FIRST AID ASSISTANCE CHATBOT

Tribhuvan University

Institute of Science and Technology

A Final Year Project Submission in

Partial Fulfillment of the Requirement for the Degree of

Bachelor of Science in Computer Science and Information Technology

Under the Supervision of

Ashok Poudel

Submitted by

Aryaman Shrestha (Symbol No. 24269/076)

Darshana Pokhrel (Symbol No. 24274/076)

Shakila Gurung (Symbol No. 24304/076)

Submitted to

TRINITY INTERNATIONAL COLLEGE

Department of Computer Science and Information Technology

Dillibazar Height, Kathmandu, Nepal

January,2024

ABSTRACT

"FIRST AID ASSISTANCE CHATBOT" is a conversational tool that focuses on text and is designed to provide quick support for first aid queries. Driven by neural networks, this self-service chatbot offers a smooth user experience in an efficient manner and specialists in providing accurate first aid advice.

Our retrieval-based chatbot is made with the convenience of the user in mind, eliminating the need for in-person visits to obtain vital information. Users can simply ask appropriate inquiries and acquire general first aid knowledge using an easy-to-use web-based chat platform. Accessibility is given priority in this application, so users can quickly obtain essential information and contribute to a safer and more educated community.

The main goal of the “FIRST AID ASSISTANCE CHATBOT” is to apply modern machine learning techniques to train neural networks on a specific dataset to reliably predict user intent. The chatbot was created with the Python programming language and incorporates Django web framework, Keras Library, and the Natural Language Toolkit (NLTK). The result is a solid, closed-domain chatbot made to offer consumers dependable first aid support via an easy-to-use web-based chat interface.

Keywords: *Neural Networks, First Aid, Python Programming Language, Retrieval-based, Closed-domain Chatbot, Machine Learning.*

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ANN	Artificial Neural Network
API	Application Programming Interface
CSS	Cascading Style Sheet
DFD	Data Flow Diagram
JSON	JavaScript Object Notation
JSX	JavaScript XML
NLP	Natural Language Processing
NLTK	Natural Language ToolKit
NLU	Natural Language Understanding
NPM	Node Package Manager
ReLU	Rectified Linear Unit
SGD	Stochastic Gradient Decent
SVG	Scalable Vector Graphics
XML	Extensible Markup Language

CHAPTER 1

INTRODUCTION

1.1 Introduction

When it comes to medical situations, timing is critical. However, handling stressful situations frequently calls for skills that medical professionals have access to, leaving people unprepared and feeling pressured. By providing easily accessible, machine learning-based guidance, the First Aid Assistance Chatbot acts as a link between knowledge as well as action, enabling people to effectively handle common medical emergencies.

The days of searching the internet or browsing through instructions in a panic are outdated. With thorough training on available medical data, this innovative chatbot uses artificial intelligence to deliver precise and quick first aid support. Whether handling a little cut, an unexpected fever, or a choking emergency, the First Aid Assistance Chatbot is prepared to give users clear and precise instructions at each crucial stage.

In simple terms, this chatbot reduces the worry that comes with medical emergencies by working as a reliable companion. It gives users confidence to react properly using data-driven algorithms in a simple way, guaranteeing that the right actions are executed in the quickest way possible. The days of being unsure and afraid when faced with medical emergencies have ended; in their place is a dynamic tool that provides people with the information and direction they need to deal with frequent health problems.

The calm and systematic way in which the chatbot delivers first aid advice turns into a virtual training ground for users during times of high stress. It not only takes care of the immediate issue but also serves as a foundation for developing a fundamental knowledge of first aid concepts. Acting as a virtual coach, this first aid chatbot highlights the value of speedy replies, following the right protocols, and being well-informed while dealing with medical emergencies. Through this engaging learning process, users can gain the information and confidence necessary to make quick decisions in emergency situations by understanding the medical responses.

1.2 Problem Statement

In a world where accidents and medical emergencies can occur at any time and in any location, there is a crucial need for rapid and precise first aid help. Prompt and effective first aid can frequently mean the difference between saving a life and worsening a medical condition. However, not everyone is fully trained in first aid skills, and anxiety and confusion can make it difficult to provide prompt treatment during an emergency. Some of the consequences that the people face is such crucial situations are listed below:

- In high-stress situations, individuals may experience anxiety and panic, making it difficult to recall first aid techniques and act promptly.
- People might not be aware of the correct steps to take when a medical emergency happens, which can lead to delays in providing essential first aid.

1.3 Objectives

The objectives for this project are:

- To make essential first aid information readily available to a wide range of users, including those with limited or no formal first aid training.
- To ensure that the First Aid Chatbot offers precise, and evidence-based first aid guidance in accordance with established medical protocols.

1.4 Scope and Limitation

The scope of the "First Aid Assistance Chatbot" project is to make first aid knowledge easily accessible via a user-friendly web chat interface. Using machine learning algorithms, it covers a range of medical scenarios, guaranteeing flexibility and continuous advancement. The project's educational aspect acts as a virtual coach, teaching core first aid skills. Its primary goal is to promote positive well-being and health awareness to help the community.

Even if the "First Aid Assistance Chatbot" project is a useful application, it has some drawbacks:

- No NLU features are integrated into the chatbot; it is trained only with NLP approaches.

- Only accessible and limited data was used to train the chatbot. As a result, the chatbot might not function properly on extremely complicated or complex medical scenarios or outside of its taught sets.
- The chatbot displays an error message when a spelling error occurs in the user question since it is unable to process the misspelt word or sentence.
- Only English-language user queries are understood by the chatbot.

1.5 Report Organization

The format of the report is as follows:

In Chapter 1, the project's introduction is covered. It is explained what the project is, and its primary goal, which is expected to be accomplished after its completion, its scope, and its limitations.

The background research and the literature review of the publications studied for the project are included in Chapter 2.

Chapter 3 covers system analysis, which includes requirement analysis, feasibility analysis, and system analysis. Requirement analysis consists of both functional and non-functional needs, whereas feasibility includes all technical, economic, operational, and schedule feasibility.

The designing phase is covered in Chapter 4. The database design, interface design, and process design that go into creating this application are all parts of system design.

The focus of Chapter 5 is testing and implementation. The tools and algorithms used to develop the application are covered in this chapter, which also focuses on testing the application with various test case types.

The conclusion and future improvements are covered in Chapter 6, which offers suggestions on what we accomplished at the project's finish and how we might improve the application going forward.

CHAPTER 2

BACKGROUND STUDY AND LITERATURE REVIEW

2.1 Background Study

The idea for chatbots originated from a genuine concern for people's welfare, particularly during medical emergencies. Realizing the difficulties people encounter in getting access to emergency medical care, it seeks to be a trustworthy ally providing timely and precise first aid information. Chatbots step in to offer accessible support when physical access to healthcare is difficult, assisting users with crucial first aid procedures tailored to their circumstances. Chatbot aims to be a helpful and kind presence, whether it's answering questions about basic medical conditions or reacting to injuries. In a world where technological advancements are becoming more and more important in the healthcare industry, chatbots seem to be caring instruments that may equip people with the information they need to handle emergencies and improve the health of their communities.

Fundamentally, the project complies with to the concepts of machine learning and natural language processing (NLP), particularly as they relate to chatbot technology. The chatbot uses methods like tokenization and lemmatization for efficient language interpretation, which are supported by NLP theories. Additionally, the project integrates important ideas from database administration, where intentions, patterns, and replies are stored in JSON (JavaScript Object Notation) data format. Designing the database structure requires an understanding of ideas such as entity-relationship diagrams. The study also investigates the value of ER (Entity-Relationship) diagrams in illustrating the connections between various database entities.

The terminologies containing various theories and concepts, such as intent, patterns, and replies, serve as the foundation for the chatbot's creation and functionality. This thorough investigation lays the foundations for the project's effective implementation, ensuring a strong theoretical foundation for the practical application of chatbot technology.

2.2 Literature Review

Chatbots are computer programs that interact with users using natural languages. This technology started in the 1960's with the aim of seeing if chatbot systems could fool users that they were real humans. However, chatbot systems are not only built to mimic human conversation and entertain users. But they've come a long way since their inception in the 1960s. While early chatbots were primarily intended to explore the limits of artificial intelligence and simulate human communication, modern chatbot systems serve a variety of useful functions across a variety of businesses [1].

In recent years, there's been a renewed interest in natural language processing. Researchers are exploring empirical methods that use machine learning to automatically extract linguistic knowledge from large text collections. Unlike older manual approaches, these methods make NLP systems smarter and more adaptable by enhancing their understanding of language, meaning, and context. This shift has the potential to reshape the future of NLP, opening exciting new possibilities for research and applications [2].

The paper mentions that rise of human-computer speech interaction is evident through the increasing prevalence of speech-based search engines and virtual assistants like Siri, Google Chrome, and Cortana. This trend leverages Natural Language Processing (NLP) techniques, notably Python's NLTK, to analyze speech and develop chatbots capable of delivering human-like responses. These chatbots play a central role in enhancing human-computer interaction. To process transcribed speech data effectively, toolkits like NLTK are essential for organizing text into sentences and words. NLTK, an open-source Python library, plays a vital role in symbolic and statistical NLP, including part-of-speech tagging and text analysis [3].

The integration of natural language processing (NLP) with radiology is studied in the paper "Bag-of-Words Technique in Natural Language Processing: A Primer for Radiologists." To use machine learning (ML), it highlights the transformation of free-form text to a numerical representation. A focus on possible meaning breakdown begins with the introduction of the bag-of-words (BOW) technique. Applying strategies like Term Count, and Term Frequency–Inverse Document Frequency(TF-IDF) to change BOW characteristics is addressed in this paper. The study highlights the practical application of (NLP) in radiology, highlighting the significance of radiologists

understanding the methods' benefits and drawbacks so that they can work with data scientists in an effective way. Overall, it provides radiologists with a comprehensive manual for implementing NLP techniques in medical imaging [4].

The paper "A Primer on Neural Network Models for Natural Language Processing" provides a brief but detailed overview of the use of neural networks in NLP. The historical background, basic ideas of neural networks, and important architectures such as feedforward and recurrent models are covered in the paper. The advantages of using vectors rather than unique IDs to represent characteristics are also covered by the author. Additionally, it is discussed how the dense vector representation is preferable to the one hot representation. It provides a well-balanced combination of theoretical ideas and real-world applications in neural network models, making it an invaluable tool for individuals just starting out in the field of natural language processing. When it comes to learning about the basic concepts and applications of neural networks in NLP, this paper is a great resource [5].

The paper "Text Classification Using Artificial Neural Networks" discusses the task of text classification, which is becoming more and more important. It highlights the usefulness of text categorization in a variety of applications, including data mining, sentiment analysis, web searching, spam filtering, and judgement. The research focuses on the use of Artificial Neural Networks (ANNs) for this purpose, highlighting the need for automated approaches due to the massive amount of textual data. The document has been divided into multiple sections that highlights the essential actions that must be performed when classifying texts, such as: Preprocessing that includes analyzing each term and document in the corpus, defining the neural networks that takes input values and weights from the input layer as input and then it goes to the hidden layer in which a function sums the weights and maps the results to the corresponding output layer units, and neural network training where the neural network is trained using forward and backward propagations. Overall, the paper provides a comprehensive overview, setting the stage for the subsequent examination of the ANN technique's application in text classification [6].

The paper, "A Comprehensive Survey of Deep Learning Models Based on Keras Framework", offers a detailed examination of deep learning models within the context of the Keras framework. The paper aims to provide an overall review of Keras based on

several factors like accuracy, complexity, mean absolute validation error, and so on. In this paper, researchers conducted a comprehensive review of the latest and most efficient methods of the Keras library in various fields, and the details of this method, such as objectives, technique/tool, datasets, issues, important findings, and accuracy, are summarized. The author also explains the different types of Keras model that can be used in neural networks. One of them is the Sequential Model in Keras, which enables the development of models layer by layer in a sequential sequence. In addition, the in-built functional APIs in Keras Library also provides more options for defining a model and adding layers. Lastly, the author also discusses about the advantages of using TensorFlow and Keras and suggest the readers to select appropriate library for the model [7].

The paper titled "Comparison of Optimization Techniques Based on Gradient Descent Algorithm: A Review" conducts an in-depth examination of optimization techniques, specifically focusing on the Gradient Descent (GD) algorithm and its variants. The paper introduces the Gradient Descent algorithm as a popular optimization tool, especially in the context of solving unconstrained optimization problems. It introduces us with the Gradient Decent Algorithm variants like Batch Gradient Descent (BGD), Stochastic Gradient Descent (SGD), and Mini-batch GD. Furthermore, the authors discuss challenges associated with these algorithms and provide an extensive overview of widely used optimization algorithms such as Learning Rate, Momentum, Nesterov Momentum, Adam, and so on. The paper also presents a comprehensive comparison of the optimization algorithms based on GD, considering aspects such as training speed, convergence rate, performance, and the associated pros and cons [8].

CHAPTER 3

SYSTEM ANALYSIS

3.1 System Analysis

3.1.1 Requirement Analysis

The system's requirements are divided into functional and non-functional categories, with the goal of optimizing the development process and maintaining structure. This technique ensures that the project stays on track to meet its objectives.

3.1.1.1 Functional Requirements

The system's functional requirements specify how the features, functionalities, and interactions of the system must work. They act as a guide for putting the required features and procedures in place as well as a design for how the system should behave.

The functional requirement of the project is shown in the use case diagram below:

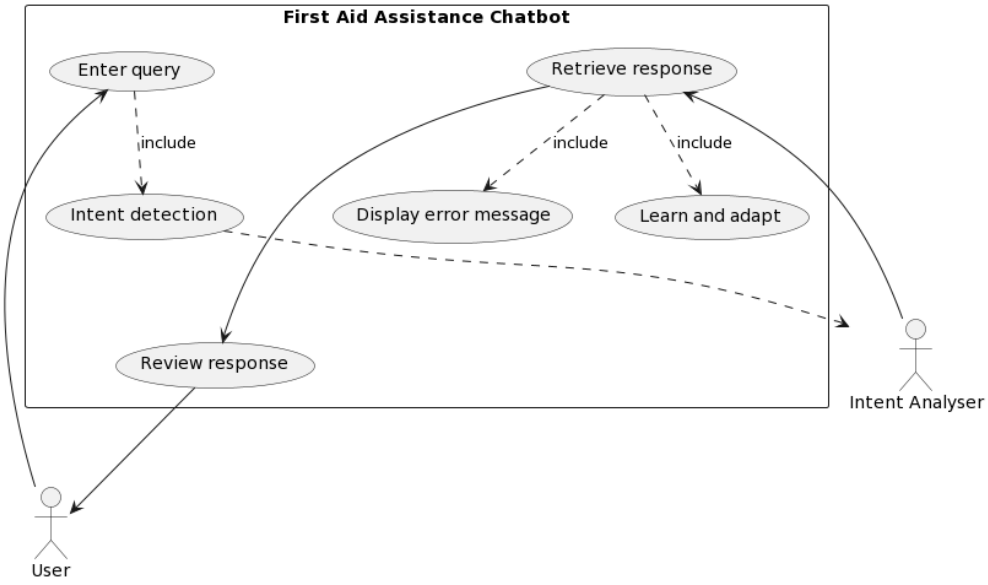

Figure 1: Use Case Diagram of First Aid Assistance Chatbot

The use case diagram in the image above demonstrates the modules that construct the workflow of a First Aid Assistance Chatbot. Users type queries, and the chatbot identifies intent, gets responses, and displays error messages as needed. The system is designed to adapt and improve over time as it learns from the various user questions. Users can examine and interact with responses, creating an interactive and user-friendly experience.

3.1.1.2 Non-functional Requirements

The non-functional requirements of the system are:

a. **Functionality:** The system should effectively perform tasks, such as accurately identifying user intent and providing relevant first aid information.
b. **Usability:** The system should feature a user-friendly interface, ensuring users can easily interact with and navigate the chatbot.
c. **Reliability:** The system should work consistently and properly, building confidence in its ability to offer reliable information and responses.
d. **Performance:** The system should exhibit efficiency and responsiveness, ensuring quick and effective handling of user queries without significant delays.

3.1.2 Feasibility Analysis

Before moving further with a project, a feasibility study is conducted to determine how feasible it is. Once the feasibility assessment is complete, the project proceeds to its subsequent stages, initiating the implementation and development phases. Some of the feasibility studies that we performed are given below:

3.1.2.1 Technical Feasibility

Our First Aid Assistance Chatbot project's technological feasibility is supported by modern NLP and machine learning technologies, which provide correct question interpretation and continuous learning. The project takes advantage of widely available programming tools and frameworks, resulting in an effective and scalable solution that is compatible with a variety of systems.

3.1.2.2 Operational Feasibility

Our project's operational feasibility is ensured by its user-friendly design, which makes it accessible to all users. The chatbot's user-friendly UI makes it useful during an emergency. The system's general design is consistent with the purpose of delivering effective and accessible first-aid assistance.

3.1.2.3 Economic Feasibility

The project is economically feasible as it can be operated in any web browser, eliminating the need for specialized software or hardware, reducing associated costs. In addition, the use of open-source software minimizes the cost of operation of the project.

3.1.2.4 Schedule Feasibility

This project was on schedule because it was a web-based approach that was suitable in terms of timing and was carried out at a reasonable hour. The Gannt Chart that shows the estimated schedule of our project is shown below:

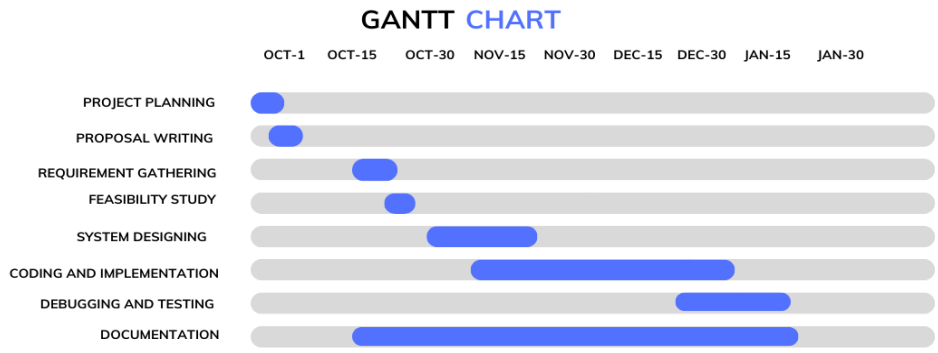

Figure 2: Gantt Chart for the System

3.1.3 Structuring System Requirements

3.1.3.1 Data Modeling

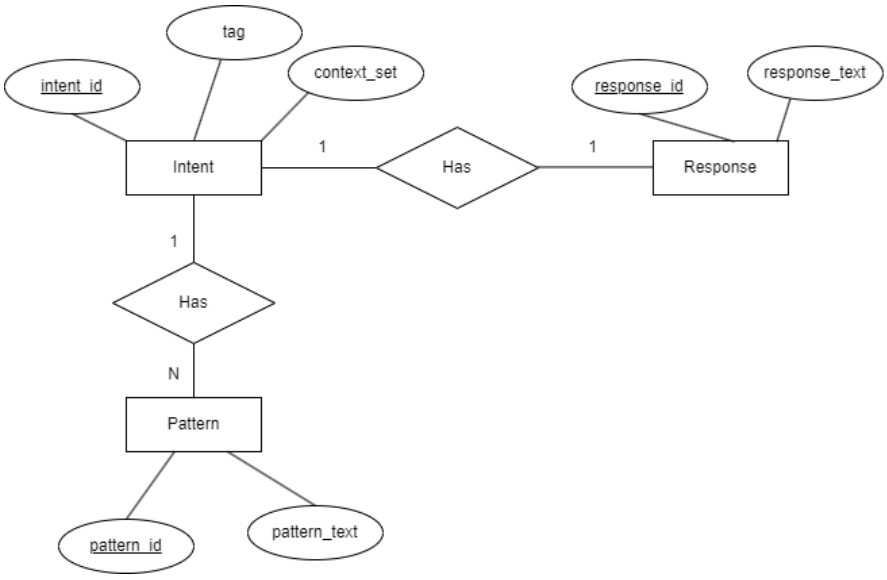

Figure 3: Entity Relationship Diagram

The ER diagram above highlights the relationships between several components of our system. Every individual "Intent" is distinguished from others by its "intent_id" and is connected to a "tag" and a "context_set." This "Intent" is associated with several "Patterns," which are each distinguished by a "pattern_text" and "pattern_id," signifying

the various input formats the chatbot can identify. Additionally, the diagram indicates that the "Intent" is connected to a "Response," identified by a "response_id" and coupled with "response_text," showing the system's capabilities to generate automated responses based on recognized user input patterns.

3.1.3.2 Process Modeling

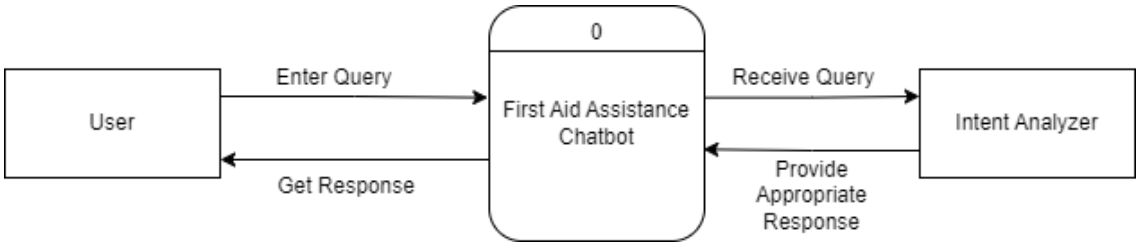

Figure 4: Level-0 DFD for the First Aid Assistance Chatbot

The level 0 DFD above shows the flow of information in a First Aid Assistance Chatbot system. The user enters a query about a medical issue, such as a cut or a burn, and the chatbot receives it. The chatbot then sends the query to an Intent Analyzer, which is a process that analyzes the query and determines the user's intent and the appropriate response. The Intent Analyzer then provides the response back to the chatbot, which delivers it to the user.

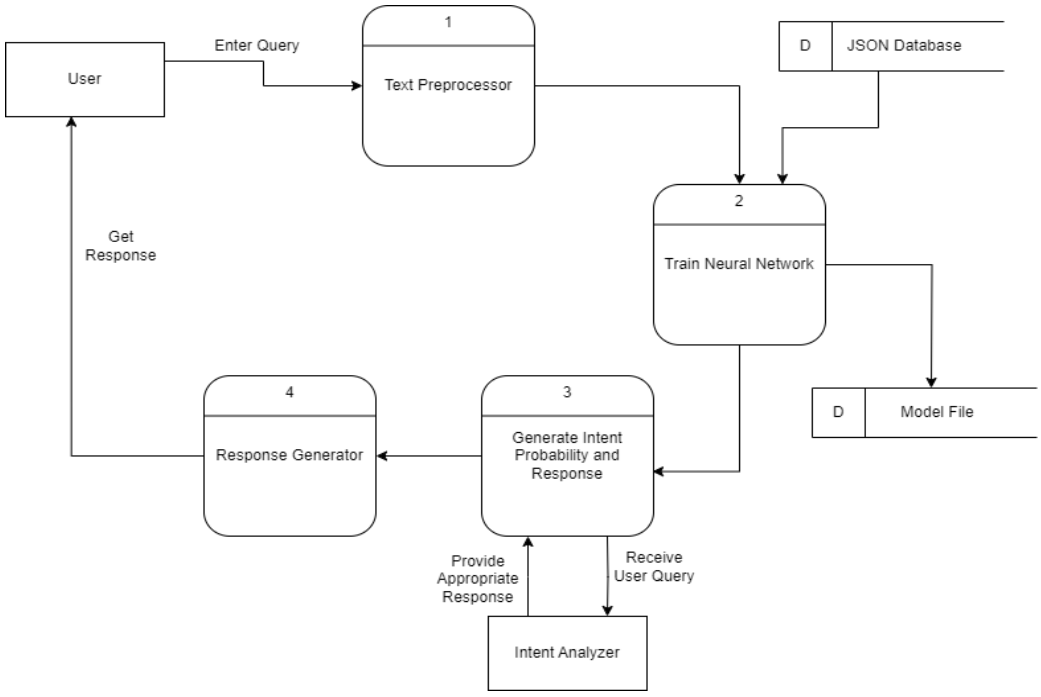

Figure 5: Level-1 DFD for the First Aid Assistance Chatbot

The above Level 1 Data Flow Diagram illustrates the information flow in a user query processing system. The User inputs queries, which are cleaned and formatted by the Text Preprocessor. The Train Neural Network process trains a model using data from the JSON database, and the Intent Analyzer determines user intent. The Response Generator then provides an appropriate response. This representation highlights the essential components involved in the system's operation.

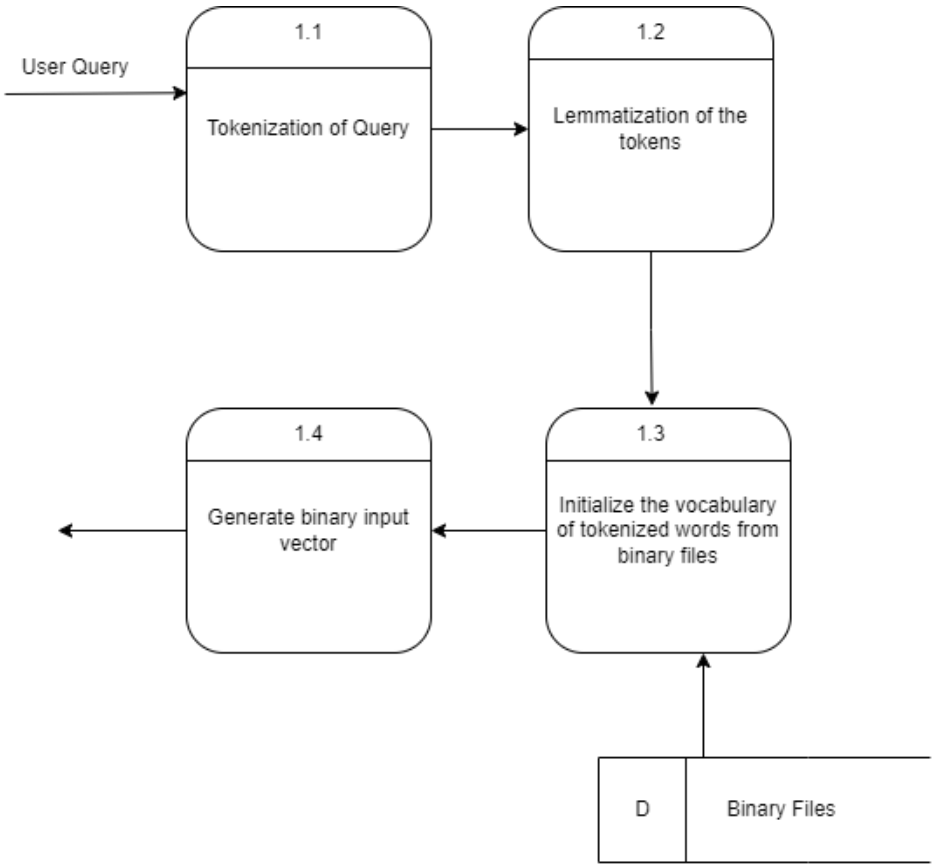

Figure 6: Level-2 DFD decomposing the Text Preprocessor

The level 2 DFD shown above outlines the text preprocessing stage of the system. The query received from user is tokenized and lemmatized, which breaks it down into words and reduces them to their base forms. The system populates its vocabulary with binary files and creates a binary input vector that represents word existence or absence. For example, if the vocabulary contains the words "cut", "burn", "fever", and "bleeding", the binary vector [1, 0, 1, 0] represents the presence of "cut" and "fever" while the lack of "burn" and "bleeding".

CHAPTER 4

SYSTEM DESIGN

4.1 System Design

System design refers to the process of defining the architecture, components, interfaces, and functionality of a software system. It involves creating a blueprint or plan that guides the development and implementation of the system.

4.1.1 Flowchart of the Model

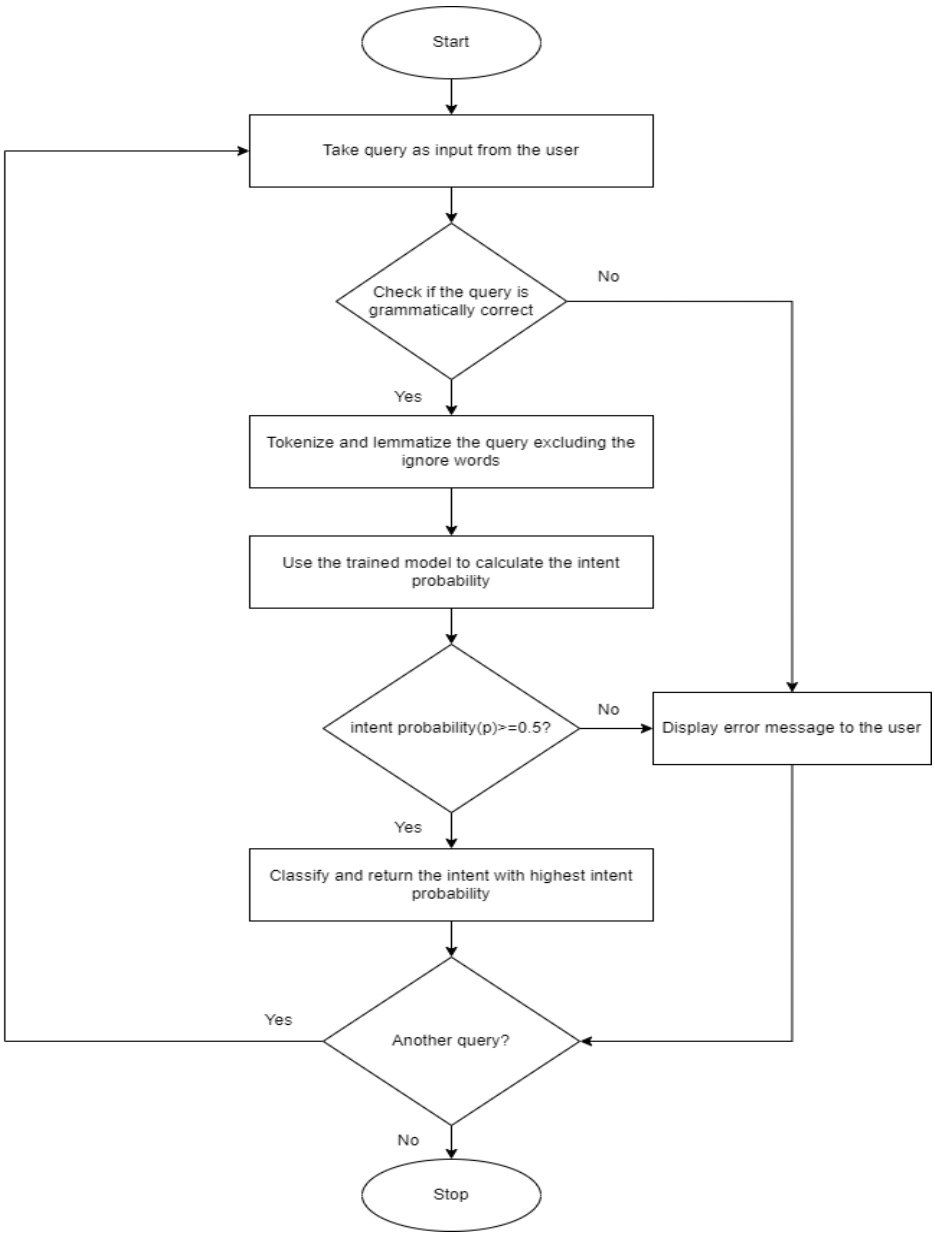

Figure 7: Flowchart of the System

The above figure outlines the workflow of the system that takes a query from a user as input and then tries to respond to it. If the query is grammatically correct, the system uses a trained model to calculate the intent of the query. If the intent probability is greater than or equal to 0.5, the system classifies and returns the intent with the highest probability. Otherwise, it displays an error message to the user.

4.1.2 Database Schema Design

The main format for exchanging and storing information in our chatbot application is JSON (JavaScript Object Notation) data due to its wide array and ease of use. Because of its simple and clear syntax, JSON is a great option for storing structured data, facilitating smooth communication between the various parts of the chatbot system.

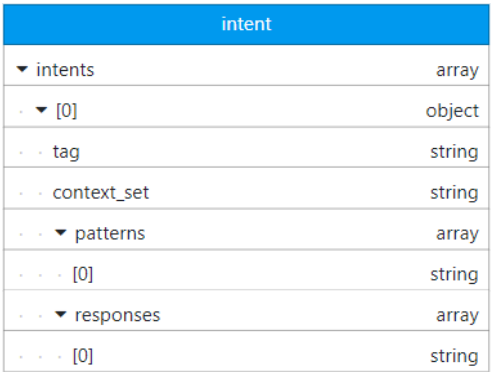

Figure 8: Schema Design of the Database

The above figure illustrates the JSON database's schema diagram. It consists of an array of “intents” each containing objects with specific properties like “tag” “context_set” “patterns” and “responses” The “patterns” are the array of strings that the chatbot recognizes, and the “responses” are the array of strings that the chatbot replies with when a pattern is matched.

4.1.3 Interface Design

To give users a simple and convenient way to communicate with our system, we have implemented a web-based chatbot interface for our project. The user interface consists of a straightforward messaging structure where users can submit queries, with all their interactions with the chatbot being displayed above the query field. React.js is used throughout the creation of the user interface.

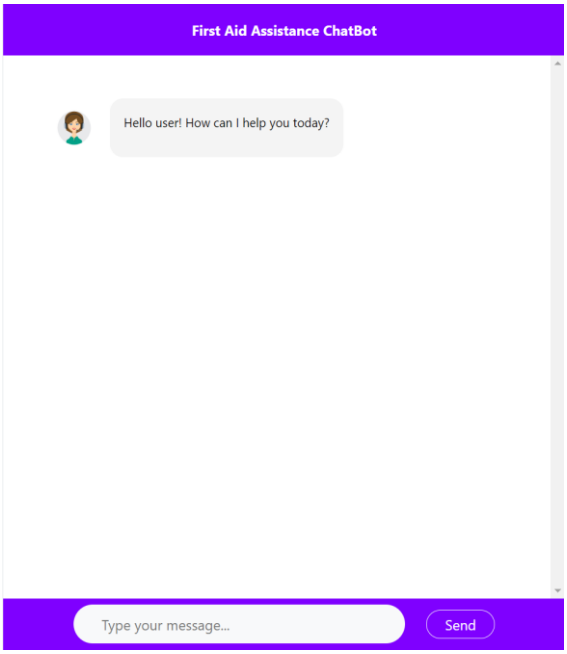

Figure 9: Interface Design for the Chatbot

4.1.4 System Architecture

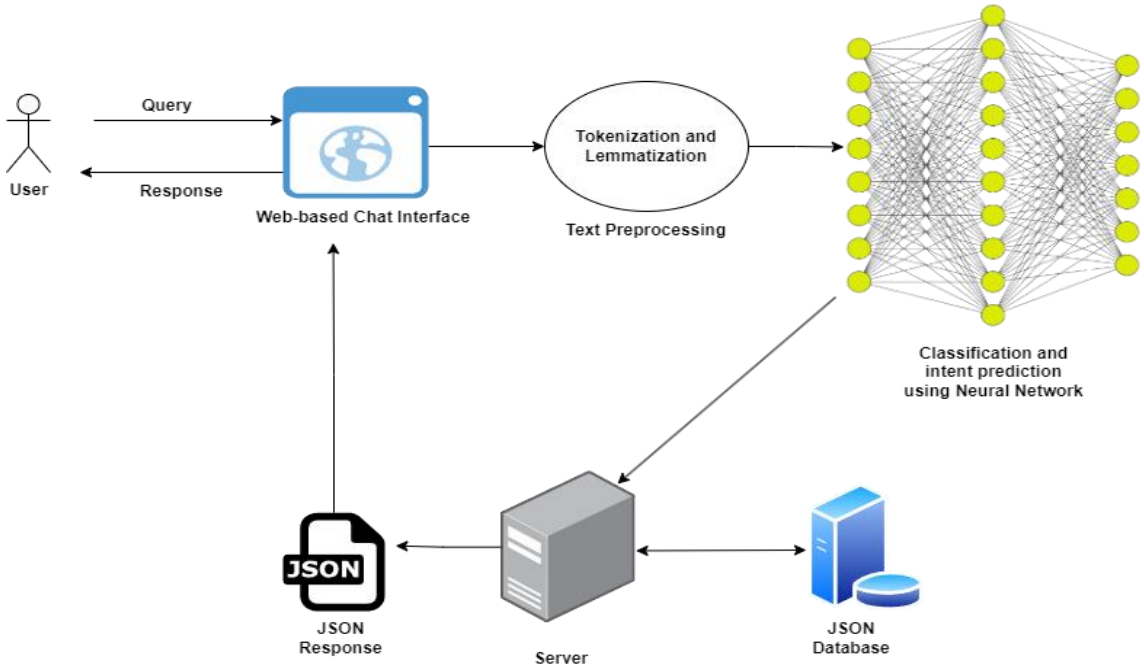

Figure 10: System Architecture of First Aid Assistance Chatbot

The figure above shows the system architecture for the First Aid Assistance Chatbot. The system architecture consists of following components:

I. Web-based Chat Interface

The visible part of the system where users interact with the chatbot in real time is the Web-based Chat Interface. Users can ask questions and get answers in the comfortable environment of a web browser. Beneath what seems to be simplicity is a complex system that uses machine learning and natural language processing to understand user inquiries and provide clear, appropriate answers.

II. Text Preprocessing

To guarantee the best possible detection by the system, the user's text is preprocessed in this essential component of the system. Tokenization and lemmatization are the two steps we use to accomplish this. To enable an in-depth look of the text, tokenization involves dividing user input into individual words or tokens. Lemmatization is used to enhance the understanding of the words by reducing them to their base or root form after tokenization.

For instance, let's consider the sentence: "The quick brown foxes are jumping over the lazy dogs." The text is then broken down into individual words using tokenization, which produces a list of tokens like ["The", "quick", "brown", "foxes", "are", "jumping", "over", "the", "lazy", "dogs", "."]. We then use lemmatization to get a more simplified representation by breaking down words into their root or base form. ["The", "quick", "brown", "fox", "be", "jump", "over", "the", "lazy", "dog", "."] is the lemmatized version in this instance.

III. Neural Networks

Following tokenization and lemmatization, a neural network classifies the processed words by using a pretrained model on relevant data. This model, which was developed using machine learning techniques, uses the patterns and context it learnt during training to assign probabilities to different intents. Finding the user's underlying intention or query in their input requires performing an intent probability calculation, which is an essential step. Using pre-trained neural networks, our system can effectively interpret tokenized and lemmatized user input, providing an accurate understanding of the user's intention.

IV. JSON Database

To effectively store and handle structured data for our system, we employ a JSON database. JSON (JavaScript Object Notation) is the fundamental format for describing information in databases, giving a flexible and human-readable approach to organize data. In our system, we utilize the "intents.json" file, which comprises an array of "intents," each containing objects with specific properties such as "tag," "context_set," "patterns," and "responses." Within each intent, the "patterns" represent an array of strings that the chatbot recognizes as user inputs, while the "responses" constitute an array of strings that the chatbot generates as replies when a matching pattern is identified.

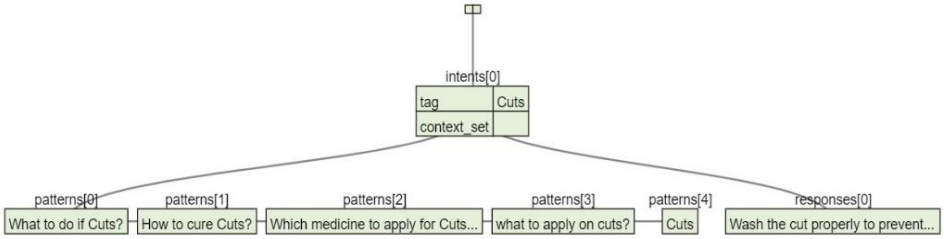

Figure 11: Data Organization in JSON Database

In the "intents.json" file, the data for a particular tag called "Cuts," is shown in the figure above. This specific tag has an object with important attributes such as "tag" for identification, "patterns" for user input string representation, and "responses" for a collection of pre-written chatbot responses. When users question about or refer to issues linked to cuts, the chatbot can efficiently retrieve and utilize patterns and responses related to the injury “Cuts”.

V. Web Server

To provide smooth communication between the chatbot interface and the underlying system components, the web server serves as an API. Acting as a link between the user and the required modules, like the database and the natural language processing engine, the web server takes user queries from the chatbot interface and handles them. The API allows data to be transmitted, including user queries and system responses, over well-defined endpoints, promoting dynamic and responsive interaction.

CHAPTER 5

IMPLEMENTATION AND TESTING

5.1 Implementation

One of the most important phases in the software development life cycle is system implementation. This is the most expensive and time-consuming step of SDLC. The implementation's goal is to develop and produce a system element that complies with the requirements for that element's design properties. The success of the entire software development project is dependent on the performance of this phase, as any flaws or deviations from the design properties can have a major impact for the functionality, reliability, and user satisfaction of the final product.

5.1.1 Tools Used

These are some of the tools used while creating the first aid assistance chatbot:

5.1.1.1 Diagram Tools

a) **PlantUML**

PlantUML is an open-source tool that allows developers to create diagrams and visual representations of software systems using a simple and human-readable syntax. For our project, the use case diagram used in the analysis stage was made using PlantUML.

b) **Draw.io**

Draw.io is a versatile and user-friendly online diagramming tool that enables the creation of various types of diagrams, including flowcharts, wireframes, and network diagrams, through an intuitive web-based interface. Draw.io was used for the creation of flowcharts of the model, system architecture, DFD, and ER diagrams.

c) **NNSvg**

NNSvg is a tool for creating publication-ready neural network diagrams in SVG format. The neural network diagram for the system architecture was made using this.

5.1.1.2 Frontend Tools

a) **React.js**

React Java Script Library was used in our project to build an interactive chatbot component.

b) **JavaScript XML(JSX)**

JSX in React is used to structure the content of the web page. All the elements in the chat interface are structured using JSX.

c) **CSS**

Cascading Style Sheets (CSS) are used to style and format the appearance of web pages. The final web pages' layout and design were maintained through the usage of CSS. It was employed to style the JSX elements—fonts, colors, and spacing—to guarantee a uniform and visually attractive user interface.

d) **Node.js and NPM**

Node.js is a JavaScript runtime that allows you to run JavaScript on the server side. We used NPM(Node Package Manager) for installing and managing libraries and dependencies, including React and its associated tools to make a request to the server side.

5.1.1.3 Backend Tools

a) **Python**

The main programming language used to create the chatbot's code was Python. Python is a great option for creating chatbots since it offers a large selection of libraries and frameworks for machine learning and natural language processing. Python served as our primary programming language of choice for developing, testing, and training our chatbot model.

b) **Django Framework**

The online application was built using Django, a Python backend web framework. We applied Django to provide API (Application Programming Interface) endpoints for the Python functions in our chatbot. This allowed the endpoints to be accessed from the front end, enabling the chatbot to respond to queries efficiently.

c) Natural Language ToolKit(NLTK)

The Natural Language Toolkit is a highly effective NLP library that includes packages designed to enable machines to understand human language and provide suitable responses. To preprocess the text, we utilized Wordnet and Punkt Word Tokenizer, both of which are included in NLTK. While Wordnet is a large lexical English database that we used to lemmatize each individual token, Punkt tokenizes the entire phrase into individual words or tokens.

d) Keras Framework

Keras is an open-source library that provides a Python interface for artificial neural networks. We used Keras tools and features to simulate the behavior of artificial neural networks and classified the text from our database using the neural networks.

5.1.2 Implementation Details of Algorithm

5.1.2.1 Bag of Words with One Hot Encoding

A key component of the text processing module in our project is the "Bag of Words with One Hot Encoding" method. Understanding and describing textual data is essential to natural language processing, and this method offers a practical means of transforming unformatted text into one that is appropriate for machine learning models. Using this method, each input pattern is transformed into a binary vector representing the presence or absence of words, and the output is encoded in a one-hot format, preparing the data for training a machine learning model to recognize intents based on textual input.

This method is implemented in our project using well-designed classes and functions. To generate a standardized input format for our natural language processing engine, we utilize tokenization, vocabulary development, and encoding user queries. This technique is critical in allowing our chatbot to understand and respond to a wide range of user inputs, hence greatly enhancing the system's overall efficiency.

Let us consider the following example:

User Query: "How should I treat a burn at home?"

Tokenization: ["How", "should", "I", "treat", "a", "burn", "at", "home"]

Vocabulary: ["How", "should", "I", "treat", "a", "burn", "at", "home", "Cuts", "Fever"]

Bag of Words: [1, 1, 1, 1, 1, 1, 1, 1, 0, 0]

An example of a question a user can enter here is "How should I treat a burn at home?" Tokenization is the procedure that breaks down this question into individual terms such as "How," "should," "I," "treat," "a," "burn," "at," and "home." The predetermined vocabulary that our system has, which includes relevant terms like "Cuts" and "Fever," is useful for analyzing scenarios related to first aid. Then, this query is converted into a binary vector using the "Bag of Words with One Hot Encoding" approach. Each element of the binary vector indicates whether a certain word from the vocabulary is present (1) or absent (0). The resulting binary vector in this case is [1, 1, 1, 1, 1, 1, 1, 0, 0], which successfully captures the key elements of the input provided by the user in a format that can be used to train our machine learning model to identify intent from textual input.

5.1.2.2 Text Classification using ANN

We use the effective approach for text classification utilizing Artificial Neural Networks (ANN) in our First Aid Assistance Chatbot project. This innovative approach is the foundation for our chatbot's capacity to recognize and classify user queries related to first aid on its own. By means of methodical training on a dataset that is specially selected for first aid scenarios, the neural network effectively acquires complex patterns and associations among words.

The neural network is made up of three layers: input, hidden layer, and output. Each node in the input layer corresponds to an input feature. The hidden layer is made up of neurons that analyze incoming data via weighted connections and activation functions, capturing complex patterns and associations. Finally, the output layer generates the model's predictions or classifications based on the information processed by the hidden layers. The visualization of the ANN model is shown in the figure below:

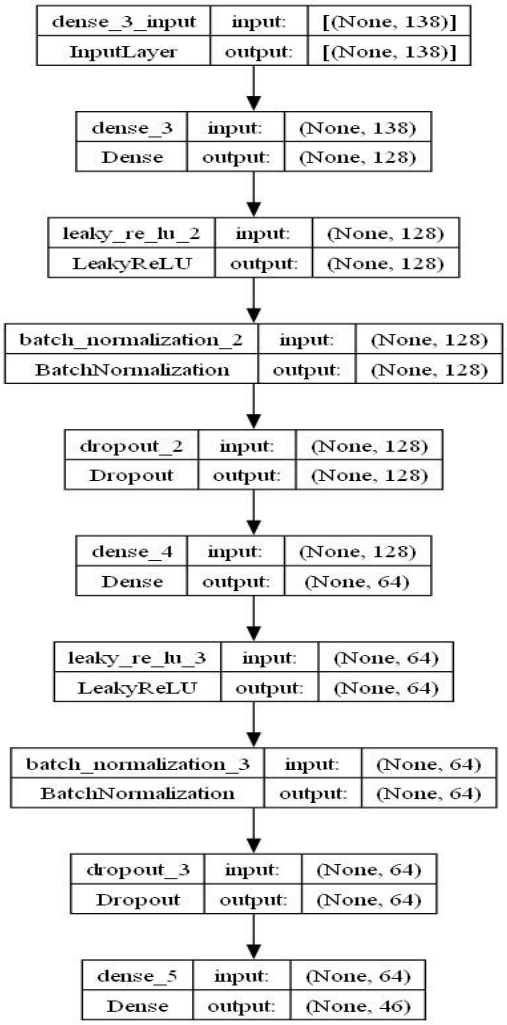

Figure 12: Visualization of the ANN Model

The ANN model that we used for our project is shown in this figure. It is made up of the following layers:

A. Input Layer

The input layer in our model is made up of dense 128 neurons, each of which is linked to an input feature with a size of 138, and these neurons work together to serve as the initial processing stage for incoming data. The ReLU activation function is used in the input layer to account for instability. We included LeakyReLU, Batch Normalization, and Dropout to improve performance. LeakyReLU prevents "dying ReLU," Batch Normalization stabilizes training, and Dropout serves as regularization, all of which improve the model's stability. The detailed explanation is included below:

i. Leaky ReLU

We used Leaky ReLU activation (alpha=0.01) in our neural network architecture to overcome the "dying ReLU" issue. This change ensures a small, non-zero gradient

for all inactive units, which promotes better information flow and increases model robustness.

ii. **Batch Normalization**

The Batch Normalization in the layer normalizes the input of each unit to have zero mean and unit variance. This normalization procedure improves training stability by preventing the occurrence of high values that might hamper the neural network's convergence during the training phase. Furthermore, Batch Normalization accelerates convergence by ensuring a uniform scale of input values across mini batches, resulting in more stable and efficient learning processes throughout the training process.

iii. **Dropout Layer**

In our model, we use a Dropout layer with a 0.5 dropout rate, which is a regularization approach that randomly deactivates some of the input units during training. This purposeful dropout strengthens the model, minimizing overfitting by increasing variability and encouraging a more generalized learning strategy.

B. Hidden Layer

We use an assembly of dense 64 units of neurons for the hidden layer. This layer captures complex patterns and relationships in the input data, serving as an intermediate processing step. Together, these 64 neurons aid in the transformation of incoming data, enabling the neural network to extract useful characteristics and representations. To improve the model's ability to understand and extract information from the input data, the hidden layer is essential to the neural network architecture's overall efficiency.

Furthermore, we apply the same Dropout, Batch Normalization, and LeakyReLU activation methods to the hidden layer. By using LeakyReLU consistently, the "dying ReLU" problem is reduced, and the hidden layer's robustness is increased. Input values are normalized using batch normalization, which encourages training stability and effective convergence. The Dropout mechanism—which has a 0.5 dropout rate—continues to operate, randomly deactivating units to safeguard against overfitting and guarantee a more broadly based learning strategy in the hidden layer.

C. Output Layer

The output layer consists of 46 neurons with the SoftMax activation function. This architecture is suited to our model's specific requirements, with each neuron representing a separate class, and the SoftMax activation function helping the development of probability distributions across these classes. The output layer, which has 46 neurons, is set up to provide a probability distribution over the various classes, allowing the model to make precise and confident predictions based on the patterns acquired throughout the training process.

5.1.2.3 Optimization using Stochastic Gradient Decent(SGD)

To minimize the model's loss function, we use Stochastic Gradient Descent (SGD) as an optimization procedure. SGD is a fundamental and frequently used approach for iteratively altering the weights of a neural network during training. This strategy includes randomization, which helps the optimization process avoid local minima and achieve more efficient convergence. With a carefully estimated learning rate of 0.005, momentum of 0.9, Nesterov momentum, and weight decay set to 1e-6, our SGD-based optimization maintains a precise balance between convergence speed and stability, resulting in successful training and performance of our neural network model.

REFERENCES

[1] E. Atwell and B. A. Shawar, "Chatbots: Are they Really Useful?," *LDV-Forum ,* vol. 22, no. 1, pp. 29-49, 2007.

[2] R. J. Mooney and E. Brill, "An Overview of Empirical Natural Language Processing," *AI Magazine ,* vol. 18, no. 4, pp. 13-24, 1997.

[3] J. Woods and A. S. Abdul-Kader, "Survey on Chatbot Design Techniques in Speech Conversation Systems," *International Journal of Advanced Computer Science and Applications,* vol. 6, no. 7, pp. 72-80, 2015.

[4] K. Juluru, H.-H. Shih, K. K. Murthy and P. Elnajjar, "Bag-of-Words Technique in Natural Language Processing: A Primer for Radiologists," *RadioGraphics ,* vol. 41, pp. 1420-1426, 2021.

[5] Y. Goldberg, "A Primer on Neural Network Models for Natural Language Processing," *Journal of Artificial Intelligence Research,* vol. 57, p. 345–420, 2016.

[6] P. Prasanna and D. Rao, "Text classification using artificial neural networks," *International Journal of Engineering & Technology,* vol. 7, no. 1.1, pp. 603-606, 2018.

[7] B. A. Sallow and B. T. Chicho, "A Comprehensive Survey of Deep Learning Models Based on Keras Framework," *Journal of Soft Computing and Data Mining,* vol. 2, no. 2, pp. 49-62, 2021.

[8] A. M. Abdulazeez and S. H. Haji, "Comparison of Optimization Techniques Based on Gradient Descent Algorithm: A Review," *Palarch's Journal Of Archaeology Of Egypt/Egyptology,* vol. 18, no. 4, pp. 2715-2743, 2021.