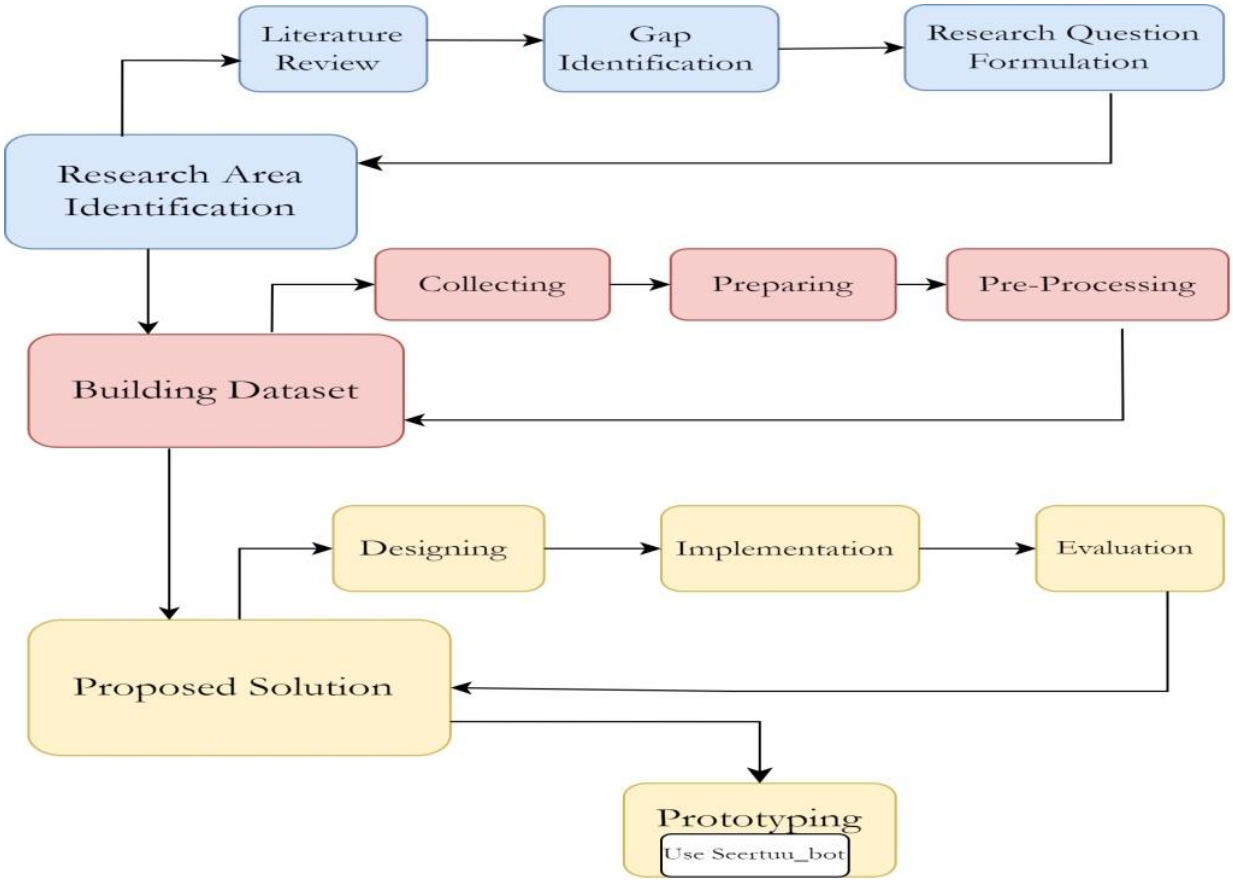


Figure* 3.1 *Research Design

3.3. Data Collection Method

Machine and Deep Learning algorithms are created or designed to develop models that learn from examples and experience. Those experiences are collections of observations or instances also known as dataset. In order to train the learning model, it is a must to have a good dataset. The accuracy of a deep learning model depends on several factors, the main one being the quality of the data set and the size of the data set. In this study, the following data collection methods were used to build the dataset.

a) Primary Data

Primary data is a type that is collected directly from main sources through interviews, surveys, experiments, etc. without going through any existing sources. Various data were collected from OSC information desk and customers through interviews. This data includes frequently asked questions by clients and other parties to obtain or improve legal understanding.

b) Secondary Data

Secondary data is data that has previously been collected by someone else but is for use by others. In this study, in addition to primary data, secondary data was collected from various social media such as Facebook, Telegram channels and OSC website. The collected dataset consists of questions about legal matters that were raised by court clients.

Generally, we have collected a dataset of above 2.1 MB, which includes both primary and secondary sources. The dataset, which consisted of frequently asked questions (FAQs) and their corresponding answers, served as a valuable resource for training the models and improving their performance in handling legal inquiries and delivering accurate responses.

Table 3.1: Initial distribution of dataset obtained

No.	Sources	Amount	Remark
1	Primary sources	400 KB	
2	Secondary sources	1700 KB	

3.4. Data preparation techniques

In the real world, raw data is often incomplete, inconsistent, and lacks specific characteristics or patterns. It is also more prone to errors. Therefore, once collected, the data needs to undergo a process of cleaning and preparation. To achieve this:

First, we have converted text from hardcopy to softcopy, which is collected through questionnaires and interviews.

Second, we have reorganized and adjusted the data collected from social sites to address any issues of incompleteness and inconsistency.

As described in table 3.1, we obtained 2.1 MB of data from various sources. However, after thorough cleaning and filtering, we ended up with 2 MB of usable data. This reduction in size is due to incomplete information, duplicated entries, and other data quality issues. Finally, we save the prepared data in plaintext format using Notepad++ for further analysis and processing. plain text format facilitated the application of various natural language processing techniques to preprocess and analyze the data.

Table 3.2: The amount of data filtered and prepared

No.	Type of dataset	Amount	Remark
1	Legal conversation dataset	2 MB (2000 KB)	

3.5. Data Preprocessing

Data pre-processing is an important step when developing a learning model to make the data more suitable for the model and it contributes to improving the dataset's quality. It involves several key steps. Because, the collected data may contain the noisy, misspelled words, formatting inconsistencies and irrelevant text. Various data preprocessing techniques were employed in the study, and some of them are described below.

Tokenizing is the process of breaking down a sentence into individual words or phrases. When working with textual datasets, tokenization is the most elementary thing performed on the text data. In-text preprocessing step, the patterns must be tokenized into tokens using spaces. For instance, "*Tajaajila gorsaa seeraa bilisaa hawaasaaf ni keennitu*?"; is defined as "*Do you provide free legal consultancy services to the community*?" This sentence is tokenized as *'Tajaajila', 'gorsaa', 'seeraa', 'bilisaa', 'hawaasaaf", 'ni', 'keennitu', '?'*.

Text Normalization: Is the process of transforming words into a standard form for input to a model. For instance, users may write the same word differently, and the machine understands those words differently. Such problem can be solved by text normalization, which transforms similar words written in different formats into one unique word. Examples of text normalization include expanding abbreviations, converting slang words to their formal equivalents and replacing contractions with their full forms. To normalize the data, we can also apply the lowercasing technique to convert all the text to lower case. In addition to this, a regular expression is used to normalize words that have similar meanings but are written in different formats. For instance, the word `haga/hanga, erga/ega, baayee/baayy'ee, osoo/otoo, mini/miti`, etc., have identical meanings in the Afaan Oromo language, they have normalized into one single word.

Stop words removal: - Eliminating stop words is a crucial process in natural language processing that involves removing frequently used and redundant words like `"fi/and",`

"kana/this", "Akka/like", "nuti/we", "yookiin/or", "amma/now", "inni/he", "kanaaf/so", "siif/for", "irra/on", "sun/that", "isa/he", "kee/yours", *etc.,*. As not all terms found in a dataset are weighted equally in designing a chatbot model, removing irrelevant stop words is necessary to map the user's queries to the right answers. This detachment of words greatly contributes to reducing the learning time of the model.

Data cleaning- is used to removing incomplete or incorrect data, standardizing text, and correcting typos. It involves filtering out any irrelevant or incorrect data and eliminating noise data from the dataset to make it more understandable by the model, such as removing punctuation marks, white spaces, removing extra symbols, etc. This step is essential for a reliable and accurate model.

3.6. Feature Extraction Methods

Feature extraction is a crucial step in machine learning that involves the extraction of meaningful attributes or features from raw data points or samples. These features are then utilized to train a model, which could ultimately predict new outcomes. Additionally, feature extraction techniques might also aid with decreasing the dimensionality of the data, producing higher-performing models. While building a chatbot and utilizing deep learning methods, there are various feature extraction techniques that one could use, like (ELMO). The next section discussed the types of feature extraction approaches.

3.6.1. Embeddings from language Models Representation (ELMo)

Embeddings from Language Models, also known as ELMo, is a neural language representation model that creates deep, context-rich word representations. These representations are used as pre-trained features for various natural language processing tasks. Unlike traditional word embedding's, such as Glove, Bag of Words and Word2Vec that only capture one fixed meaning of a given word, ignores the context of words and the word order and structure, ELMo highlights the multiple meanings of a word by capturing its context-dependent attributes.

The advantage of ELMo is that it significantly improves the performance of a wide range of natural language processing tasks such as question answering, sentiment analysis, and language inference. Moreover, it can be used to generate pre-trained embeddings for general-purpose, large-scale NLP models. This makes it particularly useful for those without vast computational

resources, as the pre-trained ELMo models can be downloaded and used without the need for expensive computing resources.

One of the strengths of ELMo lies in its ability to develop context-dependent word representations. The pre-trained deep neural network can process the text in both forward and backward directions, allowing it to represent words within their broader context, including sentence structures, varied meanings, and complex nuances. Additionally, ELMo can be incorporated into different models via a simple concatenation approach that can further improve the accuracy of models by adding more contextual information.

3.7. Model Building Algorithms

The legal consultant chatbot in this study focuses on intent recognition, which involves understanding the user's intent behind their query rather than categorizing it into predefined classes. This approach allows the chatbot to identify the purpose or objective of the user's question and generate appropriate responses. To achieve this, supervised machine learning and deep learning algorithms are commonly used. In this research, deep learning algorithms were chosen as they have the ability to extract intricate features from the data, resulting in improved performance compared to traditional machine learning algorithms.

3.8. Model Selection Techniques

Model selection is the process of choosing a model that best fits to the selected task. There have been different deep learning models proposed for various areas. However, according to Brownlee's (Brownlee, 2020) description, there is no straightforward method for identifying the best model for a given problem type. Instead, it is recommended that an expert conducts-controlled experiments to determine which model and model configuration performs best. Based on the size of collected dataset and the performance of models, the following models have been selected.

3.8.1. Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) that is commonly used in chatbots to enable the model to remember and interpret previous conversation history. In a chatbot context, LSTM can help to maintain the context of a conversation and generate more natural-sounding responses.

LSTM models work by processing sequences of input data over time, using a series of memory cells and gates to selectively store and forget information. The memory cells allow the model to remember information from earlier in the conversation, while the gates control the flow of information through the model. LSTM models can be trained on large datasets of conversation histories, allowing them to learn patterns and relationships between user queries and responses. They can also be combined with other natural language processing techniques, such as named entity recognition (NER) or sentiment analysis, to generate more accurate and relevant responses.

LSTM is a more sophisticated recurrent neural network type with a better memory capacity. The model involves three gates; the input gate, forget gate, and the output gate. The sigmoid function generates these three gates. The forget gate *Ft* controls whether the previous timestamp information is flushed out or retained in the cell state. The value in the cell state is determined by the input gate layer whether it will be retained or updated. The forget gate Ft, input gate It, and output gate Ot layer values at hidden layer ht are computed by looking at ht-1 and Xt: (Graves, A., Mohamed, A., & Hinton, 2013)

$$\text{Input Gate: } i_t = \sigma(X_t * U_i * + H_{t-1} * W_i \quad (3.1)$$

$$\text{Forget Gate: } f_t = \sigma(X_t * U_f * + H_{t-1} * W_f \quad (3.2)$$

$$\text{Output Gate: } o_t = \sigma(X_t * U_o * + H_{t-1} * W_o \quad (3.3)$$

Where *Xt* is the current timestamps input, Uf is the input values weight, ht-1 is the previous timestamp of the hidden state, b is the bias, W is the hidden states weight matrix, Ct is a vector of new candidate values, and *Nt* is New information.

Finally, the output ot is calculated as - $= \delta(* + h - 1 * +)$, and then gets the final hidden state of timestamp t as - $= * h()$. This shows how the three gates of the LSTM algorithm work together to produce h hidden state at time t. Fig.3.2. depicts the general architecture of the LSTM model.

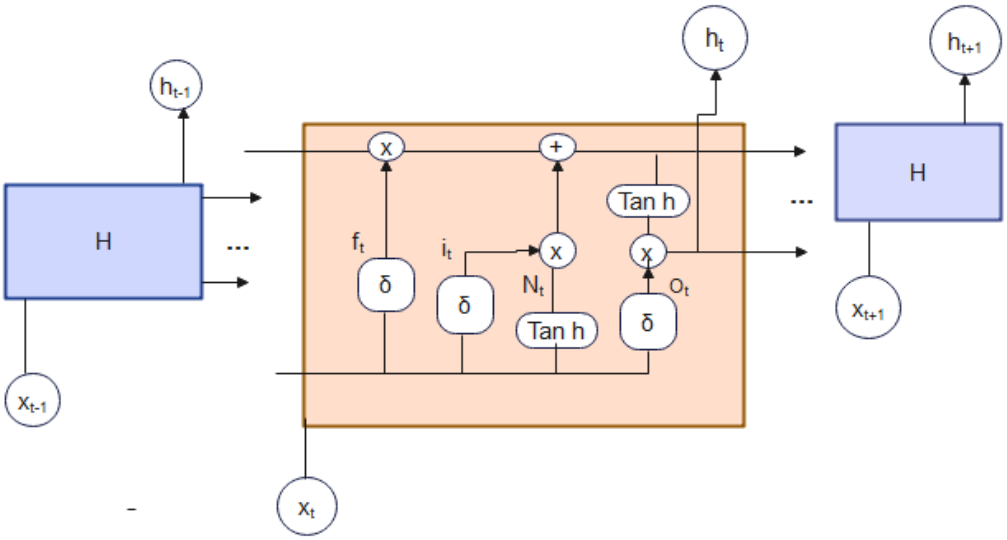


Figure 3.2: ***The LSTM Model's architecture***

Generally, LSTM is a powerful tool for building chatbots that can understand and respond to complex user queries, and can help to improve the user experience by generating more natural and engaging conversations.

3.8.2. Generative Pre-training Transformer (GPT)

Generative Pre-training Transformer (GPT) is a commonly used model for building chatbots due to its advanced Natural Language Processing (NLP) capabilities. The model selection techniques for GPT in chatbot development are crucial because it is a pre-trained model, and fine-tuning is necessary. The first step is to identify the task and the dataset, followed by fine-tuning GPT on a specific dataset. The next step is to evaluate the fine-tuned GPT model to determine its effectiveness in solving the task. Fine-tuning allows the chatbot to recognize commonly used phrases, terms, and sentence structures, and generate responses that seem more natural and human-like. One of the significant advantages of GPT is its ability to understand the context of a conversation by analyzing previous messages and generating a response based on the conversation's flow. This feature makes the chatbot more intuitive and engaging for users to interact with**. (Brady D. Lund, 2023).**

GPT works by learning from a vast amount of text data and processing it through deep layers of neural networks, allowing it to generate text similar to what it has learned. The model uses unsupervised learning, where it trains on a large corpus of text data without human supervision. It then generates the most likely sequence of words in response to the

input it receives. GPT can also contextualize the responses by tracking the user's previous input, making it seem more natural and human-like.

3.9. Model Over fit Handling Techniques

Over fitting is a common problem in deep learning, where the model becomes too complex and starts to memorize the training data instead of learning the underlying patterns. This can result in poor performance on new data and reduced generalization capabilities. To address this issue, there are several techniques for handling overfitting in deep learning models. Like Regularization, dropout, early stopping, data augmentation and ensemble methods are popular model overfitting techniques (Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, 2014). In this study, regularization techniques have been selected. This helps to prevent the model from becoming too complex and overfitting to the training data.

3.10. Weight Initialization Techniques

In deep learning algorithms, appropriately adjusting the initial weights is crucial to ensure better accuracy in the model. Prior to training the models with the training dataset, weight initialization involves assigning small initial values to the parameters of the neural network model. This process helps prevent issues such as the vanishing or exploding gradients in neural network models during backpropagation. Various weight initialization techniques exist, including Zero Start, Randomly Start, and Xavier (Glorot Start), among others. In our approach, we employed Xavier (Glorot) weight initialization, which has demonstrated superior performance in neural network models (Boulila et al., 2022).

3.11. Hyper parameter Tuning Methods

When dealing with neural network models, it is essential to fine-tune the three layers of the models using appropriate hyperparameters. The variation in hyperparameters within the same neural network plays a significant role in differentiating the accuracy levels among models. Achieving the best hyperparameter configuration is crucial for enhancing the performance of the model. However, identifying the optimal hyperparameters for the neural network model involves careful attention and experimentation. Parameters such as the number of iterations (epochs), learning rate, optimization algorithms, neurons in each layer, and batch size all need to be tuned effectively. This process can be challenging and requires an iterative and experimental approach. Two commonly used hyperparameter tuning algorithms, namely grid search and random search, are employed to find the optimal hyperparameter values for the neural network (Nabi, 2019)**.**

3.12. Model Evaluation Techniques

The evaluation of model performance is an important aspect of the learning process, which involves the use of various techniques to quantify and assess the effectiveness of the model (Géron, 2019). In this study, the performance of the proposed model was evaluated using techniques such as cross-validation, as well as metrics such as accuracy and loss. It is important to note that selecting a model with high accuracy on the training dataset alone does not necessarily indicate that the model is good, as its performance on unseen data must also be considered. Therefore, additional evaluation techniques were employed to ensure that the proposed model performed well on new, unseen data.

3.12.1. Cross-Validation

Cross-validation is a widely used technique to assess the performance of machine learning models by dividing the dataset into k equal partitions and iteratively using them for both training and testing (Kohavi, 1995). This method is particularly useful when the amount of data available is limited, as it enables the evaluation of a model's generalization capability while avoiding overfitting. By dividing the data into multiple folds, each subset is used for testing the model once, while the remaining subsets are used for training. This process is repeated for each fold, and the model's performance is evaluated based on the average performance across all iterations.

- ***Kfold CrossValidation***

K-fold cross-validation is a popular technique for evaluating the performance of machine learning models. It involves randomly dividing the dataset into K equal parts, where each part is used for testing the model while the remaining K-1 parts are used for training (Brownlee, 2018). However, because the division is random, there is a possibility that the folds may be unbalanced, which can adversely affect the performance of the model or bias the training process. This can be a particular concern when dealing with datasets of varying sizes or with imbalanced class distributions (Team, 2020). Therefore, it is important to carefully consider the division of data and balance the folds to ensure the best possible performance of the model.

- ***Stratified K-fold Cross-Validation***

Stratified k-fold cross-validation is a modified version of the K-fold cross-validation technique used for evaluating the performance of machine learning models. This method ensures that each fold of the dataset contains an equal number of samples from each class, thus maintaining the class distribution across all folds in a stratified manner

(Sharma, 2022). By distributing the data from all classes across all folds, this technique helps to minimize both bias and variance, particularly in cases where the dataset is imbalanced.

In this study, the stratified k-fold cross-validation technique was employed to ensure that each fold of the dataset was representative of the complete data, as well as to mitigate any potential biases that may arise from the imbalanced nature of the dataset. By maintaining the class distribution across all folds, this technique helps to improve the accuracy and generalization performance of the machine learning model.

3.12.2. Evaluation Metrics

Accuracy: is a widely used metric to evaluate the performance of machine learning models across different industries. It indicates the percentage of test data that is correctly classified by the model. This metric is particularly useful when all classes in the dataset are equally important. To calculate accuracy, the number of instances that are correctly classified is divided by the total number of classifications. The formula for computing accuracy is provided in Equation 3.4.

$$\text{Accuracy} = \frac{TN+TP}{TP+FP+FN+FP} \qquad (3.4)$$

Precision and Recall: Precision is a metric that measures the proportion of true positive cases among all the cases predicted as positive by the model. It focuses on the accuracy of positive predictions. On the other hand, recall, also known as sensitivity, indicates the proportion of actual positive cases that were correctly identified by the model. It measures the model's ability to capture all positive instances. The formulas for precision and recall are provided in Equations 3.5 and 3.6, respectively (Purva Huilgol, 2020)**.**

$$\text{Precision} = \frac{TP}{TP+FP} \qquad (3.5)$$

$$\text{Recall} = \frac{TP}{TP+FN} \qquad (3.6)$$

F1-score: is a metric that combines the concepts of precision and recall by calculating their harmonic mean. It provides a balanced measure of the model's performance by considering both precision and recall simultaneously. Mathematically:

$$\text{F1} - \text{score} = 2x\frac{PrecisionxRecall}{Precision+Recall} \qquad (3.7)$$

Where:

False Positive (FP) = when predicted positive and actual is negative.

False Negative (FN) = when predicted negative and actual is positive.

True Positive (TP) = when predicted positive and actual is positive.

True Negative (TN) = when predicted negative and actual is negative.

Loss: is a metric used to assess the performance of a model by measuring its training and validation errors. The training loss quantifies how well the model fits the training dataset, while the validation loss evaluates the model's performance on a separate validation dataset. These losses are calculated by summing up the errors for each instance in the respective datasets. In this work, the loss metric was utilized to determine if the model requires further adjustments or fine-tuning. By visualizing the training and validation losses on a graph, it provides insights into how well the model is fitting the data and helps identify any potential issues or areas for improvement. Monitoring the loss metric enables researchers to optimize their models and make informed decisions during the training process.

Perplexity: is a metric commonly used to evaluate the performance of language models, particularly in natural language processing tasks such as text generation or language translation. It measures how well a language model predicts or assigns probabilities to a sequence of words. Perplexity is calculated based on the concept of entropy, which measures the average uncertainty or unpredictability of the next word in a sequence given the preceding words. A lower perplexity value indicates that the language model is more certain and accurate in its predictions. Mathematically:

$$\text{Perplexity} = exp(\text{Cross} - \text{Entropy}) \qquad (3.8)$$

Where Cross-Entropy is defined as:

$$\text{Cross} - \text{Entropy} = -(1 / \text{N}) * \text{sum}(\log(\text{P}(\text{w\_i}))) \qquad (3.9)$$

3.13. Design and Development Tools

Design and development tools play a crucial role in facilitating documentation, implementation, and prototyping processes. These tools are considered essential as they simplify and streamline various tasks, making them more accessible and convenient.

3.13.1. Design Tool

Draw.io[2] is design software that enables users to generate customized diagrams and flowcharts. The software offers a wide range of shapes, along with a user-friendly drag-and-drop feature and a variety of tools to aid in the design process. Additionally, it provides the convenience of storing diagrams on either a local device or in the cloud.

3.13.2. Development Tools

Google Drive[3]: is a cloud-based file storage service provided by Google that enables users to store and access their files from any device with synchronization features.

Keras[4]: is a high-level modular deep learning framework, written in Python that operates on top of Theano or TensorFlow libraries, aimed at making the implementation of deep learning models easier for developers.

Librosa[5]: is a powerful music and audio analysis tool that is primarily used for music information retrieval (MRI). It is a Python package that helps in audio and music processing.

Tensor Flow[6]: is a numerical computation library developed by the Google team. It uses a graph to implement machine learning algorithms where tensor (multi-dimensional arrays) is used as an edge and mathematical functions as nodes. It allows the deployment and execution of machine learning algorithms on CPU, GPU, and cluster.

Colab[7]***:*** short for Colaboratory, is an open-source environment for Jupyter Notebook that runs completely on the cloud. It requires no setup and comes with many pre-installed Python packages and libraries. The environment provides both CPU and GPU runtime for machine learning tasks.

3.13.3. Data Preparation Tools

Notepad++[8]***:*** it was utilized as a free source code editor in this work to prepare an Afaan Oromo text dataset. It served as a straightforward tool for dataset preparation and allowed

[2] https://www.app.diagrams.net
[3] https://www.google.com/drive
[4] Keras.io
[5] https://librosa.org
[6] https://www.tensorflow.org
[7] Colab.research.google.com

for the saving of data in plain text format. The dataset was prepared using the Notepad++ code editor and stored in plain text form.

3.13.4. Deployment Tools

Flask[9]: Once the experiments are completed, the top-performing model is deployed on a web server, enabling it to generate authentic responses based on the provided story for verification purposes. To achieve this functionality, the Flask micro web framework was employed. The decision to use Flask was driven by the fact that deep learning models were being deployed, making it a suitable choice for implementing the web server functionality.

CHAPTER FOUR

4. PROPOSED SOLUTION FOR LEGAL CONSULTANCY CHATBOT SERVICE

4.1. Introduction

This chapter presents the design and overview of the proposed solution, which aims to address the challenges and questions raised in chapter one and address the gaps identified in the literature review. The proposed model architecture illustrates how a legal consultancy service chatbot model could be designed and developed using various deep learning algorithms and natural language processing (NLP) techniques. Additionally, the chapter discusses the way used for preparing datasets and evaluating the performance of the model.

4.2. Proposed Model Architecture

The main aim of this research is to create generative legal consultant chatbot model that enhances the understanding of legal concepts. To achieve this, a dataset was collected from various sources such as OSC, social media, individual customers, and online websites. The collected data was manually filtered and prepared in a suitable format for the deep learning model, as discussed in section 3.4. The dataset underwent several text preprocessing techniques using NLTK libraries, such as tokenization, stop word removal, text normalization, and elimination of irrelevant content, before being used for the model. It is important to note that the collected dataset was not directly utilized in the model. Therefore, to achieve these tasks, we divided the process into different phases.

Phase 1: prepare dataset, the collected data from various sources was prepared and underwent data preprocessing techniques, including tokenization, to convert sentences into a suitable form for the deep learning algorithm. This was necessary because deep learning models cannot process textual data directly and require representation using vectors, which are generated using word embedding techniques. These techniques will be discussed in detail in the upcoming section.

Phase 2: Feature Extraction, feature extraction is important for converting raw data into a computer-understandable form. In this study, ELMO feature extraction has been proposed to convert tokens into vectors. Once the tokens are converted to vectors, LSTM and GPT deep learning algorithms will be applied to learn patterns from the vectors.

Phase 3: Model selection, GPT and LSTM have been selected based on the criteria mentioned in the chapter 3 section 3.8.

Phase 4: Select best performed model, based on the achieved accuracy, select the model with the higher accuracy.

Phase 5: Prototype, develop prototype for best performed model.

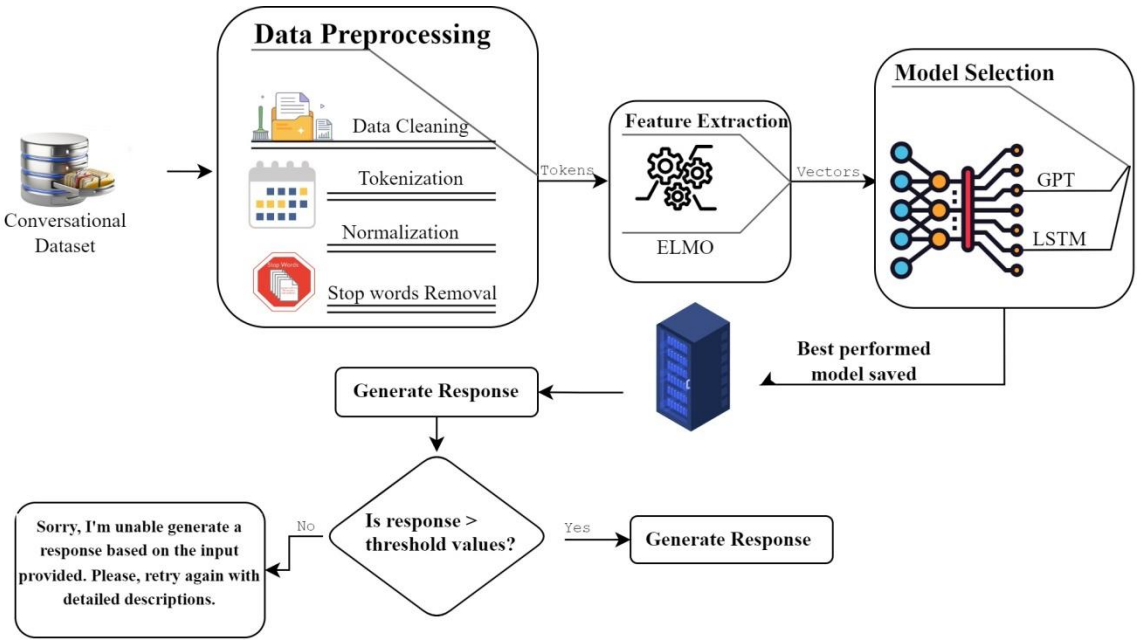


Figure 4.1: Legal consultant service chatbot model architecture

4.2.1. Conversational Datasets

This study aims to develop a legal consultant generative based chatbot that utilizes deep-learning techniques to generate responses. To achieve this, a new dataset for legal consultant chatbots in Afan Oromo language needs to be created since there are currently no published or existing datasets for legal consultant chatbots in Ethiopia or in OSC. Building this dataset for the legal consultant chatbot involves collecting data from OSC clients and various social media sources, and then preparing the dataset in a suitable format for model building. Thus, we have created a legal consultant dataset by collecting frequently asked questions from OSC clients and different social media sources, all of which were written in Afaan Oromo language.

4.2.2. Proposed Data Preprocessing

Training deep learning models can be a daunting task, especially when dealing with messy and inconsistent data. In this study, the collected data presented a challenge due to its quality, necessitating a thorough cleaning and normalization process. This crucial

preprocessing phase was carried out to prepare the Afaan Oromo conversational queries for training and testing the legal consultant service model. In this phase, we delved into the details of how the data preprocessing would be done, highlighting specific steps and methods for ensuring the data is well-prepared for use in the model. When creating a deep learning model, it's common to encounter messy and incomplete data from real-world sources. This requires working with the data and organizing it in a way that aligns with the specific use case of the study. To achieve this, several data preprocessing techniques were used in the study, such as removing irrelevant items and punctuation, getting rid of stop words, normalizing the text, and tokenizing the data.

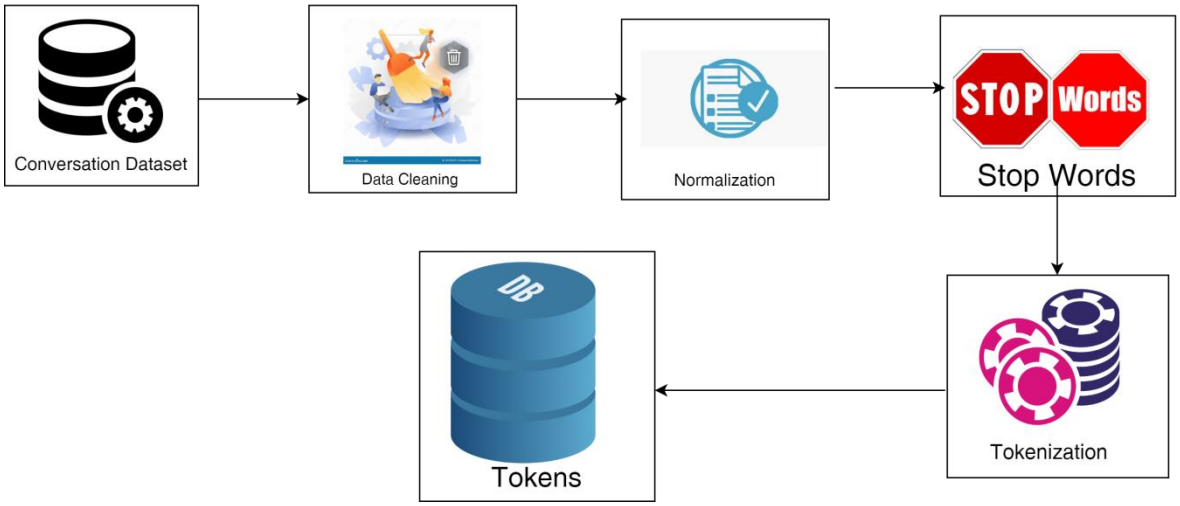


Figure 4.2: Proposed data preprocessing technique

To recall, Section 3.4.2 listed various pre-processing techniques used to make the data appropriate for prediction. This section provides a detailed explanation of how these pre-processing techniques are implemented.

4.2.2.1. Data Cleaning

Data cleaning is the process of removing incomplete or incorrect data, standardizing text, and correcting typos. In the Afaan Oromo language, it is crucial to note that the language uses punctuation marks similar to those used in English. However, these punctuation marks may not always be relevant for the specific purpose of model training. Thus, it is advisable to remove them from the dataset to ensure effective model training. The process involves filtering out any irrelevant or incorrect data, eliminating noisy data from the dataset, and making it more understandable by the model. This step includes removing punctuation marks, white spaces, extra symbols, etc., which do not significantly contribute but rather

increase the learning time of the model. Data cleaning is essential to ensure a reliable and accurate model.

4.2.2.2. Tokenization

After completing various preprocessing techniques to clean the dataset, the next step is to tokenize the text data. Tokenization involves breaking down the text into smaller units or tokens, while also removing certain special characters such as apostrophes, commas, and periods, and often normalizing the text to lowercase (Allahyari et al., 2017)**.** Tokens can consist of either single words or N-grams, which are N consecutive words combined into a single token (Abinash Tripathy and Ankit Agrawal and Santanu Kumar Rath, 2016)**.** This approach preserves the order of words and the information it conveys. For instance, in Afaan Oromoo, the sentence *"Tajaajila gorsaa seeraa bilisaa hawaasaaf ni keennitu?"* is defined as "Do you provide free legal consultancy services to the community?" This sentence is tokenized as *'Tajaajila', 'gorsaa', 'seeraa', 'bilisaa', 'hawaasaaf', 'ni', 'keennitu', '?'*. Once the text has been preprocessed and tokenized, it is ready for model training.

4.2.2.3. Normalization

Normalization is a critical step in obtaining clean and structured data from unstructured text. There are two types of normalization, namely word-level and character-level normalization. Character-level normalization is used to address different characters that may have the same sound but are written in different forms. On the other hand, word-level normalization is used to address words that have different written forms but the same meaning. In this study, we utilized word-level normalization techniques. Appendix A provides a list of the remaining normalized words after this process. Examples of words that were normalized include *"dhakaa" and "dhagaa"*, as well as *"ishii" and "ishee",* which have the same meaning but different written forms.

4.2.2.4. Stop words Removal

After cleaning and tokenizing the dataset, it is important to remove stop words to enhance the performance of machine learning algorithms. Stop words are common words that do not contribute significantly to the meaning of the text and can be filtered out before vectorizing the data. In this study, we collected a list of common stop words for Afaan Oromoo language from a reliable source **(Tesfaye, 2010)** and added them to the NLTK stop words corpus for filtering out common and irrelevant words from the dataset. The remaining stop words for Afaan Oromoo language are listed in Appendix B. Pronouns,

prepositions, conjunctions, articles, and particles are some of the most frequently occurring terms in any language and are usually classified as stop words. Removing these words from the dataset can significantly reduce the learning time of the model and improve its performance. However, the decision to remove stop words depends on the specific goal of the analysis or task at hand. In summary, incorporating stop-word removal into the text preprocessing pipeline can be an effective way to optimize the performance of machine learning models on text data.

4.2.3. Feature Extraction

Text feature extraction is a crucial step in processing textual data for deep learning algorithms. This process involves converting a list of words into vectors, which can then be used for feature dimension reduction and developing high-performance models. In the next section, we will discuss the main approach to feature extraction. Since deep learning algorithms cannot directly work with raw text, feature extraction techniques are necessary to convert text data to vectors. In this study, we propose one feature extraction method: Embedding's from Language Models Representation (ELMO). This method takes tokenized data sets and converts them into vectors using feature extractions, as illustrated in Figure 4.3. By utilizing these techniques, we can effectively transform the text data into a format that can be used for training machine learning models.

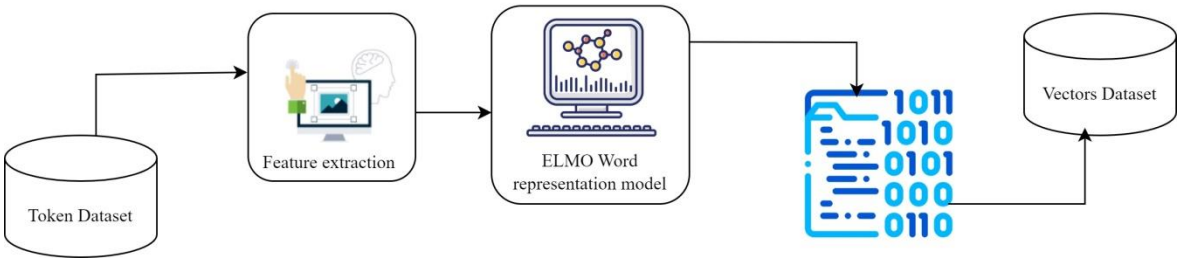


Figure 4.3. ELMO feature extraction

As described in chapter three section 3.6, ELMO is a word representation model that uses bidirectional language modeling to create contextualized embeddings for each word in a sentence. It employs self-attention to weigh the importance of each word and generates a set of features that capture the text's meaning and context. In summary, ELMO's feature extraction involves creating a set of contextualized embeddings for each word in a sentence and weighing their importance using self-attention, to represent the input text as a set of features that capture its meaning and context.

4.2.4. Model Building Algorithms

The objective of this study is to build a consultant legal service chatbot using a deep learning algorithm on categorized and labeled datasets. The study proposes two model, the GPT Model and LSTM model, which have been chosen based on specific criteria detailed in section 3.8. The first model is the GPT Model, which is designed to handle large amounts of text data and has been shown to achieve state-of-the-art results in several natural language processing tasks. The second model is the LSTM Model, a type of recurrent neural network that can capture temporal dependencies in sequential data. The study experimented with the model using the prepared dataset and evaluated them under cross-validation to identify the most suitable model for designing a legal consultant service chatbot, as shown in 4.4.

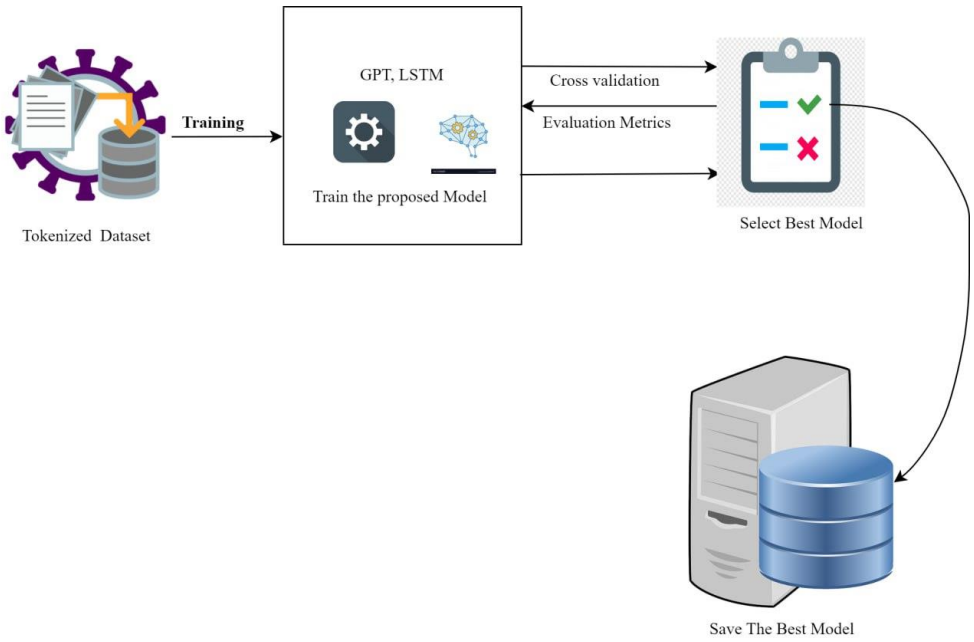


Figure 4.4. *Proposed Model Building Algorithms*

4.2.4.1. Generative Pre-training Transformer (GPT)

GPT is a generative language model that is trained on a large corpus of text data and can generate coherent and grammatically correct text based on a given prompt or context. It uses transformer architecture with self-attention mechanisms to learn the dependencies between different parts of the input sequence and generate output that is contextually relevant and meaningful. The GPT Model uses self-attention mechanisms to attend to different parts of the input sequence while generating output, as described by (Radford et al., 2018). GPT model has been trained on categorized and labeled datasets to provide accurate and reliable legal advice to users of the chatbot service.

How Generative Pre-training Transformer works: is a deep learning algorithm used to build a legal consultant service chatbot. It is designed to handle large amounts of text data and has been shown to achieve state-of-the-art results in several natural language processing tasks. The way GPT works is by using a pre-training step, where the algorithm is trained on a large corpus of text data, in this case legal documents, to learn the patterns and relationships between words and phrases. This pre-training step enables the model to understand the context and meaning of legal terms and phrases.

1. ***Pre-training:***

Architecture: GPT uses a stack of self-attention layers known as the Transformer. The Transformer architecture enables efficient processing of input sequences and capturing contextual dependencies.

Pre-training objective: GPT is pre-trained on a large corpus of unlabeled text data using a language modeling objective. The model is trained to predict the next word in a sequence given the preceding context. This process helps the model learn the statistical patterns and relationships in the language.

Self-attention mechanism: Self-attention allows the model to capture dependencies between different words in a sentence by assigning weights to the words based on their relevance to each other. This enables the model to effectively model long-range dependencies and understand the context of each word.

Positional encoding: GPT incorporates positional encoding to inform the model about the order of words in a sequence, allowing it to consider the sequential nature of the text.

2. ***Fine-tuning:***

Task-specific data: After pre-training, GPT is fine-tuned on a smaller dataset that is specific to a particular task, such as text classification or text generation. The fine-tuning process helps adapt the pre-trained model to the specific task and improves its performance.

Transfer learning: The pre-trained GPT model serves as a starting point, capturing general language patterns and knowledge. Fine-tuning allows the model to specialize for the target task while retaining the learned representations from pre-training.

Training process: During fine-tuning, the model's parameters are updated using the labeled data from the task-specific dataset. The model is optimized to minimize the loss specific to the task, such as cross-entropy loss for classification tasks. Once the fine-tuning is complete, the GPT model can be used to generate text, answer questions, classify text,

summarize documents, and perform various other natural language processing tasks based on the specific fine-tuned objective.

Overall, the GPT algorithm is an effective tool for building a legal consultant service chatbot that can provide accurate and reliable legal Consultant to communities with little knowledge of the law.

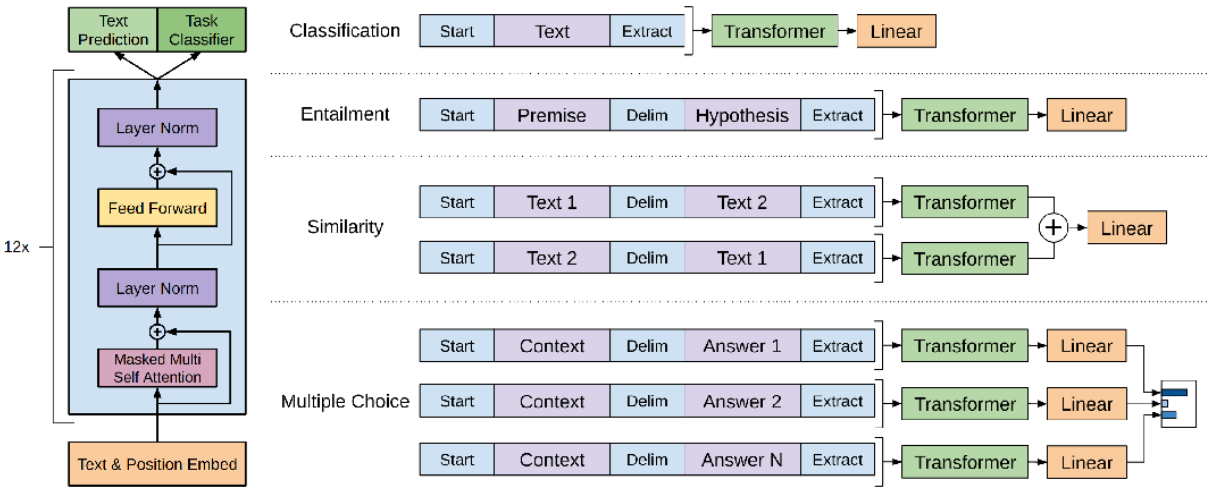


Figure 4.5: GPT algorithm

4.2.4.2. Long Short-Term Memory (LSTM)

The **LSTM** model is a type of recurrent neural network that can generate coherent and grammatically correct text based on a given prompt or context. It is trained on a large corpus of text data and uses recurrent connections and memory cells to learn dependencies between different parts of the input sequence. The LSTM model attends to different parts of the input sequence using its internal memory state, allowing it to generate output that is contextually relevant and meaningful. The model has been fine-tuned on categorized and labeled legal datasets to provide accurate and reliable legal advice to users of the chatbot service.

***How Long Short-Term Memory* works:** The Long Short-Term Memory (LSTM) model is a type of recurrent neural network used to build a legal consultant service chatbot. It is designed to handle large amounts of text data and has been shown to achieve state-of-the-art results in several natural language processing tasks. The LSTM model works by using a recurrent connection and memory cells to learn the patterns and relationships between words and phrases. This allows the model to understand the context and meaning of legal terms and phrases. The model is trained on a large corpus of text data, including legal documents, to pre-train its parameters and enable it to recognize and generate coherent

legal advice. Once the pre-training step is complete, the model is fine-tuned on a smaller dataset of categorized and labeled legal documents to improve its ability to generate accurate legal advice for users of the chatbot service. During this fine-tuning process, the model is trained to generate responses to legal questions by attending to different parts of the input sequence using its internal memory state, which allows it to recognize and incorporate relevant context into its generated responses.

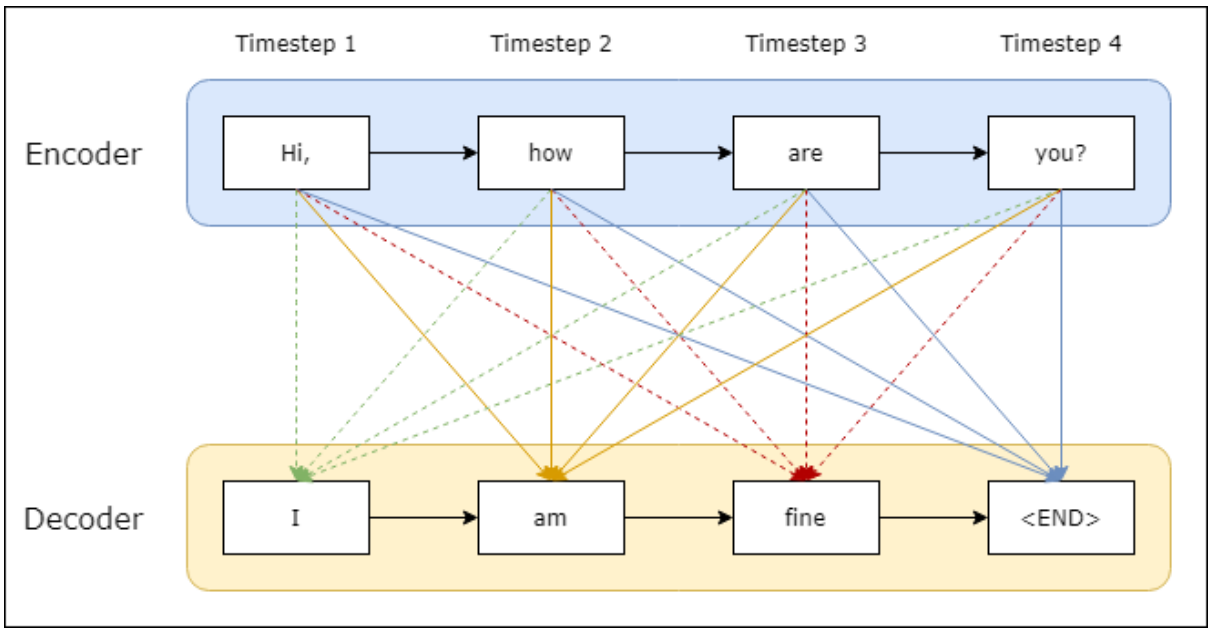


Figure 4.6: Encoder-decoder model

The encoder outputs a final state vector (memory) which becomes the initial state for the decoder. We use a method called *teacher forcing* to train the decoder which enables it to predict the following words in a target sequence given in the previous words. As shown above, states are passed through the encoder to each layer of the decoder. *'Hi,', 'how', 'are', and 'you'* are called input tokens while *'I', 'am', and 'fine'* are called target tokens. The likelihood of token 'am' depends on the previous words and the encoder states.

4.2.5. Hyper-parameter Tuning

Best optimizing hyper-parameters is a crucial step in achieving high performance for deep learning and generative-based models. It involves selecting the best set of hyper-parameters that allow the model to perform optimally. Hyper-parameter tuning can be done manually by trying out different combinations of hyper-parameters, but this process can be time-consuming and may not be feasible when dealing with a large number of hyper-parameters. Therefore, an automated approach is often preferred, where an algorithm is used to search for the best set of hyper-parameters. Two popular methods for automated

hyper-parameter tuning are grid search and random search. Grid search involves testing all possible combinations of hyper-parameters from a defined range, while random search samples hyper-parameters randomly from the defined range. In this study, we have utilized an automated approach for hyper-parameter tuning, specifically the random search algorithm. This allows us to efficiently explore a large hyper-parameter space and find the best set of hyper-parameters for our deep learning and generative-based model.

4.2.6. Proposed Model Evaluation

When designing a generative deep learning model, it is crucial to evaluate its generalization capability and ensure that it can produce diverse and novel outputs. To achieve this, various evaluation techniques such as cross-validation and evaluation metrics, including accuracy, precision, recall, confusion matrix, and f1-score, can be used, as discussed in chapter 3, section 3.12. In addition to these techniques, human evaluation by experts can also be performed to assess the model's performance. In this study, we employed Stratified K-fold cross-validation to minimize bias and variance in the generative deep learning model (Sharma, 2022). The dataset was divided into k equal-sized folds, with each fold being used as a validation set and the remaining k-1 folds used as a training set. The recommended value for k is either 5 or 10 (Kant, 2021), as a larger k value would result in higher running time and computing costs. To summarize, when designing a generative deep learning model, it is crucial to evaluate its generalization capability and ability to produce diverse and novel outputs using various evaluation techniques such as cross-validation and evaluation metrics. Human evaluation by experts can also be useful. Stratified K-fold cross-validation can be used to evaluate the model's performance and minimize bias and variance. The value of k should be chosen carefully to balance computational resources and evaluation accuracy.

4.3. Proposed Model Prototype Architecture

Based on the proposed system architecture, a prototype for a legal consultant service chatbot using deep learning generative-based models was designed. To develop the prototype, a different tool was used as discussed in the methodology phase. The model was trained and validated on each sample of cross-validation, and the best-performing classifier was deployed on the web server.

The chatbot can be accessed by the user via HTTP request. The flask server receives the user's request and submits it to the model. The model then generates a response based on the user's request, and the response is sent back to the user through the flask server. It's

important to note that the legal consultant service chatbot is designed to provide legal advice to users. However, the advice provided by the chatbot is not a substitute for professional legal advice. Users should seek the advice of a licensed attorney or legal professional for any legal issues they may have. Additionally, the chatbot's responses should not be construed as creating an attorney-client relationship between the user and the chatbot or any affiliated parties.

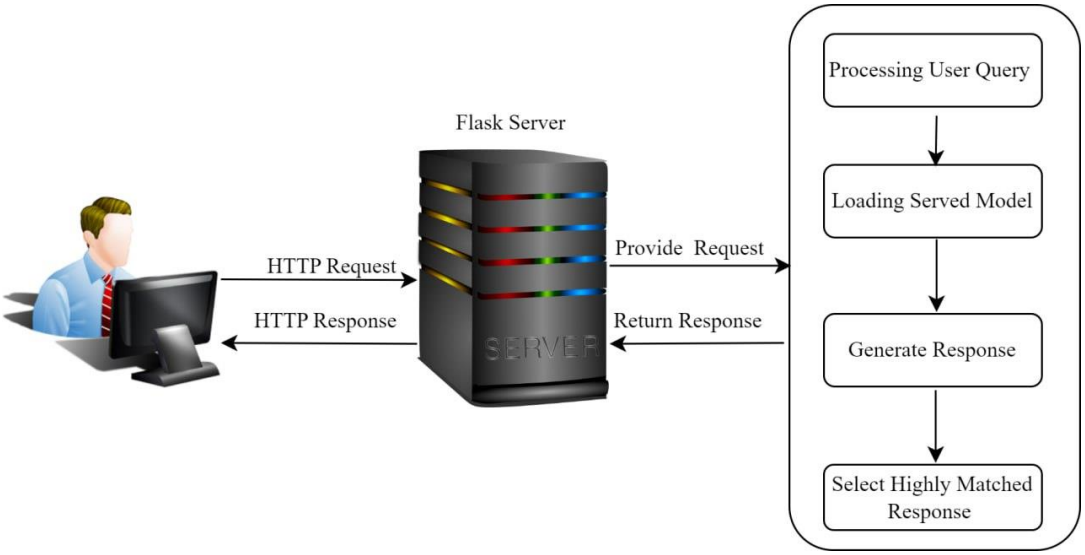


Figure 4.7: Prototype of legal consultancy service chatbot

CHAPTER FIVE

5. EXPERIMENTATION AND IMPLEMENTATION

5.1. Introduction

In the previous chapter, we explored the architecture, design, and working principle of the legal consultant chatbot that was proposed. This chapter now focuses on the actual implementation of the proposed solution in the form of code. The implementation involves several steps, including pre-processing the data, building the model, testing and evaluating it, and ultimately creating a prototype. Each of these implementation steps will be detailed and explained in the following sections.

5.2. Working Environment

We have chosen Python as our programming language for pre-processing and implementations due to its popularity, rich libraries, and open-source nature. It is a language with a vast contributor base and numerous support groups. In the field of data science and machine learning, Python is the leading language (Yli-heikkilä, 2019).

5.3. Implementation and Deployment Environment

We utilized Google Colab, a free computation platform that provides access to GPUs and TPUs provided by Google, to implement the model. Google Colab offers an Nvidia K-80 GPU with 12 GB of RAM, and it operates by connecting to Virtual Machines with a maximum lifetime of 12 hours during a specified period (Carneiro et al., 2018). In addition to Colab, we used a physical computer system with an Intel(R) Core i5 processor, 8 GB RAM, and NVIDIA GEFORCE graphics, running on a 64-bit Windows 10 operating system.

5.4. Libraries and Header File

Python is a programming language with a wealth of available libraries that contain pre-written Python modules for performing various functions. In the development of a legal consultation service chatbot, we imported and utilized several libraries for experimentation purposes. The figure below illustrates the libraries used.

```
import nltk
import pickle
import re
import keras
import tensorflow as tf
from keras.preprocessing import sequence
####for Weight initializers$##############################
from tensorflow import initializers
initializergu = tf.keras.initializers.GlorotUniform()
initializergn = tf.keras.initializers.GlorotNormal()
####for Weight initializers$##############################
from tensorflow.keras.preprocessing.sequence import pad_sequences
### for elmo embedding
import tensorflow_hub as hub
from keras.layers import Dense
from tensorflow.keras.utils import plot_model
from keras.models import Sequential, Model
from keras.layers import Embedding, Input, LSTM, Flatten,Dropout,Bidirectional
from tensorflow.keras.layers import  Conv1D, MaxPooling1D, GRU, TimeDistributed,Bidirectional
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau
from keras.initializers import Constant
from tensorflow.keras.optimizers import Adam
from keras.layers import Activation
from sklearn.model_selection import train_test_split
from sklearn.model_selection import StratifiedKFold
from sklearn.model_selection import KFold
import numpy as np
import pandas as pd
```

Figure 5.1: Sample code of importing libraries

5.5. Data preprocessing Implementation

To maximize the usefulness of the collected dataset, we implemented various preprocessing techniques, including cleaning and normalizing duplicate data, and performing feature transformation. Before applying these preprocessing steps, however, we needed to load the dataset into our working environment. Since we used Colab, we retrieved the dataset from Google Drive using the Google.colab package's drive library. We then used Python's Pandas library to store the plain text as a vector form in data frames. The implementation of data retrieval and loading can be seen below.

```
# loading dataset and conacatenate questions & answers
data_path = "Gaffii.txt"
data_path2 = "Deebii.txt"
# Defining lines as a list of each line
with open(data_path, 'r', encoding='cp1252') as f:
  lines = f.read().split('\n')
with open(data_path2, 'r', encoding='cp1252') as f:
  lines2 = f.read().split('\n')
  pairs = list(zip(lines,lines2))
```

Figure 5.2: Sample code of data loading

5.5.1. Implementation of Data Cleaning

The dataset underwent various data cleaning processes such as removing punctuation, numbers, letters, and white spaces, as well as applying text normalization techniques such as lowercasing. This was achieved using the RegEx (regular expression) and NLTK modules in Python, ensuring that the resulting data was suitable for use in Deep Learning algorithms. Unwanted characters such as punctuation marks, special characters, and symbols were removed or replaced from the datasets using these modules. The cleaned datasets were then returned for further use.

```
def clean_text(pairs):
 pairs = re.sub(r'\s+', ' ', pairs)
 pairs=re.sub('\[.*?\]', '',pairs)
 pairs=re.sub('\n', '',pairs)
 pairs = re.sub(r'[^\w\s]','',pairs)
 piars = re.sub(r'\s+', ' ', piars).strip()# Cleaning the whitespaces
#  text=re.sub('[%s?]'% re.escape(string.punctuation),'',text)
 pairs=re.sub('\w*\d\w*', '',pairs)
 pairs = re.sub('([@_]+)|[^\w\s]|#|http\S+', '', pairs)
 return pairs
clean= lambda x:clean_text(x)
```

```
Proccessed_Data=clean
Proccessed_Data
```

Figure 5.3: Implementation of cleaning dataset

5.5.2. Implementation of Tokenization and Stopwords Removal

Implementation of Tokenization is required to get the pattern of the text and obtain the tokens from the dataset. The data collected and prepared is in the form of sentences. Therefore, to make tokenization or convert a text to streams of tokens we have used the NLTK module called word_tokenize () function. To tailor the technique to the Afaan Oromo language, we have collected stop words from several previous studies conducted by (Haile, 2022)**,** (Khanna, 2021)**,** and (Fikadu, 2021)**.** These stop words comprise the most commonly prevalent terms, including pronouns, prepositions, conjunctions, articles, and particles. At this stage, we also removed Afan Oromo stop words from the dataset. Eight six (86) Afan Oromo stopwords was collected and saved in “ormic.txt” have been implemented.

```
# Tokenize the text
tokens = nltk.word_tokenize(pairs)

# Remove stopwords from the tokens
stop_words = set(stopwords.words('oromic'))
filtered_tokens = [token for token in tokens if token.lower() not in stop_words]

# Join the filtered tokens back into a string
filtered_text = ' '.join(filtered_tokens)
```

Figure 5.4: Implementation of stop-words and tokenization

5.5.3. Text Normalization

After tokenizing and cleaning the patterns in the dataset, we applied text normalization as another preprocessing technique to normalize words written differently but having similar meanings. An example is the normalization of *'Ishii' to 'Ishee'*, as shown in Appendix A The following figure 5.4 shows the normalization code. In this code it does so by defining a dictionary *word_dic* which contains a set of contraction words and their corresponding expansion words. The script uses a regular expression *word_re* to identify the contraction words present in the input text. The *word_re* regular expression is created using *re.compile* function, which takes in a string as an argument. The *join* method is used to join the words in the *word_dic* dictionary with the pipe | character, which creates a regular expression pattern that matches any of the contraction words.

```
#Normalization
# Dictionary of words to be norimalized
word_dic = { " otuu": " osoo "," otoo": " osoo "," fundura ": " fulduraa "," fuula ": " fula "," ishii ": " ishee ",
                    " of-eeggannoo":" of eeggannoo "," dhakaa ": "dhagaa "," dhaka ": " dhagaa"," diree ": " waraane ",
                    " gayee ": " gahee "," hiija": " haalo "," hiijaa ": " haalo "," yenna ": " yeroo "," tara ": " yeroo ",
                    " kessaa ": " keessaa "," marsaa ": " yeroo "," dhaqabsiiseef ": " geessiseen ",
                    " himatamtertii ": " himatamtee jirti "," himatamteertii ": " himatamtee jirti ",
                    " himatamera": " himatamee jira "," himatameera": "himatamee jira "," iddoo": " bakka ",
                    " tan": " kan "," si'a": " yeroo "," midhaa": " miidhaa "," dhokaase": " dhukaase "," al": " yeroo ",
                    " ilkee": " ilkaan "," mencaa ": " mancaa "," qaamaa": " qaama "," iubbuu": " lubbuu ",
                    " midhamaa": " miidhamaa "," miidhama": " miidhamaa "," miidhaama": " miidhamaa ",
           }

# Regular expression for finding contractions
word_re=re.compile('(%s)' % '|'.join(word_dic.keys()))

# Function for expanding contractions
def normalize(text,word_dic=word_dic):
    def replace(match):
        return word_dic[match.group(0)]
    return word_re.sub(replace, text)

# Expanding Contractions in the title, text
# Expanding Contractions in the title, text
normalized_lines = []
for pair in pairs:
    normalized_pair = (normalize(pair[0]), normalize(pair[1]))
    normalized_lines.append(normalized_pair)
```

Figure 5.5: Implementation of word normalization

5.6. Feature Extraction Implementation

Since deep learning algorithms cannot directly work with raw text, feature extraction techniques are necessary to convert text data to vectors. In this study, we implement one feature extraction method: *Embedding's from Language Models Representation (ELMO)*. This method takes tokenized data sets and converts them into vectors using feature extractions, as illustrated in Figure 4.3. By utilizing these techniques, we can effectively transform the text data into a format that can be used for training deep learning models.

5.6.1. Implementation of proposed ELMO

ELMO embeddings are used as input features in downstream tasks, created by concatenating the forward and backward LSTM embeddings and passing them through a linear layer to reduce the dimensionality. In this code, we first load the ELMO embedding module from *TensorFlow Hub*. Then, it takes a list of texts as input and returns the corresponding ELMO embeddings as a NumPy array.

```
# Load ELMO embedding module
elmo = hub.load("https://tfhub.dev/google/elmo/3", trainable=True)
# Load and preprocess data
pairs = [d for d in pairs if len(d)>0]
# Convert data to ELMO embeddings
embedded_data = elmo(pairs, signature="default", as_dict=True)["elmo"]
```

Figure 5.6: Implementation code of ELMO feature extraction

5.7. Proposed Model Implementation

After preprocessing the dataset and representing it as a vector, the next step is to build the deep learning algorithms and train the model using the training dataset. As discussed in section 3.8, this study proposed two models, GPT and LSTM. The implementation code for both models is provided below.

5.7.1. GPT Model Implementation

As mentioned in the previous section, the GPT model is a pre-trained model. However, there are several versions of GPT available, including GPT-1, GPT-2, GPT-3, and GPT-4. In this study, we have utilized the GPT-2 version. In order to implement it, we firstly load the pre-trained GPT-2 tokenizer using "*GPT2 Tokenizer.from_pretrained('gpt2')"*. Then after we tokenize the text dataset into a sequence of subwords using *tokenizer.encode(pairs, add_special_tokens=True).*

```
# Load the pre-trained GPT-2 tokenizer
tokenizer = GPT2Tokenizer.from_pretrained('gpt2')
# Tokenize the text into a sequence of subwords
input_ids = tokenizer.encode(pairs, add_special_tokens=True)
# Split the dataset into training and validation sets
train_dataset, val_dataset = train_test_split(input_ids, test_size=0.2)
# Define the data collator for language modeling
data_collator = DataCollatorForLanguageModeling(tokenizer=tokenizer, mlm=False)
# Define the GPT-2 language model
model = GPT2LMHeadModel.from_pretrained('gpt2')
# Define the training arguments
training_args = TrainingArguments(
    output_dir='./results',
    overwrite_output_dir=True,
    num_train_epochs=3,
    per_device_train_batch_size=16,
    per_device_eval_batch_size=32,
    eval_steps=400,
    save_steps=800,
    warmup_steps=500,
    learning_rate=5e-5,
    logging_dir='./logs',
    logging_steps=100,
    seed=42
)
# Define the trainer for training the model
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=TextDataset(train_dataset, tokenizer=tokenizer),
    eval_dataset=TextDataset(val_dataset, tokenizer=tokenizer),
    data_collator=data_collator,
)
# Train the GPT-2 language model
trainer.train()
```

Figure 5.7.1: Implementation of GPT model

5.7.2. LSTM Model Implementation

The LSTM classifier algorithm in this work is implemented using the Keras sequential API model with TensorFlow as the backend. The LSTM model follows an encoder-decoder structure, where the encoder produces a final state vector (memory) that serves as the initial state for the decoder. Training the decoder involves using the teacher forcing method, which allows it to predict subsequent words in a target sequence based on previous words. To indicate the end of a sequence, an '<*END*>' token is added. The model is constructed using the 'add' function to add layers sequentially on top of each other.

After constructing the model structure, the next step is to compile it. Compiling the model involves specifying a loss function, optimizer, and metrics for model prediction. In this work, the model is compiled with a categorical cross-entropy loss and Adam optimizer with accuracy metrics.

To measure the model's ability to generalize on unseen data, it is fitted using the 'model.fit()' module in Keras. The training data is provided to train the model, and the validation data is used to assess its performance. To reduce computational complexity, the data is trained in batches using the specified batch size. Since K-fold cross-validation is used, the model is re-run for each fold iteration. The Python code implementation of the proposed LSTM model is shown below.

```
#Dimensionality
dimensionality = 500
#The batch size and number of epochs
batch_size = 5
epochs = 20
#Encoder
encoder_inputs = Input(shape=(None, num_encoder_tokens))
encoder_lstm = LSTM(dimensionality, return_state=True)
encoder_outputs, state_hidden, state_cell = encoder_lstm(encoder_inputs)
encoder_states = [state_hidden, state_cell]
#Decoder
decoder_inputs = Input(shape=(None, num_decoder_tokens))
decoder_lstm = LSTM(dimensionality, return_sequences=True, return_state=True)
decoder_outputs, decoder_state_hidden, decoder_state_cell = decoder_lstm(decoder_inputs, initial_state=encoder_states)
decoder_dense = Dense(num_decoder_tokens, activation='softmax')
decoder_outputs = decoder_dense(decoder_outputs)
#Model
training_model = Model([encoder_inputs, decoder_inputs], decoder_outputs)
#Compiling
training_model.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy'], sample_weight_mode='temporal')
```

Figure 5.7.2: Implementation of LSTM model

CHAPTER SIX

6. RESULTS AND DISCUSSIONS

6.1. Chapter Overview

Up to this point, we have discussed the problem statement; the methodology employed, the proposed solution with its architecture, and provided detailed explanations of the implementations. In this chapter, we present the results obtained from the executed experiments, which include various hyper-parameter tunings, feature extraction techniques, and other methods. We also engage in a comprehensive discussion on the outcomes and overall achievements of the proposed solution, including expert evaluations and comparisons with previous works in the field.

6.2. Model Evaluation Result

As mentioned in the previous chapter, we conducted experiments to explore various deep learning algorithms, including GPT and LSTM with ELMO word embedding. To achieve the best results, multiple experiments were performed. The experiment follows a hierarchical structure, where each subsequent experiment is based on the findings and outcomes of the previous one. The results obtained from each inspection are described and explained below.

Experiment 1: In our experiments, we investigated the significance of stop words by comparing the performance of models with and without removing them. Considering the importance of every word in legal domain, we initially included all stop words in the analysis. Because, every word would be written seriously in judgment and charge writing. However, we observed that better results were obtained when we avoided or removed Afaan Oromo stop words instead of keeping them in our dataset. The results of feature extraction without removing stop words are provided in Appendix C. Therefore, our focus was primarily on evaluating the performance of models with stop words removed.

Experiment 2: To mitigate over fitting and assess the model's performance on unseen data, we employed cross-validation. In traditional k-fold cross-validation, there is a risk of obtaining highly unbalanced folds, which can introduce bias during model training. However, we addressed this issue by utilizing stratified cross-validation, which preserves the imbalanced class distribution. Since our dataset exhibited class imbalance, stratified cross-validation was employed in our study. As the optimal value of k (number of folds) was unknown, we trained our models using various k-values ranging from 2 to 10 and

selected the best performing model. Among the different k-values tested, the 10-fold stratified cross-validation demonstrated superior performance. Consequently, we employed 10-fold stratified cross-validation in this study. The results of each k-value (2-10) are presented in Appendix D.

Experiment 3: *the result of hyper-parameters*; the hyper parameter tuning process was conducted to determine the optimal parameters for the models. Chapter 3, Section 3.11 explains different hyper parameter algorithms, among which grid search and random search were utilized. Comparing the results, it was found that random search with stratified 10-fold cross-validation produced the best outcomes for the two models. Appendix D2 provides the results of grid search with stratified 10-fold cross-validation for the two models. The performance of the two selected models in terms of accuracy, precision, recall, and F1-score were analyzed to determine the top-performing model.

6.2.1. Legal Consultant Chatbot Models Evaluation Results

As discussed in previous chapters, we have established a plain text dataset to develop a generative legal consultant chatbot. Using this dataset, we trained and tested models and evaluated them using 10-fold stratified cross-validation. In addition, we used other evaluation metrics, such as precision, recall, F1 score, and perplexity. The mean or average accuracy of each test for GPT and LSTM models are presented in Table 6.4, based on the obtained vectors.

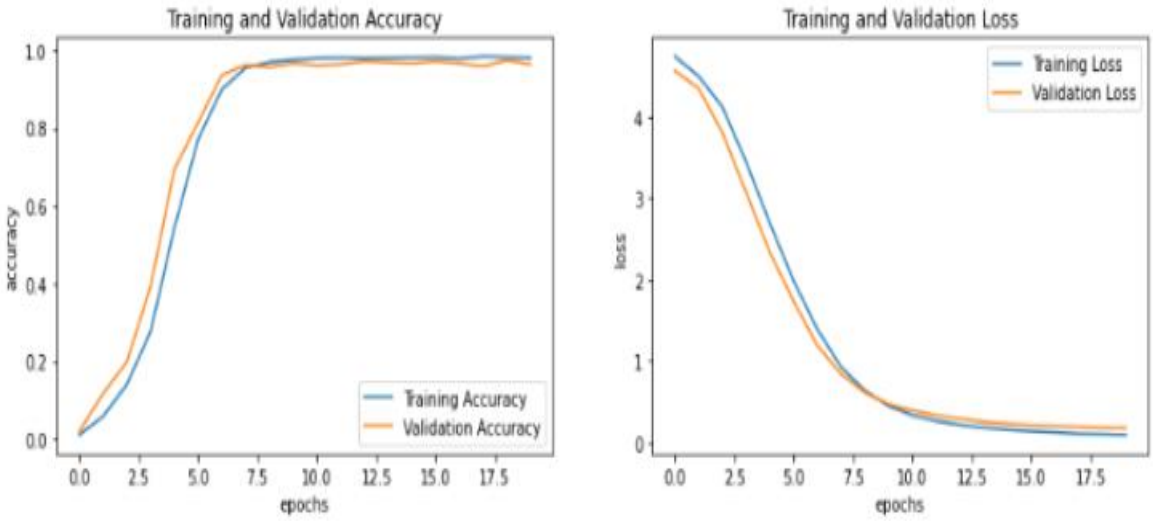


Figure 6.6.1: *Legal Consultant Chatbot Models Evaluation Results*

Table 6.1: 10 fold stratified CV of average accuracy for two models

Feature Extraction	*GPT model with 10 Fold Average Accuracy (mean) %*	*LSTM model with 10 Fold Average Accuracy (mean) %*	*Remark*
ELMO	**87.5**	**81.2**	

The result shown in Table 6.1 is the result obtained by default parameters. Through default parameters, the GPT model with ELMO, and LSTM model with ELMO yielded higher results, that 87.5% and 81.2 % CV average accuracy, respectively. Among those models GPT yielded higher CV average accuracy on LSTM models.

Experiment 4: The figure presented in Figure 6.1 the occurrence of overfitting in the models. To mitigate this issue, L2 regularization was implemented in this study. The application of regularization had an impact on the results obtained from the non-regularized model. After incorporating the regularization technique, the model was evaluated and the subsequent results were obtained.

Table 6.2:10-fold stratified CV of average accuracy result after regularization

Feature Extraction	*GPT model with 10 Fold Average Accuracy (mean) %*	*LSTM model with 10 Fold Average Accuracy (mean) %*	*Remark*
ELMO	**88.9**	**84.6**	

The introduction of L2 regularization techniques resulted in a noticeable change in both the training loss and test loss of the model. The graphical representation below displays the outcomes of the model in terms of train loss and test loss.

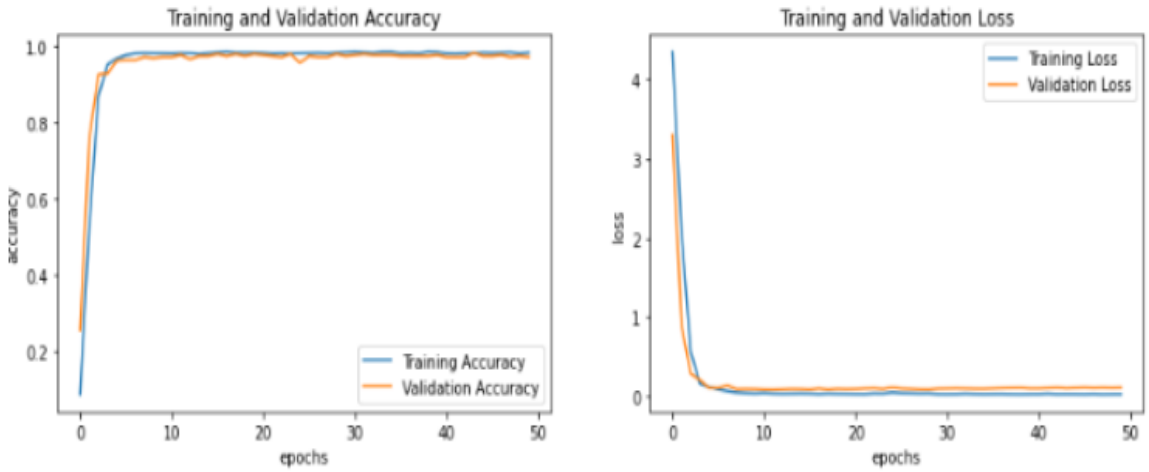


Figure 6.6.2: Legal Consultant Chatbot Models Evaluation Results after L2 regularization

6.3. Hyper-parameter Tuning Results

In this study, the random search hyperparameter tuning method was employed to search for the optimal combination of hyperparameters for the proposed models. The Keras_Tunner hyperparameter tuning algorithm was utilized to identify the best performing hyperparameters. Through this process, the study successfully determined the optimal hyperparameters for the proposed models, including parameters such as epochs, batch size, number of neurons, activation function, optimizers, learning rate, filters, kernel size, and hidden layers. Various values were explored for each parameter, resulting in the identification of the most effective combination. Through the exploration of different values, the following optimal hyperparameter values were determined for all proposed models.

Table 6.3: Optimal hyperparameters

Models	*Epoch*	*Batch Size*	*Optimizer*	*Activation*	*Learning Rate*	*Kernel Size*	*Hidden Layer*	*Filters*
GPT	**10**	**32**	Adam	GELU	5e-5			
LSTM	**20**	**5**	rmsprop	softmax	0.001	7	2	64

GPT-2 does not use traditional CNN components such as kernel size, hidden layers, and filters. It is a transformer-based language model that relies on self-attention mechanisms and feed-forward neural networks.

In the GPT-2 architecture, the model consists of a stack of identical layers. Each layer includes a multi-head self-attention mechanism and a position-wise feed-forward neural

network. The size of the hidden layer and the number of attention heads vary depending on the specific variant of GPT-2. The self-attention mechanism in GPT-2 allows the model to capture relationships between different positions in the input sequence, enabling it to understand and generate coherent text (Radford et al., 2019).

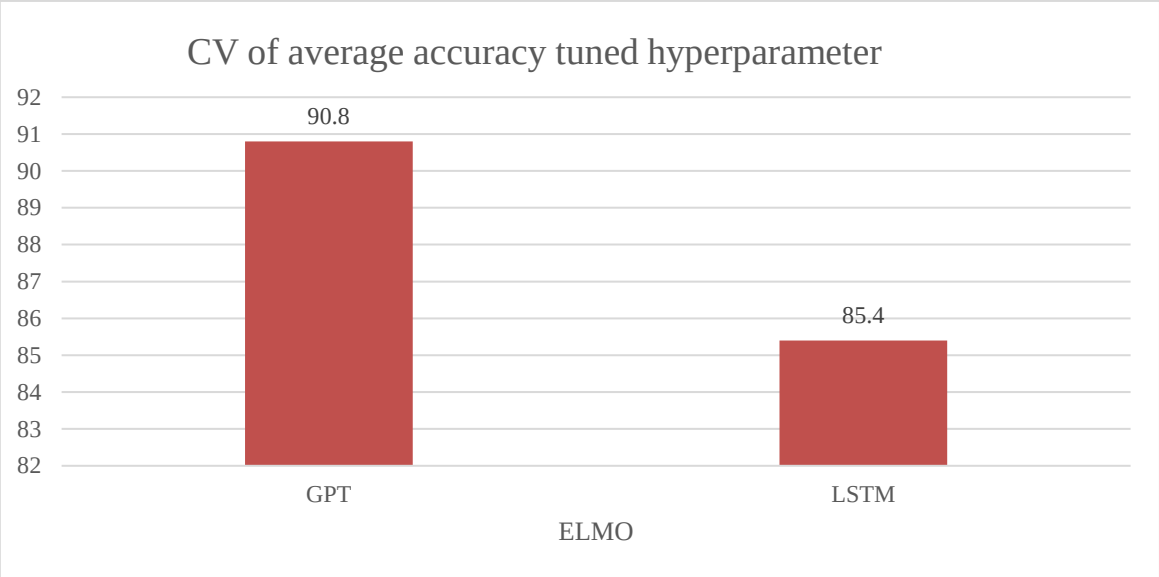


Figure 6.6.3: Result of hyper-parameters

6.4. Threshold values

The threshold value refers to a numerical or qualitative criterion used to filter or control the output of the model. It is a value that helps determine a generated response is considered acceptable or relevant based on certain criteria. Determining the threshold value in GPT or any generative model is an important aspect of controlling the output quality and relevance (S. Dodge et al. (2020)). In this study, the following approaches have been used to determine the values of threshold for proposed model.

Manual selection: we have been experimented with different threshold values by adjusting its value up or down to find a balance between generating diverse responses and maintaining quality.

Evaluation Metrics: perplexity evaluation metrics have been also used in order to select good threshold value. Perplexity is typically calculated using the cross-entropy loss, which measures the difference between predicted and actual token distributions.

6.5. Result of Human Evaluation

As mentioned in chapter three, we selected the model with the most favorable outcomes for deploying the prototype. To evaluate the model, we integrated it into a web platform

using Flask server and Telegram. To facilitate the evaluation process, we developed a user interface utilizing HTML, CSS, and JavaScript (refer to Appendix E0). Two legal experts, including a judge and an advocate from the OSC, along with twenty court clients, participated in the evaluation. The experts assessed the model's performance by reviewing the evaluation sheets provided in Appendix E1. The evaluation primarily focused on the model's ability to correctly answer questions. We utilized a specific formula to calculate the accuracy of the proposed model.

$$Correctness = \frac{total\ number\ of\ questions \text{ accurately answered}}{total\ number\ of\ questions \text{ provided for the model}} * 100 \qquad (6.1)$$

Table 6.4: The human evaluation results on model performance

Number of persons	*Total number of entered questions*	*Total number of answered correctly*	*The total number of questions not correctly answered*	*Correctness in %*
OSC law experts (2)	20	15	5	75%
Court clients (20)	50	37	13	74%
Total	70	62	18	**88.5%**

Based on the evaluation conducted by various law experts and court clients, the model's performance was assessed by comparing the number of questions provided to the model with the total number of accurately generated responses. As shown in the aforementioned table, the assessment revealed that our proposed model achieved an accuracy rate of 88.5% according to the law experts' evaluation, while 11.5% of the answers were found to be incorrect.

6.6. Discussion

To the best of our knowledge, this study represents the first attempt to develop a generative legal consultant chatbot specifically tailored to the Oromia Supreme Court, utilizing generative and classical deep learning models trained on Ethiopian law and Afan Oromo language. Through a series of experiments conducted on a newly collected dataset, promising results have been obtained. The study focused on comparing the performance of the generative model GPT and the deep learning model LSTM with ELMO word

embedding. Additionally, the impact of removing stop-words from the dataset and retaining them was examined. The results, as depicted in Figures 6.5 and 6.10, and Tables 6.2 and 6.4, indicate that the GPT model outperformed the LSTM model in terms of f1-score, accuracy, and precision when addressing questions from court clients. Overall, the findings highlight the effectiveness of training the GPT model with optimized parameters, removing stop-words, and leveraging ELMO feature extraction for achieving favorable outcomes.

Furthermore, this section addresses the research questions posed in Section 1.4 of Chapter One, highlighting the diverse techniques employed to provide answers. The study utilized a range of approaches to tackle these questions, and now we will delve into the specific techniques employed and how the proposed model effectively addressed each question individually.

RQ1*: How to design & develop naturalness conversations maintained during Afaan Oromoo converse of the model?* To address this question, various techniques and methods were utilized. The first technique involved reviewing and analyzing Afan Oromo conversations. The second technique focused on preparing the dataset and optimizing it for model training through data processing techniques. The third technique revolved around selecting a model that could maintain the natural flow of conversation. In this study, the GPT model was chosen for its superior accuracy. GPT models are specifically designed to preserve the naturalness of conversations by leveraging their capacity to learn patterns and generate coherent text.

RQ2: Which are the deep learning models that may improve the performance of chatbot for consultancy services? To address this question, we initially established criteria for model and feature selection. Based on these criteria, we carefully selected the models and performed feature extraction. Subsequently, we conducted training and testing of the models using the prepared dataset. To identify the best-performing models, we employed techniques such as model performance evaluation using stratified 10-fold cross-validation, classification metrics including accuracy, precision, recall, and f1-score, and human evaluation. Through our empirical analysis, we found that the GPT model with ELMO word embedding yielded the best results for providing legal consultant services for the Oromia Supreme Court.

RQ3: How can we improve the quality of legal consultant Chatbot model? In addition to dataset preparation and model selection, optimizing the models by finding suitable

hyperparameter techniques is crucial. In this study, we compared the grid search and randomized hyperparameter techniques to identify the optimal parameters for our models. Initially, we evaluated the model with default parameters. However, after tuning the parameters using the randomize technique, we observed improved results compared to the default parameters. Furthermore, we employed L2 regularization as a regularization technique in our models.

As discussed in Chapter Two, Section 2.3, numerous studies have been conducted on developing chatbots using deep learning and other NLP techniques. For the purpose of comparison, we have carefully selected two studies that closely align with our research and share similar objectives.

In a study conducted by Segni (Segni, 2021), DNN and Seq2Seq models were employed, achieving an accuracy of 84.4% with the DNN model. However, when comparing this study to our proposed models, our models outperform in terms of accuracy, as discussed in Table 5.1. It should be noted that the author of the mentioned study utilized a small dataset consisting of only 25 tags and did not employ techniques to address model overfitting. In contrast, our proposed models incorporate cross-validation methods and the L2 regularization technique to mitigate overfitting. Additionally, the author did not utilize a threshold value to prevent the model from generating inappropriate responses, which is a consideration in our proposed model.

The second study conducted by Fekadu (Fekadu, 2021) utilized the BiLSTM model and achieved an accuracy of 82.6% with the incorporation of L2 regularization algorithms to address model overfitting. When comparing Fekadu's study to the presented model in our study, our models perform better in terms of accuracy across all models. Additionally, the author manually set the hyperparameters for the neural network, which can be a challenging and time-consuming task to find the optimal parameters. In contrast, our study utilized the random search hyperparameter tuning algorithm, which automates the process of finding the optimal hyperparameters for the models.

Table 6.5: Comparison of our model with previous work

Author	*Methodology*		*# of instance in Dataset*	*Types of Chatbot*	*A model with the best accuracy % result*
	Model	***Feature Extraction***			
Segni Bedasa (2021)	DNN, Seq2Seq	BOW		Retrieval Based	84.4
Fekadu Eshetu (2021)	BiLSTM	BOW		Retrieval Based	82.6
Proposed model	GPT, LSTM	ELMO	2 MB	Generative Based	90.8

6.7. CONCLUSION, CONTRIBUTION, AND FUTURE WORK

6.7.1. Conclusion

The study presents the development of an Afaan Oromo text-based legal consultant chatbot using deep learning approaches. Various approaches used in modern chatbot design and related works are reviewed in this study. To achieve the objective, a primary dataset containing 2MB was collected from various sources, including the Oromia Supreme Court, Telegram, Facebook, and websites, as no prepared datasets were previously available for this purpose. The collected dataset was reviewed by two legal experts to ensure its accuracy. Several data preprocessing techniques, such as tokenization, stop word removal, and removing unrelated items, were employed to prepare and clean the dataset for deep learning models. Additionally, word embedding techniques, specifically ELMO models, were utilized to represent textual data with vector representations.

For this work, we proposed two deep learning models: GPT and LSTM. We evaluated these models using 10-fold cross-validation techniques. To address the issue of model over fitting, we applied the L2 regularization technique, considering the smaller size of our dataset. The performance of the proposed models was assessed by tuning the appropriate network hyperparameters using a random search hyperparameter tuning algorithm for all models.

In section 6.2, the results of the model are discussed, revealing that the GPT model outperforms other algorithms. This superiority can be attributed to the fact that the GPT model's output layer incorporates information from both forward and backward directions in the current state, enabling it to learn intricate sequence correlations effectively. Overall, this study has determined that the GPT model exhibits the highest performance among the proposed deep learning models, achieving an accuracy of 90.8%.

6.7.2. Contribution

We have employed various techniques and approaches to tackle the problem stated in chapter one, section 1.4, and to address the research questions. Through conducting experiments, we have made significant contributions by introducing new elements and enhancing the quality of chatbot responses. The primary contribution of this study is as follows:

- *We have created a new dataset collection*, which is crucial in the field of deep learning. Prior to this study, there was no existing legal dataset

specifically prepared in the Afan Oromo language. Therefore, we employed various techniques to create this new dataset.

- *Improving the quality of chatbot responses*: To enhance the quality of chatbot responses, we utilized two techniques. The first technique involved selecting a generative model, while the second technique involved employing a hyperparameter tuning technique.

6.7.3. Recommendation

The prevalence of machine learning in the legal or law domain, especially in Ethiopian law, is currently low. Moreover, the Ethiopian courts face numerous challenges. Therefore, we recommend that other researchers explore the legal domain, specifically Ethiopian law, and utilize different algorithms to address the problems faced by the Ethiopian courts and modernize their operations. The positive results obtained from the model evaluation in this study, along with the favorable human evaluation results from law experts, indicate the potential impact of this research. Transforming this study into a project has the potential to enhance legal skills and raise awareness of court clients. We suggest that the Oromia Supreme Court review this study and consider its conversion into a project.

6.7.4. Future Work

This work is aimed to develop an Afaan Oromo text-based legal consultant chatbot for OSC. In this study efforts have been made to fill some gaps, regardless of that still there are a lot of issues that have not been addressed by this research. Therefore, we have mentioned some issues to be used as future work. Increasing the dataset size by collecting more legal documents, frequently asked questions, and relevant case studies will allow the chatbot to have a broader knowledge base and better understand user inquiries. Moreover, expanding the chatbot's language coverage to include local languages spoken in the target region will greatly enhance its accessibility and usability for users who may prefer or be more comfortable interacting in their native language. In addition to this, to provide a more seamless and intuitive user experience, future work can focus on integrating voice chat functionality into the legal consultant chatbot. This can involve leveraging advanced language models, such as transformer-based architectures like GPT-3.

7. REFERENCES

Abinash Tripathy and Ankit Agrawal and Santanu Kumar Rath. (2016). *N-grams, which are N consecutive words combined into a single token*.

Allahyari et al. (2017). *removing certain special characters and normalizing the text to lowercase*.

Amazon Lex Documentation. (2020). *official Amazon Web Services (AWS) documentation and resources for detailed information about Amazon Lex and AWS Lambda.* https://docs.aws.amazon.com/lex/ and https://docs.aws.amazon.com/lambda/

Amazon Web Services. (2018). *Build Conversation Bots*. *17*.

Asafa, M. M. (2010). *The Phonology and Morphology of the Verb in Wallo Oromo*.

Barker, S. (2017). How chatbots help. *MHD Supply Chain Solutions*, *47(3):*, 30.

Blumstein, A., & Rosenfeld, R. (2008). A Test of the Race-Crime Invariance Thesis: Black-White Differences in the Impact of Employment and Poverty on Crime. *Criminology*, *46*(4), 789–824. https://doi.org/DOI: 10.1111/j.1745-9125.2008.00130.x

Botsify. (2021). *researching and exploring different chatbot building tools available in the market*.

Boulila et al. (2022). *superior performance in neural network models*.

Brady D. Lund. (2023). A Brief Review of ChatGPT: Its Value and the Underlying GPT Technology. In *University of North Texas*.

Brownlee. (2020). *straightforward method*.

Carneiro et al. (2018). *implement the model*.

Chatterbot GitHub Repository: (2022). *The GitHub repository of Chatterbot contains the source code, issue tracker, and additional resources related to the library.* https://github.com/gunthercox/ChatterBot

Chu, M., Jones, A., Clancy, A., & Donnelly, S. (2014). Beginning to Dream with Families, School, University and Community: Starting Collaborative Partnerships for Everyone's Learning. *Education in a Democracy: A Journal of the National Network for Educational Renewal*, *6*, 47–68. http://www.nnerpartnerships.org/wpcontent/files/nner_journal_online_final_2014.pdf)

Chung, K., & Park, R. C. (2019). Chatbot-based healthcare service with a knowledge base for cloud computing. *Cluster Computing*, *22*, 925–1937. https://doi.org/DOI: 10.1007/s10586-018-2334-5

Colby, K. M., Weber, S. G., & Hilf, F. D. (1971). Artificial paranoia. *Journal of Nervous and Mental Disease*, *153*(6), 394–401. https://doi.org/10.1097/00005053-197112000-00005

Colby, K. M. (1981). Modelling a paranoid mind. *Behavioral and Brain Sciences*, *4*(4), 515.

D. Duresa. (2016). *Large Vocabulary Continuous Speech Recognition System for Afaan Oromo using Hidden Markov Model*. Unpublished" Master"s Thesis. Thesis Submitted to ASTU, 107 pp.

Dale, R. (2016). The return of the chatbots. *Natural Language Engineering and Ambridge University Press*, *22*(5), 811–817. https://doi.org/DOI: 10.1017/S1351324916000243

Dialogflow. (2010). https://dialogflow.com/

F. Colace, M. De Santo, M. Lombardi, F. Pascale, A. Pietrosanto, and S. Lemma. (2018). Chatbot for e-learning: A case of study. *Int. J. Mech. Eng. Robot. Res*, *7*, 528–533. https://doi.org/doi: 10.18178/ijmerr.7.5.528-533.

Fikadu. (2021). *Designing and Developing Bilingual Chatbot for Assisting Ethio-Telecom Customers on Customer Services*. Unpublished" Master‟s Thesis. Thesis Submitted to ASTU, 110 pp.

Framework Definition. (2021a). https://techterms.com/definition/framework

Framework Definition. (2021b). *Https://Techterms.Com/Definition/Framework*.

G. Cox. (2019). *ChatterBot Documentation*.

Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition. *O'Reilly Media Chapter 2: End-to-End Machine Learning Project.*

Google. (2023). *Dialogflow*. https://cloud.google.com/dialogflow/docs

Graves, A., Mohamed, A., & Hinton, G. (2013). Speech Recognition with Deep Recurrent Neural Networks. *IEEE International Conference on Acoustics, Speech and Signal Processing*, 6645–6649.

Habtamu. (2020). *development of a chatbot to assist individuals with Ethiopian law-related queries*. Unpublished" Master‟s Thesis. Thesis Submitted to ASTU, 116 pp.

Hussain, M., Abbas, H., Hussain, S., & Kang, B. H. (2019). Artificial Intelligence-Based Chatbot: A Review. *Journal of Intelligent & Fuzzy Systems*, *36*(4), 3305–3316. https://doi.org/10.3233/JIFS-179897

IBM. (2021). *consulting IBM's official documentation and resources.* https://www.ibm.com/cloud/watson-assistant/

J. Khare. (2019). *Applications of Natural Language Processing*.

Jabberwacky. (1982). *designed to operate as a text-to-voice system and aimed to mimic natural human conversation for entertainment purposes.* Rollo Carpenter.

Joseph, S. R., Hloman, H., Letsholo, K., Kaniwa, F., & Sedimo, K. (2016). Natural Language Processing: A Review. *International Journal of Research in Engineering and Applied Sciences*, *6*(3), 207–210. http://www.euroasiapub.org

Jurafsky, D. and J.H. Martin. (2017). *Speech and Language Processing (2nd Edition)*. 418–440.

K. D. Ashley. (2017). Artificial intelligence and legal analytics: New tools for law practice in the digital age. *Cambridge University Press*, *6*, 45–125.

Kandpal, M., Goyal, R., & Gupta, V. (2020). Recent advancements in chatbot technologies: A systematic review of literature. *International Journal of Information Management*, *53*, 102096. 10.1016/j.ijinfomgt.2020.102096

Kandpal et al. (2020). Recent advancements in chatbot technologies: A systematic review of literature. *International Journal of Information Management*, *53*, 102096. https://doi.org/10.1016/j.ijinfomgt.2020.102096

Kapetanios. (2016). Deep learning. *Research Communities*.

Kohavi, R. (1995). A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI)*, *2*, 1137–1143.

L. Cui, S. Huang, F. Wei, C. Tan, C. Duan, and M. Z. (2017). Superagent: A customer service chatbot for E-commerce websites. *55th Annu. Meet. Assoc. Comput. Linguist. Proc. Syst. Demonstr*, 97–102. https://doi.org/doi: 10.18653/v1/P17-4017.

Lemaitre, C., C. A. Reyes, and J. (2004). Advances in Artificial Intelligence. *In 9th Ibero-American Conference on AI*, 22–26.

M. Partner et al. (2022.). *HORIZON SCANNING forward thinking Artificial Intelligence*.

Mohammad Nuruzzaman, O. K. H. (2018). *A Survey on Chatbot Implementation in Customer Service Industry through Deep Neural Networks*.

Mohammed Abdurahman. (2022.). *A Grammar of the Oromo Language*.

Nabi. (2019). *hyperparameter tuning algorithms*.

Nay, C. (2011). Knowing what it knows: selected nuances of Watson's strategy. *IBM Research News*, 50.

Negesse, A. (2015). Oromo Grammar: An Introduction. In *Book*.

Nuruzzaman, M., & Hussain, M. (2018). Internet of Things (IoT) Based Smart Healthcare System: A Review. *Journal of Medical Systems*, *42*(7), 130. https://doi.org/DOI: 10.1007/s10916-018-0997-6

Nuruzzaman M, H. (2018). A survey on Chatbot implementation in customer service industry through deep neural networks. *International Conference on E-Business Engineering (ICEBE), IEEE.*, 54–61.

OpenAI. (2021). *GPT-3.5 Architecture*.

Pandorabots. (2019). *allows users to create and publish chatbots across various platforms*. https://home.pandorabots.com/

Peter C. Brown, Henry L. Roediger III, M. A. M. (2012). *Make it Stick: The Science of Successful Learning*. The Belknap Press of Harvard University Press. https://doi.org/DOI: N/A (as it is a book)

Purva Huilgol. (2020). *Precision and Recall*.

Radford et al. (2018). *self-attention mechanisms*.

Radford et al. (2019). Language Models are Unsupervised Multitask Learners. *The Paper Introducing GPT-2, Which Discusses Its Architecture, Model Size Variations, and Performance on Various Language Tasks.*

Risley, J. (2016). *Microsoft introduces Cortana-powered chatbot for Skype, opening up framework to developers*. Geek Wire.

S. Machiraju and R. Modi. (2018). *Developing Bots with Microsoft Bots Framework*.

Segni. (2021). *Developing Afaan Oromo Text-based Chatbot for Enhancing Maize Productivity*. Adama science and technology.

Sharma. (2022). *Stratified k-fold cross-validation*.

Shawar, B. A., & Atwell, E. (2007). Stemming Arabic Script: A New Approach and Its Applications. *Journal of Quantitative Linguistics*, *14*(1), 47–73. https://doi.org/DOI: 10.1080/09296170701267951

Shawar, Bayan and Atwell, E. (2002). *A comparison between Alice and Elizabeth chatbot systems*.

Shubhashri G et al. (2018). *LAWBO: A Smart Lawyer Chatbot*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). *Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research*. *15(1)*, 1929–1958.

Team. (2020). *varying sizes or with imbalanced class distributions*.

Thompson, I. (2021). *About World Languages*. http://aboutworldlanguages.com/oromo

Tolessa, D. (2013). Aspects of Oromo Grammar: An Inventory and Analysis of Nominal Inflection. *Nordic Journal of African Studies*, *22*(1), 66–88.

Turning, A. M. (1950). *Computing Machinery and intelligence*. *236*, 433–460.

Turning, A. M. (1950). Computing Machinery and Intelligence. *Mind*, *LIX (236)*, 433–460.

TutorialsPoint. (2020). *No Title*. https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligence_natural_language_processing.htm

Vorada et al. (2021). *provide confidential legal guidance and support to sexual violence victims and survivors chatbot.*

Wallace, R. S. (2003). The Anatomy of AIML. *AAAI Spring Symposium on New Directions in Question Answering*, 161–165.

Weizenbaum, J. (1966). ELIZA: a computer program for the study of natural language communication between man and machine. *Commun ACM*, *9(1)*, 36-45.

Wikipedia. (2021). *Oromo_language*. https://en.wikipedia.org/wiki/Oromo_language

Worswick, S. (2010). *Mitsuku Chatbot: Mitsuku now available to talk on Kik messenger*. https://www.pandorabots.com/mitsuku/

X. Zhang et al. (2020). Personalized Digital Customer Services for Consumer Banking Call Centre using Neural Networks. *Proc. Int. Jt. Conf. Neural Networks*. https://doi.org/doi: 10.1109/IJCNN48605.2020.9206709.

Yli-heikkilä. (2019). *guide practitioners and scientists in the domain of data science for choosing their main tool, the programming language.*

8. APPENDICES

Appendix A: Normalization Words

Table 1: List of normalized words

Dhaka=dhagaa	"iddoo": "bakka"
Of-eeggannoo =of eeggannoo	"himatamera": "himatamee jira"
"gayee ": "gahee"	"irra": "gubbaa"
Fundura=fulduraa	"dhaqabsiiseef ": "geessiseen"
Ishii=ishee	"himatamteertii ": "himatamtee jirti"
Otuu=osoo	"kessaa ": "keessaa"
"si'a": "yeroo"	"dhokaase": "dhukaase"
“Otoo”:”osoo”	"marsaa ": "yeroo"
Diree=waraane	"tan": "kan"
"ilkee": "ilkaan"	"al": "yeroo"
"yenna ": "yeroo"	"mencaa": "mancaa"
"hiijaa ": "haalo"	"Fuula" "fula"
"himatameera": "himatamee jira"	"himatamtertii ": "himatamtee jirti"
"himatamera": "himatamee jira"	"dabree","darbee
"ugaa","dhugaa"	"Dhakaa ":dhagaa "
"mini","miti"	"baayyee","baay’ee"
"haga","hanga"	"Waahee","Waa’ee"
ykn","Yookiin"	"eennu","eenyu"

Appendix B: Afaan Oromo Stop-words

Table 2: List of stop-words

Isin	eega	yeroo	koo
Illee	kun	teenya	tanaafuu
Itti	kiyya	gama	isaaf
Narraa	sun	akkasumas	kanaafi
keessatti	natti	kanaafuu	aanee
yookaan	keenya	kee	ittuu
Ammo	tanaaf	isatti	sitti
Dura	booddee	iseen	nu
Eegasii	alatti	ani	utuu
Ishii	henna	amma	ala
ishiirraa	kan	duuba	yoo
Booda	immoo	silaa	isii
Nuti	keessa	ati	an
ta'ullee	gidduu	yommuu	tun
Bira	yoom	isiin	sana
Keessan	jara	akka	irra
Na	nurraa	inni	ishiif
Kana	siin	isaa	malee
Hanga	fi	isaanirraa	
akkuma	kanaaf	hogguu	
Naaf	keenna	waan	
Gubbaa	jala	warra	

Appendix C: Results of Feature Extractions without Remove Stop-Words

Table 3: Feature extractions without remove stop-words of two models

Feature Extraction	*GPT model with 10-Fold Average Accuracy (mean) %*	*LSTM model with 10-Fold Average Accuracy (mean) %*	*Remark*
ELMO	**89**	**84**	

Appendix D: Result of Stratified CV Average Accuracy of two Models with *k* Values (2-10) on Legal conversation Dataset on Default Parameters

Table 4: Average accuracy (%) of GPT model results

Feature Extraction	***K-values***								
	2	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
ELMo	80.44	85.98	86.68	89.69	90.46	91.77	92.85	93.25	**90.8**

Table 5: Average accuracy (%) of LSTM model results

Feature Extraction	***K-values***								
	2	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
ELMo	82.45	87.62	86.21	89.81	89.99	89.66	91.69	92.54	**81.**

Appendix D2: Result of Grid Search with Stratified 10 Fold Cross Validations

Table 6: Grid search result with stratified 10 fold CV of legal conversation model

Feature Extraction	*GPT model with 10 Fold Average Accuracy (mean) %*	*LSTM model with 10 Fold Average Accuracy (mean) %*	*Remark*
ELMo	**86**	**82.5**	

Appendix E: Sample Legal Conversational Dataset

```
Abbaa alanga raga akkamin funaanu danda’aa?
Abbaan alangaa raga yakkaa funaanuuf hammam itti fudhachuu danda’aa?
Shakkamaan tokko mirga wabii argachhuf hamam turuu danda’aa?
Shakkamaan tokko sagalee ragaa osoo hin dhageefatiin sa’aa meeqaaf turuu danda’a?
Abbaan alangaa yeroo bellamaa guyyaa meeqaaf gaafachuu danda’aa?
Turtiin galmee shakkamaa poolisii bira qorannoof turuu hammamiif?
MM “himanicha ni mormitaa” yeroo jedhuu maal jechuu dha?
Yakka kanaan, kanaan dura himatamtettaa?
Yakkichi darbinsa yeroon sii hin himachisuu
Dhimma MM kanaan himatamte belleema irraa yoo jiraattee?
Yakka himatamef amanuun adabbii salphisuu danda’aa?
Yoomi fi sadarkaa kamitti mirga wabii gaafachu danda’aa?
```

```
Namaan,barreefamaan
 Sa’aa 48
Mana murti ergaa dhaabate booda.
sa’aa 48 tiif turuu danda’aa
guyyota 14 gaafachuu danda’a
 Haala yakkichaatu murtessaadha. Yoo walxaxaa ta’uu baatee sa’aa 48 booda abbaa alangaaf qindaa’ee dhihaata.
Yakka kanaan, kanaan dura himatamtettaa? Yakkichi darbinsa yeroon sii hin himachisuu, Dhimma MM kanaan himatamte
jiraattee?
Eyyeen ,
Erga ragaan Abbaa alangaa dhagaa'amee dhumee
Bu’uura s/d/f/y keww. 191 akka siif daanga’u.
Mana murtii sadarkaa sadarkaan jirannitti guyyaa jahaatama keessaati oliyyachuu qabda.
barreefamaan
Mirga deebii barreefamaa dhiheefachuu hambisaa
Eeyyen:-
MM dhimicha ilaaluuf angoo kan hin qabnee  yoo ta’ee
Dhimmichi kanaan dura murtii kan argatee yoo ta’ee
Himanicha daangaa yeroon kan dhorkamee yoo ta’ee
```

Figure 1: Sample collected dataset

Appendix F: Prototype of Legal Consultant Service Chatbot

Appendix G: Questionaries Form

SeerluBot

Gaaffilee

Kabajamtoota Deebii kennitoota keenya, kaayyoon gaaffilee kanaa mana murtii Waligalaa Oromiyaatiif chatbootii gorsaa bilisaa seeraa keennuu danda'uu misoomsudhaaf kuus-daataa walitti qabuuf kan qophaa'edha. Kanaaf gaaffilee armaan gadii kanneenif odeeffannoo sirrii kan ta'uu akka nuuf kennitan kabajaan isin gaafanna. deebii isiin kennitan gatii guddaa kan qabuu fi odeeffannoon walitti qabame iccitii ta'ee kan eegamu yoo ta'u, kaayyoo qorannoo kanaaf qofa kan itti fayyadamnu ta'a.

Kutaa I. Odeeffannoo Duubbee Deebii kennitootaa

Qajeelfamoota:

- ✓ tokkoon tokkoon gaaffiidhaaf akka barbaachisummaa isaatti mallattoo " X" kaa'i
- ✓ Maqaa kee barreessuun barbaachisaa miti.

1. saalaa

A. Dubartii ☐ B. Dhiira ☑

3. Sadarkaa Barnootaa

A. Qalamaa ☐ B. Dippiloomaa ☑ C. Digirii fi isaa ol ☐

4. Gara Mana Murtii maalif Dhuuftan?

A. Nama himadhee ☑ B. Himatamee ☐ C. Abukaatoo ta'e ☐ D. Abuukaatoo itisaa ta'e ☐

E. Ragaan ta'e ☐ F. dhimma birootiif ☐

Kutaa II. Gaaffii Gorsaa bilisaa Seeraa Chatbot waliin walqabatu

4. Hubannoo seeraa ni qabdu?

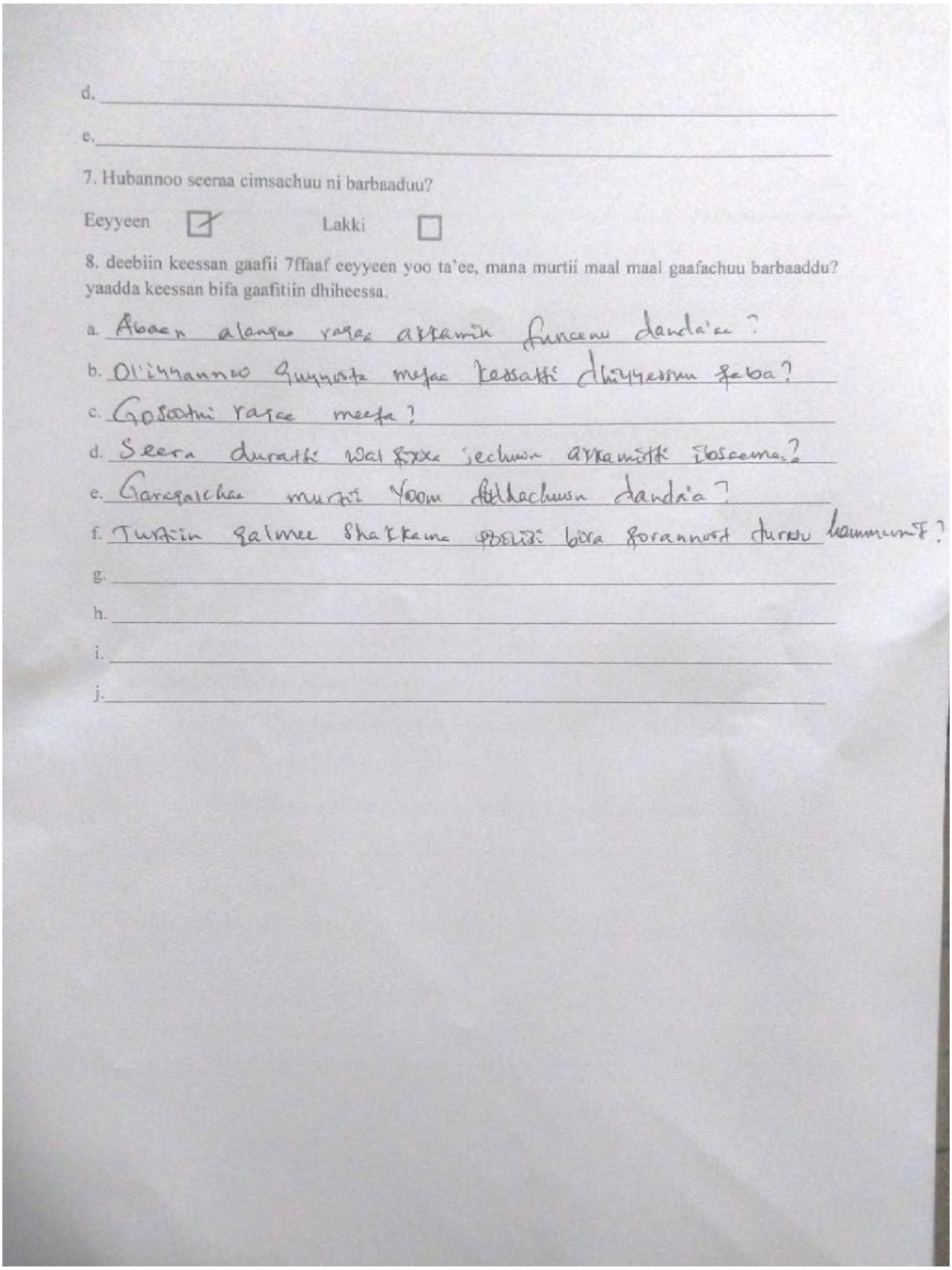
Eyyeen ☐ Lakki ☑

5. Deebiin keessan gaafii lakka 4 lakki yoo ta'ee, dhimma maalii irratti hubannoo argachuu barbaaddu?

a. Dhimmota tajaajila mana murtii

b. Dhimmaa Yakka

c. _____



d. ______

e. ______

7. Hubannoo seeraa cimsachuu ni barbaaduu?

Eeyyeen ☑ Lakki ☐

8. deebiin keessan gaafii 7ffaaf eeyyeen yoo ta'ee, mana murtii maal maal gaafachuu barbaaddu? yaadda keessan bifa gaafitiin dhiheessa.

a. Abaan alangaa ragaa akkamiin funcenu danda'aa?

b. Ol'iyyannoo Guyyoota meqaa keessatti dhiyyeessuu qaba?

c. Gosootni ragaa meeqa?

d. Seera durattii wal jechuun akkamitti ibsama?

e. Gareegalchaa murtii yoom fudhachuun danda'a?

f. Turtiin galmee shakkamaa poolisii bira qorannoof turuu hammamiif?

g. ______

h. ______

i. ______

j. ______

Figure 3: Questionaries form

Appendix H: Human Evaluation Form

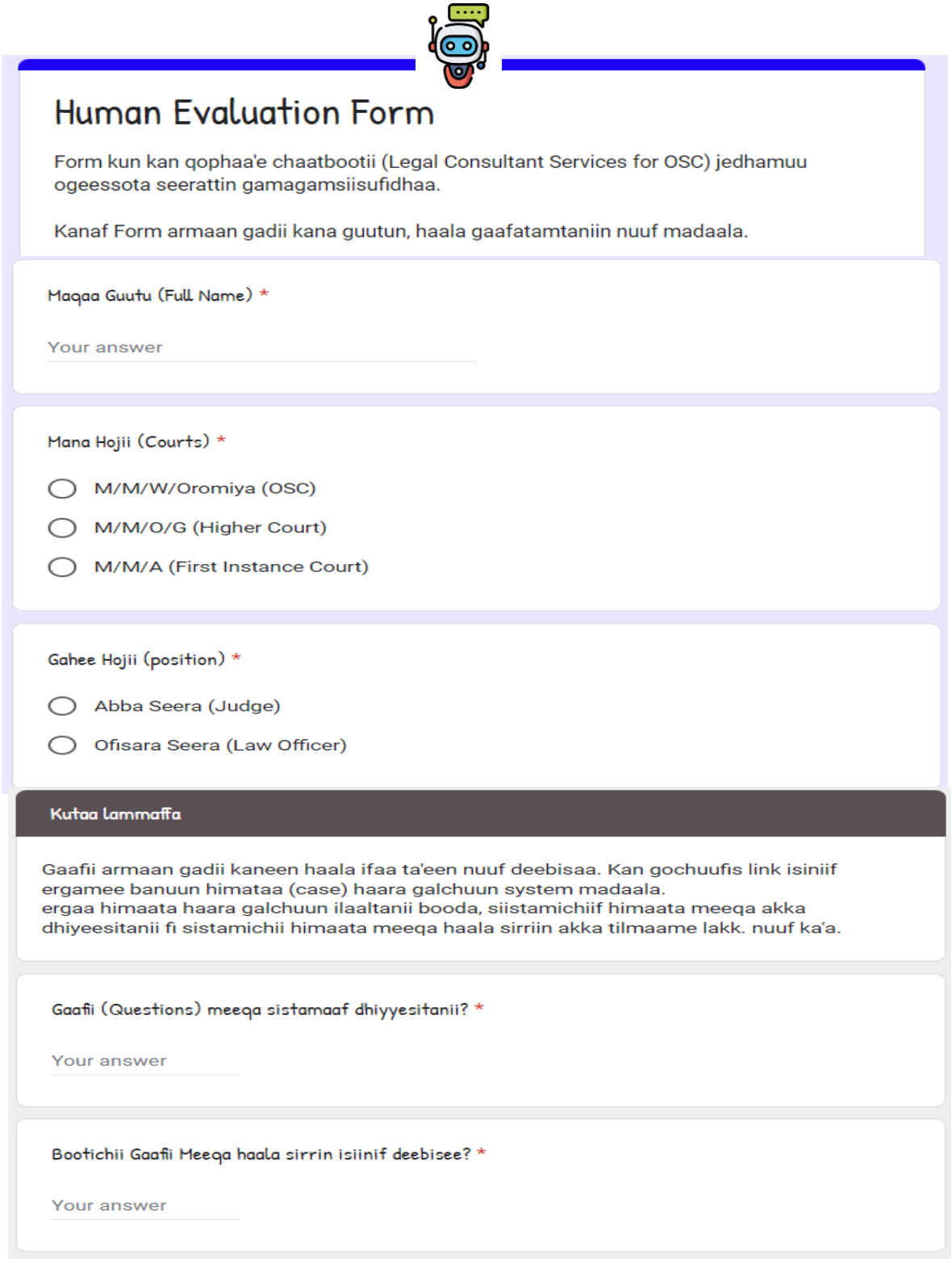


Figure 4: Form prepared for human evaluation