Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

*Digital Object Identifier 10.1109/ACCESS.2017.DOI*

# Advanced NLP models for Technical University Information Chatbots: Development and Comparative Analysis

**GIRIJA ATTIGERI, (Member, IEEE), ANKIT AGRAWAL, AND SUCHETA KOLEKAR(MEMBER, IEEE)**
Dept. of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Karnataka, India
Corresponding author: Sucheta Kolekar (e-mail: kolekar.sucheta@gmail.com).

**ABSTRACT** The process to obtain the information about university/institute is a critical stage in the academic journey of prospective students who are seeking information about the specific courses which makes that university/institute unique. This process begins with exploration to general information about universities through websites, rankings, and brochures from various sources. Most of the time, information available on different sources leads to discrepancies and influences student's decisions. By addressing inquiries promptly and providing valuable information, universities can guide individuals in making informed choices about their academic future. To address this, the chatbot application is the most effective tool to be implemented and make it functional on university's functional website. A chatbot is an artificially intelligent tool which can interact with humans and can mimic a conversation. This tool can be implemented using advanced Natural Language Processing (NLP) models to provide the pre-defined answers to the student's queries. Chat-bot is very helpful for query resolution during the counseling process of the institute as it will provide official/uniform information and can be accessed 24x7. Therefore, the aim of this research work was to implement a chat-bot using various NLP models and compare them to identify best one. In this work, five chat-bot models were implemented using neural networks, TF-IDF vectorization, sequential modeling and pattern matching. From the results, it was observed that neural network-related models had better accuracy than TF-IDF and pattern matching model, and sequential modeling is the most accurate model because it prevents over-fitting. Furthermore, a chat-bot having any kind of optimizer can improve the result and it is most important that pattern matching, and semantic analysis should be the parts of a chat-bot for real time scenarios.

**INDEX TERMS** conversational AI, Natural Language Processing, Artificial Intelligence, Chat-bots, Neural Networks, Sequential Modeling, Pattern Matching, Semantic Analysis

## I. INTRODUCTION

Chatbots are essential for counseling in engineering institutes for a number of reasons. The first benefit of these chatbots is that they make counseling services more accessible by eliminating time and location constraints and offering engineering students who could be facing difficulties in their personal, professional, or academic lives instant assistance. Due to chatbots' real-time functionality, kids may get help whenever it's convenient for them, which encourages a proactive approach to problem solving.

Chatbot is an artificially intelligent entity that can interact with humans and mimic conversations. The input to the chatbot could be text-based or spoken (voice-based queries). Chatbots are majorly used for information retrieval. It can run on a local computer or mobile phone, though most of the time, chatbots are accessed through the web browser. A chatbot mainly works by asking a question or query regarding a specific topic. They work on the principle of Artificial Intelligence (AI) and Natural Language Processing (NLP) to provide the answers to the user's queries, and a predefined knowledge base helps to develop a response to the question [1].

There are three major types of chatbots, namely the Rule-based, Retrieval-based model, and Generative-based model. In the Rule-based Model, the bot responds to queries using pre-programmed rules. This kind of chatbots can answer a simple and limited set of questions. Retrieval-based Models select an appropriate response from a group of pre-defined responses using limiting conditions(heuristic). The bot created can understand the entire conversation and respond based on the context of the conversation. Lastly, Generative-based models generate responses from previous and current experiences. This highly sophisticated chatbot type requires complex computational models and vast data to train [2]. The captured vast data of question and answers can be pre-trained to generate the model and subsequently the model will be able to generate the accurate responses. Usually, neural network based models such as Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) are more efficient to work on such a use-case.

The standard form of query will be asked in natural language format e.g. "English". To generate the appropriate responses, understanding and analysis of query is important and that can be done by Natural Language Processing (NLP).

As per natural language analysis of the queries, there are three major types of queries can be defined for experimentation:

- **Simple Query:** A simple query consists of a single and unconstrained query desire(desired output) and has a single unconstrained query input. For example, in the query "What is the capital of USA?", the desired output of the query (i.e., Capital) is explicit, single, and not bound to any constraint. The input to the query (i.e. USA) is also single and unconstrained.
- **Complex Query:** A complex query consists of a single query desire, which can be either constrained or unconstrained and the input to the query is multiple and explicit but it can be constrained or unconstrained. For example, in the query "What was the capital of the USA during World War II?", the query has multiple inputs (i.e. USA and World War II) and the desired output is single, unconstrained, and implicit (i.e. Capital).
- **Compound Queries:** A compound query is a query with a conjunction or dis-junction operator connecting two simple or complex queries. For example, "What are the capitals of the USA and Germany?" is a compound query because it has "and" in it.

To develop the conversational agents, the broader field of AI and natural language processing continues to evolve with advancements in machine learning, deep learning, neural networks, and other technologies. The structured way of generating knowledge base with specific patterns and responses is possible through specific mark-up language. AIML stands for Artificial Intelligence Markup Language, an XML base markup language meant to create artificial intelligent applications. In 2021, Md Mabrur Husan Dihyat et al. [6] wrote that AIML uses pattern-matching techniques to formulate query answers. The basic unit of the AIML script is called category tag, which is formed by user input patterns and chatbot responses according to the input. The question is stored in the $< pattern >$ tag inside each category, while the corresponding answer is stored in the $< template >$ tag. The design comprises words, spaces, and wildcard symbols such as $\wedge$ and $*$. Wildcard symbols are used to replace strings in AIML.

Specifically the focus of the work is to use a conversational AI based intelligent chatbot, which is a good solution for answering the student's specific queries regarding the admission process. It will provide 24x7 assistance, and the information will be uniform and precise [7]. The contribution of the paper are:

- Preparation of questions related fo counseling process
- Handing a various forms of the same query using semantic analysis
- Capability to process all types of questions: simple to complex
- Implementation and analysis of chatbots sing various technologies

## II. LITERATURE SURVEY

The integration of chatbots into counseling services within engineering institutes has gained traction as a means to provide accessible and timely support to students facing academic, personal, or career-related challenges. Some of the chatbots used in counseling process are briefed and comparative analysis is explained in this section. A study by Davis and Smith [8] emphasized the potential of chatbots in counseling to overcome geographical and time constraints. The implementation of chatbots allowed students to seek guidance beyond traditional office hours, leading to increased accessibility. The proposed chatbot emphasizes on the set of questions which are frequently getting asked during counseling process. Research by Johnson and Lee [9] explored the role of chatbots in providing emotional support to engineering students. The study found that chatbots equipped with sentiment analysis capabilities effectively identified and responded to students' emotional states, contributing to a supportive environment. The development proposed chatbot focuses on all types of questions which are generally need to be answered before taking decision to join any university for the course. This decision process is quite an emotional for the students and parents hence appropriate questions to be answered is a crucial task. Career-oriented chatbots were investigated by Patel et al. [10] for offering personalized career guidance to engineering students. Results indicated that students who engaged with career-focused chatbots demonstrated a clearer understanding of their career paths and increased confidence in their choices. The proposed chatbot mainly focuses on various preferences of students to decide the university for the admission. The work of Chang and Wang [11] delved into privacy concerns associated with counseling chatbots. The study highlighted the importance of secure communication channels and transparent data handling practices to ensure the confidentiality of sensitive information shared during counseling sessions. User acceptance of counseling chatbots was explored by Yang and Liu [12]. Their research found a positive correlation between the user-friendliness of chatbots and students' willingness to engage. Clear communication of the chatbot's capabilities and limitations also played a crucial role in user acceptance.

B. R. Ranoliya et al. [13] have developed a chatbot for university-related FAQs. This chatbot was implemented using Artificial Intelligence Markup Language (AIML) and Latent Semantic Analysis (LSA). Authors need to try other ways of implementation significantly when the dataset of questions increases.

Vibhor Sharma et al. [14] discussed that **ELIZA** was created by a German Computer Scientist, Joseph Weizenbaum, in 1966. It is considered to be the first chatbot in computer history. Eliza used "pattern matching" and substitution methodologies to simulate conversations.

In 2020, Eleni Adamopoulou et al. [15] that ELIZA responds like a psychotherapist by returning the user's query in an interrogative form. The downside of ELIZA is its limited knowledge, so it can only discuss a limited range of topics. Furthermore, ELIZA cannot maintain long conversations and cannot learn context from the conversation.

Shahnawaz Khan and Mustafa Raza Rabbani [16] implemented a chatbot using AI and NLP models for Islamic finance and banking customers. Authors have used the traditional NLP model to implement a chatbot where the chatbot's performance is not compared with various other methods.

Eleni Adamopoulou et al. [15] discussed a chatbot called **PARRY** created in 1971. PARRY is considered to be more evolved than ELIZA since it has a "personality" and a more effective control structure. In 2012, Sandeep A Thorat et al. [17] wrote that the ELIZA chatbot system also has language comprehension capabilities and can have variables like mistrust, anger, and fear.

Vishal Tiwari et al. [18] have implemented a chatbot using neural networks and NLP for COVID-19-related queries. The dataset of questions and answers have used to train and generate the responses. The neural network model requires a considerable dataset, and the queries are vague, so preparing the model and developing specific responses takes longer. Sushil S. Ranavare and R. S. Kamath [19] have implemented a chatbot for placement activity using the DialogFlow method. The proposed approach needs structured data handling per the pre-defined dialogs and cannot accommodate semantic queries.

Luke Fryer et al. [20] explained about **Jabberwacky**, written in CleverScript, an Artificial Intelligence tool. Eleni Adamopoulou et al. [15] authors wrote that Jabberwacky was created in 1988 and used contextual pattern-matching algorithms to answer queries based on previous discussions.

In 2020, Verma Shivang et al. [21] mentioned about Jabberwacky's main goal that was to transition from a text-based system to a fully voice-driven system. Ann Neethu Mathew et al. [22] have implemented an NLP-based personal learning assistant for school education. The chatbot proposed in this paper requires the potential to cover the whole subject's contents which can be achieved by enhancing the ontology and knowledge base.

In 2020, Shivang Verma et al. [21] implemented the Artificial Linguistic Internet Computer Entity (**ALICE**), which is Natural Language Processing chatbot. This chatbot uses heuristic pattern and matching algorithms to conduct conversations. ALICE was written using Artificial Intelligence Markup Language (AIML), an XML-based schema for writing heuristic conversational rules.

In 2020, Eleni Adamopoulou et al. [15] wrote that ALICE entirely relied on pattern-matching algorithms without recognizing the context of the entire conversation. Also, ALICE lacks intelligent traits and cannot generate human-like responses that express emotions and attributes.

Mamta Mittal et al. [23] have developed a Web-based chatbot for Frequently Asked Queries (FAQ) in Hospitals. Authors have used ML algorithms to train the dataset and NLP methods for text processing. Authors have used the Gradient descent algorithm, but no comparison has been provided concerning other algorithms.

In 2020, Shivang Verma et al. [21] explained about **Mitsuku**, an intelligent chatbot created by Steve Worswick using AIML. Santosh Maher et al.  note maher2020chat-bots implemented Mitsuku for a general type of conversation and interacted with the user using the rules written in AIML. Mitsuku can also be integrated with social media platforms like Telegram or Twitter. The chatbot is hosted at Pandorabot and employs NLP with heuristic patterns. Mitsuku can retain and utilize large amounts of conversational history in future conversations.

Quynh N. Nguyen et al. [24] performed an empirical study of user interaction with chatbot vs. menu interface. The results conclude that chatbots provide lower user satisfaction over the menu interface due to the vague nature of queries and generated answers. The authors suggested that implementing chatbots should focus on perceived autonomy, perceived competence, and cognitive effort. Hence, various methods of implementations need to be compared.

In 2020, Shivang Verma et al. [21] wrote about **Siri**, a virtual assistant developed by Apple launched in 2010. Siri uses a natural language interface that enables it to take actions, make recommendations and perform specific actions in response to voice queries. With users usage, Siri can adapt to the user's language usage and searches. Siri has many features, including handling device settings, scheduling events, and searching queries online.

Eleni Adamopoulou et al. [15] wrote about Siri's weaknesses. Siri's main disadvantage is that it depends on the internet to function. Siri can understand many languages, but many languages are not supported, while the navigational instructions are only available in English. Furthermore, Siri struggles to understand solid accents and commands in the presence of external noise.

Songhee Han and Min Kyung Lee [25] have implemented a FAQ chatbot for Massive open online courses. The authors suggest a conceptual framework for chatbots and explain how it is essential to implement and integrate chatbots for conversation-centric tasks.

In 2019, Alaa A Qaffas [26] wrote about IBM's **Watson**. Watson is a question-and-answer unit that can answer questions in natural language. Watson uses NLP and machine learning algorithms to extract insights from previous conversations. The downside of Watson is that he only supports English. Eleni Adamopoulou et al. [15] authors wrote that Watson was created in 2011, and later "Watson Health" helped doctors diagnose diseases.

In 2020, Eleni Adamopoulou et al. [15] wrote about **Google Assistant**, which consists of the next generation of **Google Now**. Google Now was created by Google in 2012 and gave responses based on users' preferences and locations. Google Assistant is a deeper artificial intelligence and has a friendlier user interface. The main disadvantages of Google Assistant are that it has no personality and violates the user's privacy because it is directly linked to their Google accounts.

In 2020, Eleni Adamopoulou et al. [15] wrote about **Cortana**, a digital assistant developed by Microsoft in 2014. It understands voice instructions, identifies time and location, sends emails, creates reminders, and manages lists. Cortana has a significant flaw in that it can run software that installs malware.

In 2016, Martin Abadi et al. [27] wrote that TensorFlow is a programming language for expressing and executing machine learning algorithms. With few or no adjustments, TensorFlow computations can be conducted on various heterogeneous systems, ranging from mobile devices like phones and tablets to large-scale distributed systems. The framework is extensible and may be used to define multiple algorithms, such as deep neural networks and inference approaches.

In 2018, Shahzad Qaiser et al. [28] wrote that Term Frequency and Inverse Document Frequency (TF-IDF) is a numerical statistic that illustrates the relevance of keywords to specific document in other words.

B. R. Ranoliya et al. [13] have developed a chatbot for university-related FAQs. This chatbot was implemented using Artificial Intelligence Markup Language (AIML) and Latent Semantic Analysis (LSA). Authors need to try other ways of implementation, significantly when the dataset of questions increases.

In 2015, Imran Ahmed et al. [29] wrote that Artificial Intelligence Markup Language (AIML) has a syntax similar to Extensible Markup Language (XML) and is used for pattern-matching algorithms.

In 2018, Ruby Rani et al. [30] wrote that in IR systems, stop words are words with little or no semantic importance. Removing such common words can lead to more effective corpus indexing and boost the performance of an IR system.

In 2021, Dan Ofer et al. [31] wrote that splitting text into atomic units of information in a selected language representation (called tokens) is known as tokenization. Although some approaches employ individual letters, most English NLP models use words as tokens. Individual-character tokens provide more versatility, particularly for out-of-vocabulary or misspelled words and languages lacking unambiguous word division.

Based on literature, following research gaps have been identified to formulate the research:

- The existing research lacks in identifying best suitable models to implement chatbots. Hence, there is a need for implementation and comparative analysis of various models to select the best one.
- Implementation of chatbot by considering all simple and complex queries related to university/institution is major missing.
- Existing chatbots lack in domain information related to educational institutions. Hence, there is a need of generating extensive question-answer repository for such chatbots.
- There is a requirement to implement conversation AI which considers domain knowledge and semantics of questions while answering.

## III. RESEARCH FORMULATION

Technical Engineering colleges follow an online admission process along with counseling for providing information regarding various courses. Students and Parents have a lot of queries regarding the entrance process which are clarified on calls or by visiting the university. There might be some miscommunication of information, or many a times officials get busy on other calls which prevents any other student from getting their queries resolved.

Engineering colleges follow an online admission process which constitutes a counseling process and stream selection. During this phase, the students and parents have various kinds of questions, such as:

- Queries regarding the college
- Queries regarding the branch they are about to select
- Queries regarding the students' options after college
- Queries regarding the placements in the final year
- Queries related to the various branches and the difference between them along with questions related to the relatively new branches.

All the queries can be clarified by visiting the college or over a phone call with the officials. Due to the shear volume of queries it might be difficult for the officials to answer all the queries properly which may lead to miscommunication of information, and sometimes the they might be busy on some other call which prevents the student from getting the information they need.

Students may have to rely on online platforms such as Quora or Telegram groups for getting information and getting their queries resolved. These sources of information are not reliable because the information is not provided by a university officials. Furthermore, students might need to navigate through the entire college website to find a particular piece of information which can be tedious.

## IV. METHODOLOGY

The development of university information chatbots involves defining objectives, understanding user requirements, selecting a suitable platform, integrating with university systems, and deploying across various channels. The overview of an developed solution is explained step-wise as follows:

- Preparation of Questions Related to Counseling Process: The solution begins with a meticulous preparation of questions related to the counseling process. This involves understanding the varied needs and concerns of users, including prospective students, and parents. The question preparation phase includes input from counseling experts to ensure the chatbot is equipped to address a wide range of inquiries related to admissions, academic programs, career guidance, and support services.
- Handling Various Forms of the Same Query Using Semantic Analysis: To enhance the effectiveness of the chatbot, semantic analysis is employed to handle various forms of similar queries. Through natural language processing techniques, the chatbot is trained to recognize the semantic meaning behind different expressions of the same question. This enables the chatbot to provide consistent and accurate responses regardless of how users phrase their queries, ensuring a more user-friendly and efficient interaction.
- Capability to Process All Types of Questions: Simple to Complex: The developed solution ensures the chatbot's versatility in processing questions of varying complexity. Whether users have straightforward queries about admission deadlines or complex inquiries regarding academic policies, the chatbot is designed to comprehend and respond appropriately. The system is equipped with an extensive knowledge base and advanced algorithms to tackle a diverse set of questions, providing comprehensive support across the counseling spectrum.
- Implementation and Analysis of Chatbots Using Various Technologies: The implementation of the chatbot solution is characterized by the use of various technologies to optimize performance and user experience. Technologies such as natural language processing (NLP), machine learning algorithms, and possibly deep learning models are integrated to enhance the chatbot's understanding of user intent and context. The solution embraces a comparative analysis of different technologies to select the most suitable ones, considering factors like accuracy, scalability, and ease of integration with existing systems.

Figure 1 depicts the methodology for developing a chatbot. The dataset is created by collecting all the questions asked about a particular technical university from various social media portals and the university's students and faculty. Answers are obtained for these questions from authorized sources from the university. The dataset has- around 250 questions formed in different ways. The following methodology is used for using these questions and answers to design

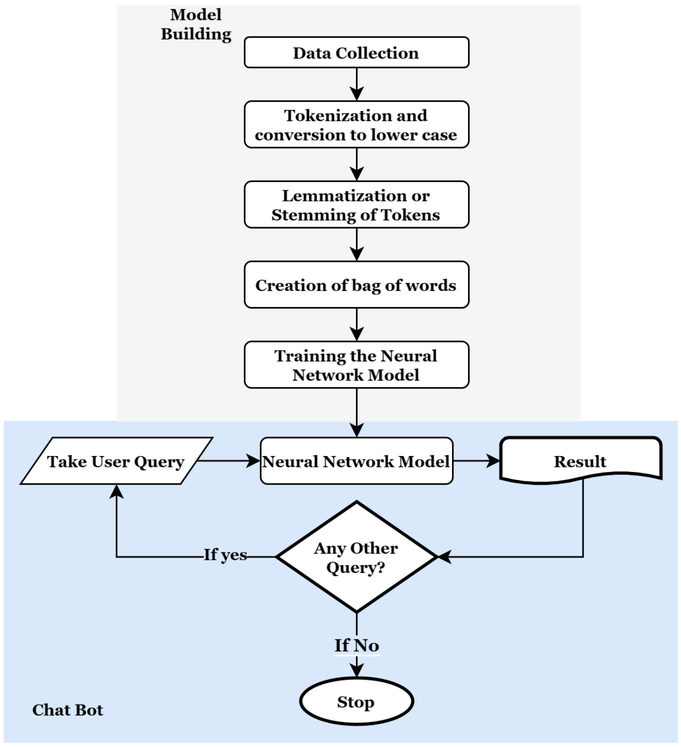


FIGURE 1. Methodology for Chatbot Development

a chatbot. In the first step, raw data is pre-processed and converted into a format that is easier and more effective for further processing steps. It also normalizes the raw data in the dataset and reduces the number of features in the feature set. This leads to a decrease in the complexity of fitting the data to each classification model.

The pre-processing steps are explained below:

- **Converting to Lowercase**: The raw text is changed to lowercase to avoid numerous variants of the same word, and all the terms, regardless of their casing, are standardized/normalized to lowercase so they can be counted together.
- **Tokenization**:Tokenization is dividing a text stream into meaningful elements called tokens. Tokens can be words, sentences, or any other part of the sentence.
- **Bag Of Words**: Neural networks cannot understand words and require numbers as input. Therefore, the Bag of Words model is used to convert words into machine-recognizable vectors of numbers. There are two types of Bag of Words: a list of zeros and ones that signify whether the word is present in the sentence. The other kind counts the number of occurrences of the most frequently used words. The size of the Bag of Words is the number of unique root words. The Bag of Words is represented as a list of 0s and 1s where each position in the list means if a comment exists or not in the sentence.
  sentence=hello, how are you?
  words=["are", "bye", "hello", "hi", "how", "i", "thank",

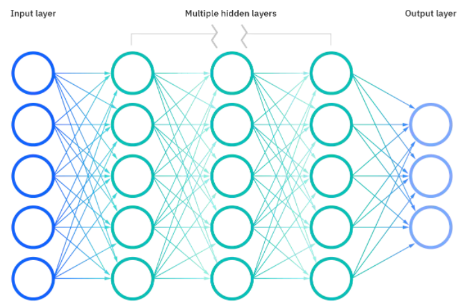


**FIGURE 2.** Structure of a Neural Network

"you"]
bag=[1,0,1,0,1,0,0,1]

- **Removing Stop Words**: The stopwords are the words that occur most frequently in a document and contain very little information that is not usually relevant. For example, in the English language, there are some words such as "a," "about," "above," "after," "again," and "against" all contain meagre information, and thus they are called stopwords. Removing stopwords reduces vector space and improves the model's performance by increasing accuracy and reducing the training time and number of calculations.
- **Stemming**:The stemming is the process of removing prefixes and suffixes(affixes) of words. When several types of features are stemmed into a single feature, it reduces the amount of features in the feature space and increase the performance of the classifier. For example, the words "likes," "like," "likely," "liked" and "liking" will all be stemmed from their root form "like". Consolidating all term versions with the same root form guarantees that the root form's frequency is assessed collectively, increasing the likelihood of the root form appearing.
- **Lemmatization**: The Lemmatization and stemming are similar as they combine several versions of a word into a single root form. When reducing and aggregating words, lemmatization considers the context and lemma of each word. As a result, it makes no distinction between words with slightly different meanings. The words "went," "going," "gone," and "go" will all become "go," even though "went" has distinct characters from "go," which the stemming algorithms do not do.

After pre-processing next step is to use it for building a model for a chatbot. This is done using a Neural network as shown in Figure 2.

The neural network framework is based on the biological neural network formed inside the human brain. An input layer, one or more hidden layers, and an output layer make up an artificial neural network. Each artificial neuron in each layer is connected to the neuron in the next layer and has a weight assigned to it.

A Feed Forward Neural Network is one of the most fundamental neural network types because it conveys information in only one direction. Contrary to feed-forward neural networks, Long Short-Term Memory (LSTM) uses recurrent neural networks, where the information flow is non-linear. While dealing with sequential data or data with a temporal link, LSTMs are favored. However, LSTMs have disadvantages: they are comparatively slow and require a sizeable high-quality dataset to get acceptable results.

Once the model is ready, the next is to design a chatbot. There are many ways of creating a chatbot; all of them will have different performances. Even if the same query is fed to all the chat-bots, their responses might be different. In this section, the following five chat-bots are explained:

- Smart Bot: In this chat-bot, the neural network is created using TensorFlow (TfLearn)
- Sam: In this chat-bot, the neural network is created using PyTorch
- Big Mouth: This chat-bot is created using TF-IDF Vectorization
- Hercules: This chatbot is created using Sequential Modeling
- ALICE: This chatbot is created using AIML

### A. SMART BOT: NEURAL NETWORK USING TENSORFLOW (TFLEARN)

As the name suggests, TensorFlow runs computations based on tensors. In machine learning, a tensor is a generalization of vectors and matrices, represented as an n-dimensional array of a base datatype. All elements of a tensor always have the same data type.

The explanation of Algorithm 1 is given in the following paragraphs. The data is stored in *intents. json* file, and it contains a list of goals. Each plan or class has a tag, a pattern, and a response. The "tag" defines the purpose or class. The "pattern" lists possible questions for the corresponding category. The "response" is a list of possible answers to the questions of that "tag." The chatbot will take the message from the user, identify the "tag" of the message, and give the corresponding response.

Every "pattern" of every "intent" is tokenized using nltk.word_tokenize() and is appended to the "words" list. All the tags are stored in the "labels" list. "words" is a list containing all the words in the database. Every word in "words" is converted to lowercase using the *lower()* function. Now, all the words in the "words" list are stemmed using *LancasterStemmer().stem()* function and all the duplicate comments are removed using *set(words)*. "words" and "labels" are then sorted. A bag of words as "bag"(empty list) is created, where the size of "bag" is the number of root words in the database. For every word in the "words" list, if that word exists in the sentence, then one is appended to "bag"; else 0 is appended to "bag".

A Deep Neural Network is created using TfLearn. The size of the input layer is equal to the size of the "bag." The input to the neural network is "bag." Two fully connected hidden layers of eight neurons each are added to the network.

Fully Connected layers mean that all possible connections are present, wherein every input of the input vector influences every output of the output vector. An output layer of size equal to the number of tags in the dataset is added to the network. The softmax activation function is applied to each neuron in the output layer. "n_epochs" is the number of times the model will see the same training data. In this model, "n_epochs" is set to 1000. The softmax activation function converts the output to a list of probabilities, with each value denoting the possibility that the sentence belongs to the corresponding tag and that the sum of all probabilities equals 1. After the model is trained, the variables are stored in the data—pickle file. The training dataset is passed through the model as a bag of words, and the model is trained. When the user query is passed through the neural network, the tag with the highest probability is chosen, and the corresponding response is given to the user [32].

**Algorithm 1** Algorithm for chatbot using TensorFlow

```
$data \leftarrow load\,data\,from\,JSON\,file$
If the model has already been trained, load the variables from the pickle file
$Initialize\,lists\,words,\,labels,\,docs_x,\,docs_y$
while $intent \in data$ do
    while $pattern \in intent$ do
        $wrds \leftarrow\, tokenize\,words\,in\,the\,pattern$
        $Apppend\,wrds\,to\,words$
        $Append\,wrds\,to\,docs_x$
        $Append\,tag\,of\,the\,intent\,to\,docs_y$
    end while
    if $tag \notin labels$ then
        $Append\,tag\,of\,the\,intent\,to\,labels$
    end if
end while

$remove\,punctuations\,from\,"words"$
$Stemming\,and\,converting\,to\,lowercase\,of$
$,"words"\,and\,store\,in\,"words"$
$sort"words"and"labels"$
Initialize lists: training, output
while $sentence \in docs_x$ do
Initialize "bag"(bag of words)
    $stem\,every\,word\,in\,"sentence"\,and$
    $store\,in\,"wrds"$
    while $word \in words$ do
        if $word \in wrds$ then
                append 1 to $bag$
        else
                append 0 to $bag$
        end if
    end while
    $Append\,"bag"\,to\,"training"$

    $output_{row}[labels.index(docs_y[x])] = 1$
    $Append\,"bag"\,to\,"training"$
    $Append\,output_{row}\,to\,output$
end while
convert “training” and “output” to array and save the variables in pickle file
create a Deep Neural Network using tflearn. The size of the input layer is same as
the size of the bag of words. Then add 2 hidden layers (fully connected) of 8 neurons
each. The size of the output layer is equal to the number of tags
$set\,"number\,of\,epochs"\,to\,1000$

$save\,the\,model$
```

## B. SAM: NEURAL NETWORK USING PYTORCH

The explanation of Algorithm 2 is given in the following paragraphs. The data is stored in *intents. json* file, and it contains a list of intents. Each intent or class has a tag, a pattern, and a response. The "tag" defines the intent or class. The "pattern" is a list of possible questions for the corresponding class. The "response" is a list of possible answers to the questions of that "tag." The chatbot will take the message from the user, identify the "tag" of the message, and give the corresponding response.

Pre-processing steps are applied to the data. Every question of every intent is tokenized using *nltk.word_tokenize()* and is appended to the "all_words" list. Every unique tag is stored in the "tags" list. Now, "all_words" is a list that contains all the tokenized words of the dataset, and "tags" is a list that contains all the tags of the database. All the punctuation tokens are removed; every word in the dataset is converted to lower case using the *lower()* function, and the words are stemmed using *PorterStemmer().stem()* function from nltk. "all_words" list is sorted using *sorted(all_words)* function and all the duplicate words are removed using *set(all_words)* function. "tags" list is also sorted. To create a bag of words, a "bag" list is created. "bag" list is of length equal to the size of the list "all_words" or the number of unique stemmed words in the database. "bag" list is initialized with 0. For every word in "sentence," use its corresponding "index" in "all_words" to set bag[index] to 1.

A Feed Forward Neural Network is created using a torch—module, a base class for all neural network modules. A feed-forward neural network is an artificial neural network in which the connections between nodes do not form a cycle. One linear input layer of size equal to the "bag" list is created. Two hidden linear layers having eight neurons are created. One output layer of size similar to the size of the "tags" list is formed. A ReLu activation function is defined. Training data is passed through the input layer; then, the activation function is applied. This data is fed to the hidden layer, and then the activation function is used, which is provided to another hidden layer. The activation function is applied, and this data is fed to the output layer. The learning rate is a vital hyperparameter that determines how fast the neural network converges to an optimum value. In this model, the value of the learning rate is 0.001. When the user query is passed through the neural network, the "tag" with the highest probability is chosen, and the response is given to the user.

## C. BIG MOUTH: USING TF-IDF VECTORIZATION

The term "Term Frequency"(TF) is used to count how many times a time appears in a document [33]. There are 5000 words in document "T1," and the word "alpha" appears ten times. As a result, the term "alpha" frequency in document "T1" will be

**TF**=t/s

*where t is number of occurrences in a file and s is the total number of words in the document*

TF=10/5000=0.002

The inverse document frequency gives less weight to frequently occurring words and more weight to infrequently occurring words. For example, if we have ten documents and

**Algorithm 2** Algorithm for chatbot using PyTorch

```
$Data = Load\, JSON\, Data$
$Initialize\, lists\, tags, xy, all_{words}$
while $intent \in data$ do
    $Tag\, of\, the\, intent\, is\, stored\, in\, "tag"$
    while $question \in intent[question]$ do
        $xy \leftarrow xy + tokenized\, sentence$
    end while
end while
Remove stop words
Apply stemming, remove all duplicates, sort it and add them to "tags"
$Initialize\, lists\, X_{train}, y_{train}$
while $pattern \in xy$ do
    while $tag \in xy$ do
        $Create\, bag\, of\, words\, using\, "pattern"\, and\, "all_{words}"\, and$
        $store\, in\, "bag"$
        $append\, "bag"\, to\, X_{train}$
        $append\, tag\, of\, intent\, to\, label$
        $append\, "label"\, to\, y_{train}$
    end while
end while
A Forward Neural Network is created using PyTorch with two hidden layers with ReLu activation functions. The output layer is of size equal to the number of tags in the database. The ReLu activation function is applied to the input and hidden layers. The title with the highest probability is chosen. Training loss is calculated and printed after every 100 epochs, and the final loss is computed.
$query = input\, from\, user$
$X = query\, is\, tokenized\, and\, converted\, to\, bag\, of\, words$
$output = query\, is\, passed\, through\, the\, model$
if $probability > 0.75$ then
    if $tag \in tags$ then
                    Randomly print one of the responses in that tag
    end if
else
                    Print that the bot does not understand the query
end if
```

the term "alpha" appears in five of them, we may calculate the inverse document frequency as

**IDF**=$\log(M/m)$
*where M is the total number of documents in the corpus and m is then number of documents containing the required term*
IDF=$\log(10/5)$=0.301
The detailed steps of TF-IDF Vectorization are shown in Algorithm 3 and Algorithm 4.

**Algorithm 3** Algorithm for TF-IDF Vectorization

```
Data= Query input from the user
Output= Response from chat-bot

$sentTokens \leftarrow Sentence\, tokenized\, data\, from\, database$
Append the query to sentTokens
$tfidf \leftarrow TF - IDF\, vectorized\, data\, with\, stop\, words\, removed$
$vals \leftarrow Cosine\, Similarity\, between\, the\, user$
 $query(tfidf[-1])\, and\, tfidf$
$reqTFIDF \leftarrow Maximum\, Cosine\, Similarity\, in\, vals$
if $reqTFIDF > 0$ then
    $return\, the\, corresponding\, response$
else
    $return"\, I\, do\, not\, understand.."$
end if
```

In 2020, Abhishek Jaglan et al. [34] authors wrote that textual data can not be employed in the model directly, instead it has to be converted to numerical vectors. This can be done by assigning a unique number to each word, and given data can be encoded with the length of vocabulary of known words. The Bag-of-Words model is a way of representing whether the words exists in the "sentence" or not regardless of their

**Algorithm 4** Algorithm for Greeting in TF-IDF Vector chatbot

```
Data=Query from the user
Output=Response from the chat-bot if the query is a greeting

$greetingInput \leftarrow List\, of\, greeting\, input\, words$
$greetingOutput \leftarrow List\, of\, greeting\, output\, words$
while $word \in query$ do
    if $lowerCase(word) \in greetingInput$ then
        $return\, random\, greetingResponse$
    end if
end while
```

sequence of appearance.

Term Frequency-Inverse Document Frequency (TF-IDF) stores the component of resulting scores assigned to each word. The goal of TF-IDF vector is to calculate the word frequency scores for the text that are more interesting (less common). Term Frequency is used to calculate the frequency of each word, whereas, Inverse Document Frequency down scales the score of frequently occurring words.

The explanation of Algorithm 5 is given the following paragraph. The corpus consists of full stop separated answers in the form of a text file. The corpus is loaded and is tokenized using *nltk.word_tokenize* and *nltk.sent_tokenize*. The tokens are lemmatized using *nltk.stem.WordNetLemmatizer().lemmatize()*. Words which are there in *string.punctuation* (set of punctuations) are removed. Input is taken from the user in form of a "query". Greeting is implemented using the pattern matching algorithm. If the query contains any words from GREETING_INPUT (list of predefined greeting inputs), then the chat-bot will return a random response from GREETING_OUTPUT (list of predefined greeting outputs). "query" is tokenized and lemmatized and the result is appended to the list "sent_tokens". sklearn is the library used for TF-IDF vectorization and to calculate the cosine similarity. Cosine similarity is calculated between every sentence from the corpus and the user query, the sentence having the highest cosine similarity is given as the output (cosine similarity values are sorted in descending order and the first value is selected) [35].

### D. HERCULES: USING SEQUENTIAL MODELING

The data is stored in *intents. json* file and contains a list of intents. Each intent or class has a tag, a pattern, and a response. The "tag" defines the intent or class. The "pattern" is a list of possible questions of the corresponding class. The "response" is a list of possible answers to the questions of that "tag." The chatbot will take the message from the user, identify the "tag" of the message, and give the corresponding response.

The explanation of Algorithm 6 is given in the following paragraphs. Every question of every intent is tokenized using *nltk.word_tokenize()* and is appended to the "words" list. All the tags are stored in the "labels" list. "words" contains all the words in the database. Every word in "words" is

**Algorithm 5** Algorithm for chatbot using IF-IDF Vectorization

```
$data = load\, data.txt$
$sent_{tokens} = sentence\, tokenization\, of\, data$
$word_{tokens} = word\, tokenization\, of\, the\, text$
$Remove\, punctuations\, from\, the\, text$
$Initialize\ lists\ GREETING_{INPUT},\ GREETING_{OUTPUT}$
$with\, sample\, greeting$
$inputs\, and\, outputs$
while $word \in sentence.split$ do
    if $word \in GREETINGS_{INPUT}$ then
        Print out a random $GREETINGS_{OUTPUT}$
    end if
end while
$user_{Input} = Input\, from\, the\, user$
$user_{response} = tokenized\, user\, query$
$Append\, user_{response}\, to\, sent_{tokens}$
$tfidfVec = Create\, tfidf\, Vector\, of\, sent_{tokens}$
$vals = Cosine\, Similarity\, between\, "sent_{tokens}"\, and\, "user_{response}"$
$idx = Id\, of\, the\, sentence\, with\, the\, highest\, Cosine\, Similarity$
if $req_{tfidf} = 0$ then
        Print "I am sorry, I didnt understand you"
else
        Print $sent_{tokens}[idx]$
end if
```

converted to lowercase using *lower()* function. Now, all the words in the "words" list are lemmatized using *WordNetLemmatizer().lemmatize()* function and all the duplicate words are removed using *set(words)*. Both "words" and "labels" are sorted. A bag of words is created having the variable name "bag," where the size of "bag" is the number of root words in the database. For every word in the "words" list, if that word exists in the sentence, then one is appended to "bag"; else, 0 is appended to "bag".

In 2014, authors Nitish Srivastava et al. [36] wrote that dropout is a technique that prevents overfitting and provides a way of efficiently combining different neural networks. The term "dropout" refers to dropping out units randomly from the hidden or visible layers in the neural network. By dropping a team, it is temporarily removed from the web and its incoming and outgoing connections by setting its weight to zero.

A Neural Network with three layers is created. A dense or fully connected input layer is equal to a "bag" size with a ReLu activation function. Dropout is used, which will drop 50% of the units. A fully connected hidden layer of 64 neurons is created, and the ReLu activation function is applied. Dropout is used, which will drop 30% of the units. A dense output layer of size equal to the number of tags in the database is created with the Softmax activation function. The Softmax activation function converts the output to a list of probabilities, wherein each value denotes the likelihood of the sentence belonging to the corresponding tag, and the sum of all possibilities equals 1. The model is optimized using "Adam." The Adam Optimizer is an adaptive learning rate method, which computes individual learning rates for different parameters. When the user query is passed through the neural network, the tag with the highest probability is chosen, and the response is given to the user.

**Algorithm 6** Algorithm for chatbot using Sequential Modeling

```
$data = load\, data\, from\, JSON\, file$
$Implemet\, Lemmatization$
$Initialize\, lists\, words, classes, doc_x, doc_y$
while $intent \in data$ do
    while $pattern \in intent$ do
        $wrds = tokenize\, words\, in\, the\, pattern$
        $Append\, "wrds"\, to\, words$
        $Append\, "wrds"\, to\, doc_x$
        $Append\, "tag"\, to\, doc_y$
    end while
    if $tag \notin labels$ then
        $Append\, "tag"\, to\, labels$
    end if
end while
remove punctuations from "words"
$words = stemed\, "words"\, converted\, to\, lowercase$
sort "words" and "labels"
Initialize lists: training, output
while $sentence \in docs_x$ do
    $Initialize\ "bag"(bag\, of\, words)$
    $wrds = stem\, every\, word\, in\, "sentence"$
    while $word \in words$ do
        if $word \in wrds$ then
                append 1 to $bag$
        else
                append 0 to $bag$
        end if
    end while
    $Append\, "bag"\, to\, training$

    $output_{row}[labels.index(docs_y[x])] = 1$
    $Append\, "bag"\, to\, training$
    $Append\, output_{row}\, to\, output$
end while
create a Sequential model with softmax activation function
```

### *E. ALICE:USING AIML*

In AIML, *categories* are the basic unit of knowledge. Each category has a *pattern* and a *template*. The *pattern* describes the query, and *template* describes the chatbot's responses. The *template* tag can have a list of possible responses for the chatbot to choose from, and it will randomly give one response.
There are two types of AIML classes:

- **Atomic Category:** It is an AIML classification where the query are an exact match. This type of classification does not contain any wildcards.
  $< category >$
  $< pattern > Good\, Morning < /pattern >$
  $< template > Good\, Morning\, to\, you\, too! < /template >$
  $< /category >$
- **Default category:** wildcard symbols such at $\wedge$ and $*$ are used in the *pattern*. $*$ wildcard captures one or more words and $\wedge$ wildcard captures 1 or more words.
  $< category >$
  $< pattern > Hi, \wedge < /pattern >$
  $< template > Hi,\, Good\, to\, see\, you < /template >$
  $< /category >$

These five chat-bots were created to see how different technologies and algorithms impact a chatbot's performance. Confusion matrices were calculated using the sklearn library.

Furthermore, all five chat-bots were implemented, and several epochs were varied. Lastly, 250 queries were executed on all the chat-bots, and all the responses were observed.

## V. RESULT ANALYSIS

The results of all the implemented chat-bot are represented in accuracy and validation. The first section contains the confusion matrices and accuracy, calculated based on a sample training dataset. In the second section, 150 queries were implemented on all the chat-bots, and their responses were observed to check if they categorized the query correctly or not. Lastly, the last section contains screenshots of the conversation with the bots.

### *A. CONFUSION MATRICES AND ACCURACY BASED ON SAMPLE TEST DATASET*

A test dataset having 144 queries of one university is used to test the chatbot models. Confusion matrices and accuracies are calculated using sklearn metrics library. (using confusion_matrix and accuracy_score functions)

**TABLE 1.** Confusion matrix of all the Chatbots

| | | |
|---|---|---|
| Smart Bot | Actual True | Actual False |
| Predicted True | 10 | 2 |
| Predicted False | 2 | 130 |
| Sam | Actual True | Actual False |
| Predicted True | 8 | 4 |
| Predicted False | 4 | 128 |
| Big Mouth | Actual True | Actual False |
| Predicted True | 8 | 4 |
| Predicted False | 4 | 128 |
| Hercules | Actual True | Actual False |
| Predicted True | 10 | 2 |
| Predicted False | 2 | 130 |
| ALICE | Actual True | Actual False |
| Predicted True | 3 | 9 |
| Predicted False | 9 | 123 |

The neural network was created using TensorFlow in this model, and multiple pre-processing steps were applied. The Lancaster Stemming algorithm was used in the pre-processing phase, which is more accurate. Furthermore, the softmax activation function is applied to the output layer, increasing the neural network's performance. Table 1 is the confusion matrix of Smart Bot.

In this model, the neural network was created using PyTorch, and the ReLu activation function was applied to the input and hidden layers. This impacted the performance of the model. Table 1 is the confusion matrix of Sam.

In this model, the neural network is not created. Instead, TF-IDF Vectorization converts every sentence into a vector, and Cosine Similarity calculates the similarity between every sentence and the query. This model needs to understand the meaning of the query; it simply finds the most similar sentence. Table 1 is the confusion matrix of Big Mouth.

In this model, Sequential modeling is used while creating the neural network, which was designed to prevent the prob-

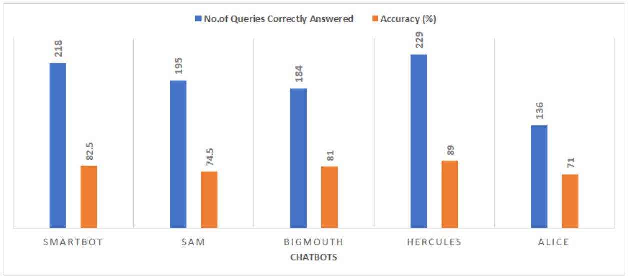


**FIGURE 3.** Query-wise Accuracy of the all Chatbots

lem of overfitting; this improves the model's performance. Table 1 is the confusion matrix of Hercules.

In this model, AIML is used to create pattern-matching rules. No computation is done here, and the query is matched to the predefined rules. The programmer needs to understand the AIML functionalities to get acceptable results deeply. Table 1 is the confusion matrix of ALICE.

### *B. QUERY ANALYSIS ON CHATBOTS*

One hundred fifty simple queries were created, and 15 had spelling mistakes. All these queries were implemented on all the chat-bots, and their responses were observed to check if they categorized the question correctly or not implemented.

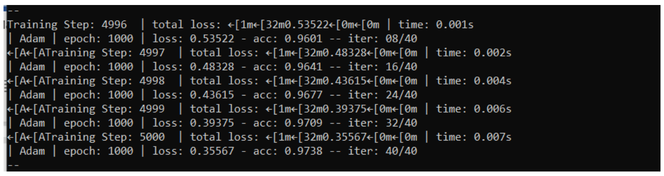
Figure 3 shows the number of queries correctly answered by each chatbot along with the accuracy of each model. Table 2 depicts how many questions were correctly answered by each model.

Table 2 depicts the various queries and how many questions were correctly answered by each model.

### *C. CONVERSATIONAL ANALYSIS OF CHATBOTS*

In this section, simple and compound queries are implemented on all the chatbots, and the conversation shown in the form of screenshots for sample queries. For Smart Bot, Sam, and Hercules, training loss is calculated, and its variations are noted when the number of epochs changes.

#### 1) Smart Bot



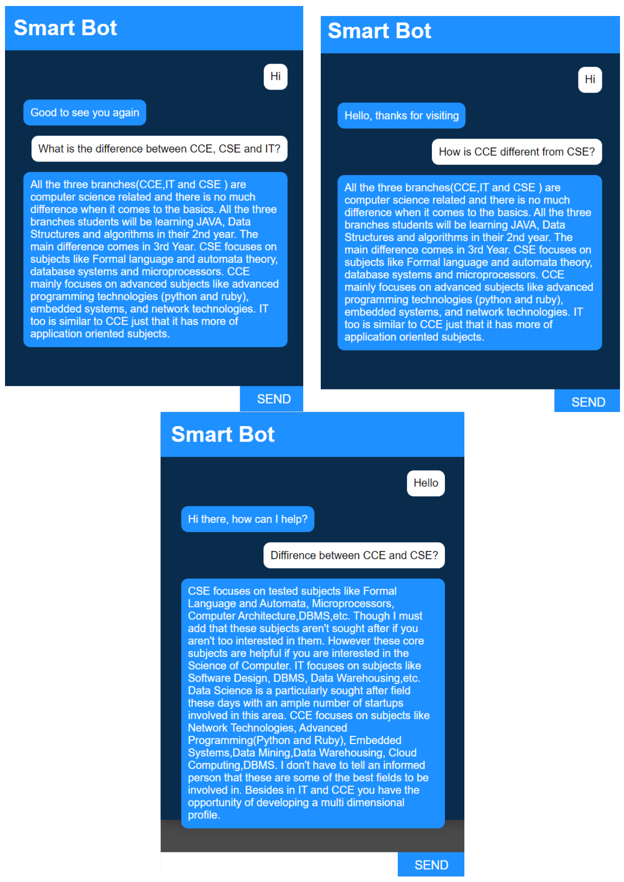
```
--
Training Step: 4996  | total loss: ←[1m←[32m0.53522←[0m←[0m | time: 0.001s
| Adam | epoch: 1000 | loss: 0.53522 - acc: 0.9601 -- iter: 08/40
←[A←[ATraining Step: 4997  | total loss: ←[1m←[32m0.48328←[0m←[0m | time: 0.002s
| Adam | epoch: 1000 | loss: 0.48328 - acc: 0.9641 -- iter: 16/40
←[A←[ATraining Step: 4998  | total loss: ←[1m←[32m0.43615←[0m←[0m | time: 0.004s
| Adam | epoch: 1000 | loss: 0.43615 - acc: 0.9677 -- iter: 24/40
←[A←[ATraining Step: 4999  | total loss: ←[1m←[32m0.39375←[0m←[0m | time: 0.006s
| Adam | epoch: 1000 | loss: 0.39375 - acc: 0.9709 -- iter: 32/40
←[A←[ATraining Step: 5000  | total loss: ←[1m←[32m0.35567←[0m←[0m | time: 0.007s
| Adam | epoch: 1000 | loss: 0.35567 - acc: 0.9738 -- iter: 40/40
--
```

**FIGURE 4.** Training of Smart Bot

As shown in Figure 4 At the 1000th epoch, the training loss of the model is 0.35567, and the accuracy is 0.9738. If we increase the number of epochs to 1500, the training loss of the model reduces to 0.15079, and the accuracy increases to 0.9949. On the contrary, if the number of epochs becomes

TABLE 2. Question-wise performance of chatbots

| Query | SB | SM | BM | HS | AL |
|---|---|---|---|---|---|
| What is the difference between CCE and IT? | × | ✓ | ✓ | ✓ | ✓ |
| How is CCE different from IT? | ✓ | × | × | ✓ | × |
| What is the difference between CCE and CSE? | × | ✓ | ✓ | ✓ | ✓ |
| How is CCE different from CSE? | ✓ | ✓ | × | ✓ | × |
| What is CCE course at college X? What is its scope compared to CSE and IT? | ✓ | ✓ | × | ✓ | × |
| How is the CCE branch? | ✓ | × | ✓ | ✓ | ✓ |
| How is the Computer and Communication branch? | ✓ | × | ✓ | ✓ | × |
| Is the CCE branch good? | ✓ | × | ✓ | × | ✓ |
| How is the CCE branch at college X? | × | × | ✓ | ✓ | ✓ |
| Is the CCE branch good? | ✓ | × | ✓ | × | ✓ |
| How is the ECE branch? | ✓ | ✓ | × | ✓ | ✓ |
| Is the ECE branch good? | ✓ | ✓ | × | ✓ | ✓ |
| Is ECE branch good? | ✓ | ✓ | × | ✓ | ✓ |
| Is the ECE branch bad? | ✓ | ✓ | × | ✓ | ✓ |
| How is the Electronics and Communication branch? | × | ✓ | × | ✓ | × |
| Is the Electronics and Communication branch good? | × | ✓ | × | ✓ | × |
| Is Electronics and Communication branch good? | × | ✓ | × | ✓ | × |
| Is the Electronics and Communication branch bad? | × | ✓ | × | ✓ | × |
| How are the placements? | ✓ | ✓ | ✓ | ✓ | ✓ |
| How are the placements for CCE? | ✓ | ✓ | ✓ | ✓ | × |
| How are the placements for Computer and Communication branch | ✓ | ✓ | × | × | × |
| Are the placements good? | ✓ | ✓ | ✓ | ✓ | × |
| Do cce students get placed in good companies? | ✓ | × | ✓ | × | × |
| what is the placement situation like? | ✓ | ✓ | × | ✓ | × |
| How are the placements for ECE students? | ✓ | ✓ | × | ✓ | × |
| How are the placements for ECE students at college X? | ✓ | ✓ | × | ✓ | × |
| Are the placements of ECE students good? | ✓ | × | × | ✓ | × |
| Do ECE students get placed in good companies | ✓ | × | × | × | × |
| How are placements for EEE and ECE? | ✓ | ✓ | ✓ | ✓ | × |
| How are placements for ECE and EEE? | ✓ | ✓ | ✓ | ✓ | × |
| Do ECE students get more job opportunities than EEE students? | × | × | × | × | × |
| Do companies prefer ECE students over EEE students? | × | × | × | × | × |
| Do companies prefer EEE students over ECE students? | × | × | × | × | × |
| Whether separate paper exists for GATE exam or they need to write the paper of Computer Science? | ✓ | × | ✓ | ✓ | ✓ |
| What is the procedure for CCE students for appearing in GATE? | × | × | ✓ | × | × |
| What is the scope in government sector? | ✓ | ✓ | ✓ | ✓ | ✓ |
| Can CCE students apply for government jobs? | ✓ | ✓ | ✓ | ✓ | × |
| Can students apply for government jobs after graduation? | ✓ | ✓ | ✓ | ✓ | ✓ |
| Can CCE passout students apply for government jobs? | ✓ | ✓ | ✓ | ✓ | × |
| Can CCE students apply for gvnt jobs? | ✓ | ✓ | ✓ | ✓ | × |
| What is the ratio of boys to girls in a class? | ✓ | ✓ | ✓ | ✓ | ✓ |
| What is the ratio of girls to boys? | ✓ | ✓ | ✓ | ✓ | ✓ |
| Is entrepreneurship encouraged by the college? | × | ✓ | × | ✓ | × |
| What is the prospect of becoming entrepreneur after completing the course? | ✓ | ✓ | × | ✓ | ✓ |
| What are the opportunities for a budding entrepreneur? | ✓ | ✓ | × | ✓ | × |



Hi

Good to see you again

What is the difference between CCE, CSE and IT?

All the three branches(CCE,IT and CSE ) are computer science related and there is no much difference when it comes to the basics. All the three branches students will be learning JAVA, Data Structures and algorithms in their 2nd year. The main difference comes in 3rd Year. CSE focuses on subjects like Formal language and automata theory, database systems and microprocessors. CCE mainly focuses on advanced subjects like advanced programming technologies (python and ruby), embedded systems, and network technologies. IT too is similar to CCE just that it has more of application oriented subjects.

Hi

Hello, thanks for visiting

How is CCE different from CSE?

All the three branches(CCE,IT and CSE ) are computer science related and there is no much difference when it comes to the basics. All the three branches students will be learning JAVA, Data Structures and algorithms in their 2nd year. The main difference comes in 3rd Year. CSE focuses on subjects like Formal language and automata theory, database systems and microprocessors. CCE mainly focuses on advanced subjects like advanced programming technologies (python and ruby), embedded systems, and network technologies. IT too is similar to CCE just that it has more of application oriented subjects.

Hello

Hi there, how can I help?

Diffirence between CCE and CSE?

CSE focuses on tested subjects like Formal Language and Automata, Microprocessors, Computer Architecture,DBMS,etc. Though I must add that these subjects aren't sought after if you aren't too interested in them. However these core subjects are helpful if you are interested in the Science of Computer. IT focuses on subjects like Software Design, DBMS, Data Warehousing,etc. Data Science is a particularly sought after field these days with an ample number of startups involved in this area. CCE focuses on subjects like Network Technologies, Advanced Programming(Python and Ruby), Embedded Systems,Data Mining,Data Warehousing, Cloud Computing,DBMS. I don't have to tell an informed person that these are some of the best fields to be involved in. Besides in IT and CCE you have the opportunity of developing a multi dimensional profile.

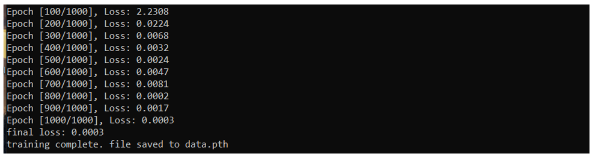
FIGURE 5. Interface of Smart Bot

500, the training loss becomes 0.24558, and the accuracy becomes 0.9817.
It can be observed that increasing the number of epochs increases the accuracy of the chatbot and decreases the training loss.

Figures shown in 5, it can be seen the chat-bot gives the correct output for the queries for which it was not trained. For example, the first query is in the dataset whereas the second query is not in the dataset. Furthermore, even if there is the spelling mistake in the query asked by the user, the model gives the correct output.

For complex and Compound queries, the chatbot gives accurate results. It can be seen in the above figures that the chatbot provides the answers to both queries.

### 2) Sam



```
Epoch [100/1000], Loss: 2.2308
Epoch [200/1000], Loss: 0.0224
Epoch [300/1000], Loss: 0.0068
Epoch [400/1000], Loss: 0.0032
Epoch [500/1000], Loss: 0.0024
Epoch [600/1000], Loss: 0.0047
Epoch [700/1000], Loss: 0.0081
Epoch [800/1000], Loss: 0.0002
Epoch [900/1000], Loss: 0.0017
Epoch [1000/1000], Loss: 0.0003
final loss: 0.0003
training complete. file saved to data.pth
```

**FIGURE 6.** Training of Sam

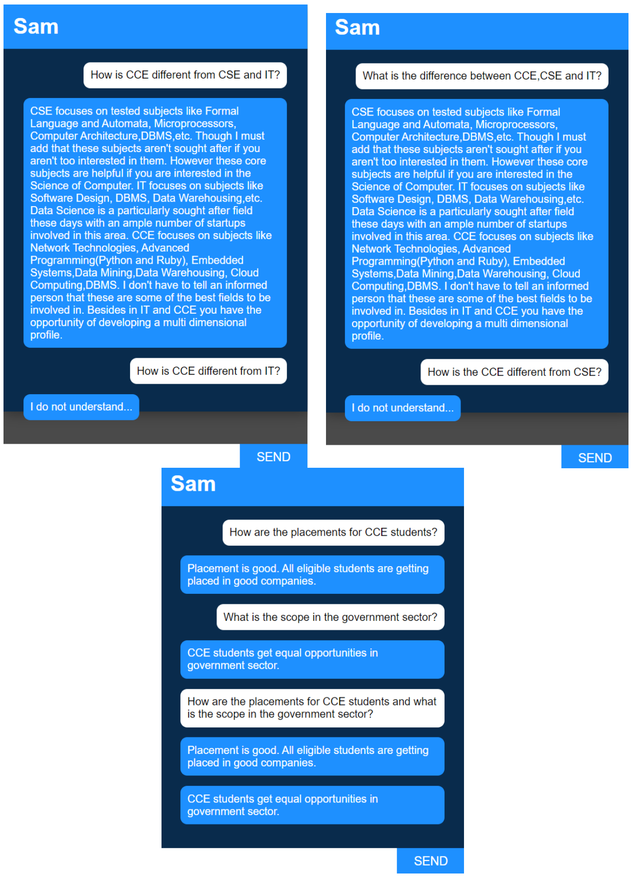
At the 1000th epoch, the training loss of the chatbot is 0.0003 as shown in V-C2. If the number of epochs is increased to 1500, the loss in training decreases to 0.0001. On the contrary, if the number of epochs is reduced to 500, the training loss rises to 0.0017.

It can be observed that by increasing the number of epochs, the training loss decreases.

Figures shown in V-C2, the model gives the correct answers to the queries in the dataset but the wrong answers to the queries not in the dataset. One of the possible explanations for this is the overfitting of the model on the training dataset. Overfitting makes the model relevant only to the dataset on which it was trained and irrelevant to all the other datasets.

In the above figure, it is observed that the chatbot does not give the correct response if there is a spelling mistake in the query.

For complex and Compound queries, the chatbot gives accurate results. It can be seen in the above figures that the chatbot provides the answers to both queries.



Sam

How is CCE different from CSE and IT?

CSE focuses on tested subjects like Formal Language and Automata, Microprocessors, Computer Architecture,DBMS,etc. Though I must add that these subjects aren't sought after if you aren't too interested in them. However these core subjects are helpful if you are interested in the Science of Computer. IT focuses on subjects like Software Design, DBMS, Data Warehousing,etc. Data Science is a particularly sought after field these days with an ample number of startups involved in this area. CCE focuses on subjects like Network Technologies, Advanced Programming(Python and Ruby), Embedded Systems,Data Mining,Data Warehousing, Cloud Computing,DBMS. I don't have to tell an informed person that these are some of the best fields to be involved in. Besides in IT and CCE you have the opportunity of developing a multi dimensional profile.

How is CCE different from IT?

I do not understand...

Sam

What is the difference between CCE,CSE and IT?

CSE focuses on tested subjects like Formal Language and Automata, Microprocessors, Computer Architecture,DBMS,etc. Though I must add that these subjects aren't sought after if you aren't too interested in them. However these core subjects are helpful if you are interested in the Science of Computer. IT focuses on subjects like Software Design, DBMS, Data Warehousing,etc. Data Science is a particularly sought after field these days with an ample number of startups involved in this area. CCE focuses on subjects like Network Technologies, Advanced Programming(Python and Ruby), Embedded Systems,Data Mining,Data Warehousing, Cloud Computing,DBMS. I don't have to tell an informed person that these are some of the best fields to be involved in. Besides in IT and CCE you have the opportunity of developing a multi dimensional profile.

How is the CCE different from CSE?

I do not understand...

Sam

How are the placements for CCE students?

Placement is good. All eligible students are getting placed in good companies.

What is the scope in the government sector?

CCE students get equal opportunities in government sector.

How are the placements for CCE students and what is the scope in the government sector?

Placement is good. All eligible students are getting placed in good companies.

CCE students get equal opportunities in government sector.

**FIGURE 7.** Interface of Sam

### 3) Big Mouth

In the above figure, the model gives the correct answers to the direct queries, but if the queries are asked differently, it may provide a different response.

As shown in Figure 8, it is observed that the chatbot might not give the correct response if there is a spelling mistake in the query.

For complex and Compound queries, the chatbot gives accurate results. It can be seen in the above figures that the chatbot provides answers to both queries.

### 4) Hercules

At the 200th epoch as shown in Figure 9, the training loss of the model is 0.0354, and the accuracy is 0.9831. If we increase the number of epochs to 250, the training loss of the model rises to 0.2196, and the accuracy reduces to 0.9800. On the contrary, if the number of epochs becomes 150, the training loss becomes 0.1089, and the accuracy becomes 0.9400.

It can be observed that increasing the number of epochs increases the accuracy of the chatbot and decreases the training loss. A similar trend is seen if the number of epochs is reduced.

In the Figure 10, it can be seen the chat-bot gives the correct output for the queries for which it was not trained. For example, the first query is in the dataset whereas the second query is not in the dataset. In the above figure, it is also observed that the bot can give the correct response even if there is a spelling mistake in the query.

For complex and Compound queries, the chat-bot gives accurate results. It can be seen in the above figures that the chat provides answers to both the halves of the query.

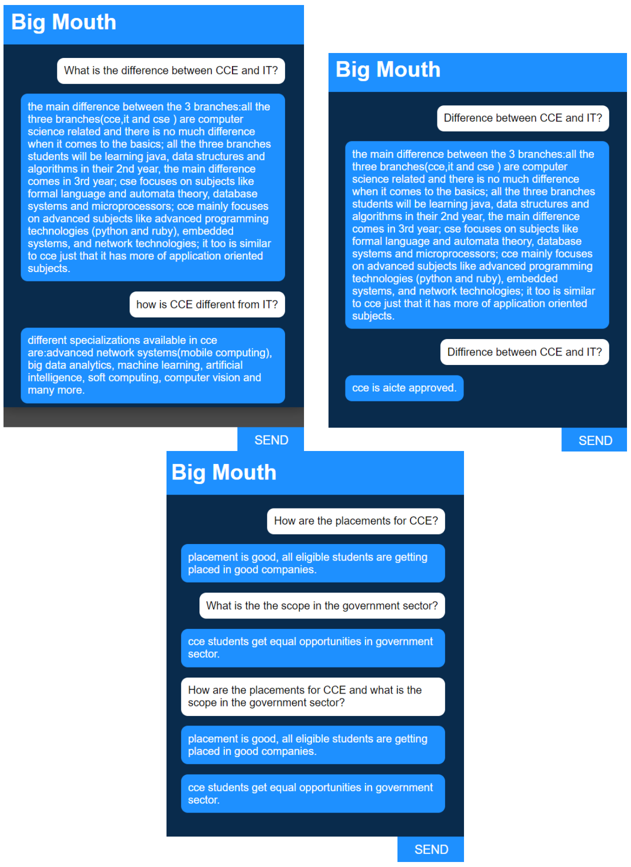
### 5) ALICE

The above picture shows that the chatbot gives the correct output for the queries it was not trained for. For example, the first query is in the dataset, whereas the second query is not in the dataset.

The above figure shows that the bot does n give the correct response even if there is a spelling mistake in the query.

For complex and Compound queries, the chatbot does not give accurate results. It can be seen in the above figures that the answer provided by the chatbot needs to be completed, as it only answers one query.

Considering Sam as the base chatbot, it can be observed that Smart Bot and Hercules perform better than Sam. Furthermore, it can be observed that Big Mouth and Alice do not perform well as compared to Sam. SmartBot, Sam, and Hercules are all neural network-based models with a " yes " percentage greater than 60 percent. If these three chat-bots



What is the difference between CCE and IT?

the main difference between the 3 branches:all the three branches(cce,it and cse ) are computer science related and there is no much difference when it comes to the basics; all the three branches students will be learning java, data structures and algorithms in their 2nd year, the main difference comes in 3rd year; cse focuses on subjects like formal language and automata theory, database systems and microprocessors; cce mainly focuses on advanced subjects like advanced programming technologies (python and ruby), embedded systems, and network technologies; it too is similar to cce just that it has more of application oriented subjects.

how is CCE different from IT?

different specializations available in cce are:advanced network systems(mobile computing), big data analytics, machine learning, artificial intelligence, soft computing, computer vision and many more.

Difference between CCE and IT?

the main difference between the 3 branches:all the three branches(cce,it and cse ) are computer science related and there is no much difference when it comes to the basics; all the three branches students will be learning java, data structures and algorithms in their 2nd year, the main difference comes in 3rd year; cse focuses on subjects like formal language and automata theory, database systems and microprocessors; cce mainly focuses on advanced subjects like advanced programming technologies (python and ruby), embedded systems, and network technologies; it too is similar to cce just that it has more of application oriented subjects.

Difference between CCE and IT?

cce is aicte approved.

How are the placements for CCE?

placement is good, all eligible students are getting placed in good companies.

What is the the scope in the government sector?

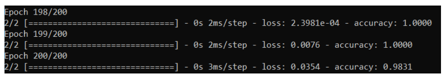
cce students get equal opportunities in government sector.

How are the placements for CCE and what is the scope in the government sector?

placement is good, all eligible students are getting placed in good companies.

cce students get equal opportunities in government sector.

FIGURE 8. Interface of Big Mouth

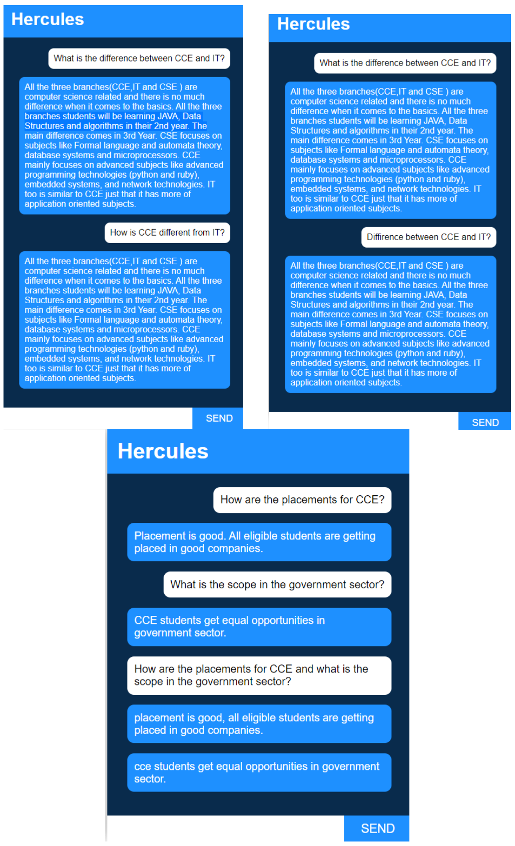


```
Epoch 198/200
2/2 [==============================] - 0s 2ms/step - loss: 2.3981e-04 - accuracy: 1.0000
Epoch 199/200
2/2 [==============================] - 0s 2ms/step - loss: 0.0076 - accuracy: 1.0000
Epoch 200/200
2/2 [==============================] - 0s 3ms/step - loss: 0.0354 - accuracy: 0.9831
```

FIGURE 9. Training of Hercules

are arranged in the increasing order of rate of "yes," then the order will be Sam<Smart bot<Hercules. Since all three models are neural network-related, it can be concluded that:

- stemming algorithms applied to the data might affect the neural network's performance. The Smart Bot LancasterStemmer algorithm was used, whereas the Sam PorterStemmer algorithm was used.
- The type of activation function and the layers on which it is applied might affect the neural network's performance. In Smart Bot, the softmax activation function is applied to the output layer, and no activation function is applied to any other layer. On the other hand, in Sam, the ReLu activation function is applied to each node in the input layer, and every node in the hidden layers, and no activation function is applied to the output layer.
- It can be noticed that Hercules has a Sequential Neural Network designed to prevent overfitting. This may be why Hercules has the highest percentage of "yes."
- It can be observed that Hercules is the only chatbot with any optimizer applied, improving its performance.



What is the difference between CCE and IT?

All the three branches(CCE,IT and CSE ) are computer science related and there is no much difference when it comes to the basics. All the three branches students will be learning JAVA, Data Structures and algorithms in their 2nd year. The main difference comes in 3rd Year. CSE focuses on subjects like Formal language and automata theory, database systems and microprocessors. CCE mainly focuses on advanced subjects like advanced programming technologies (python and ruby), embedded systems, and network technologies. IT too is similar to CCE just that it has more of application oriented subjects.

How is CCE different from IT?

All the three branches(CCE,IT and CSE ) are computer science related and there is no much difference when it comes to the basics. All the three branches students will be learning JAVA, Data Structures and algorithms in their 2nd year. The main difference comes in 3rd Year. CSE focuses on subjects like Formal language and automata theory, database systems and microprocessors. CCE mainly focuses on advanced subjects like advanced programming technologies (python and ruby), embedded systems, and network technologies. IT too is similar to CCE just that it has more of application oriented subjects.

What is the difference between CCE and IT?

All the three branches(CCE,IT and CSE ) are computer science related and there is no much difference when it comes to the basics. All the three branches students will be learning JAVA, Data Structures and algorithms in their 2nd year. The main difference comes in 3rd Year. CSE focuses on subjects like Formal language and automata theory, database systems and microprocessors. CCE mainly focuses on advanced subjects like advanced programming technologies (python and ruby), embedded systems, and network technologies. IT too is similar to CCE just that it has more of application oriented subjects.

Diffirence between CCE and IT?

All the three branches(CCE,IT and CSE ) are computer science related and there is no much difference when it comes to the basics. All the three branches students will be learning JAVA, Data Structures and algorithms in their 2nd year. The main difference comes in 3rd Year. CSE focuses on subjects like Formal language and automata theory, database systems and microprocessors. CCE mainly focuses on advanced subjects like advanced programming technologies (python and ruby), embedded systems, and network technologies. IT too is similar to CCE just that it has more of application oriented subjects.

How are the placements for CCE?

Placement is good. All eligible students are getting placed in good companies.

What is the scope in the government sector?

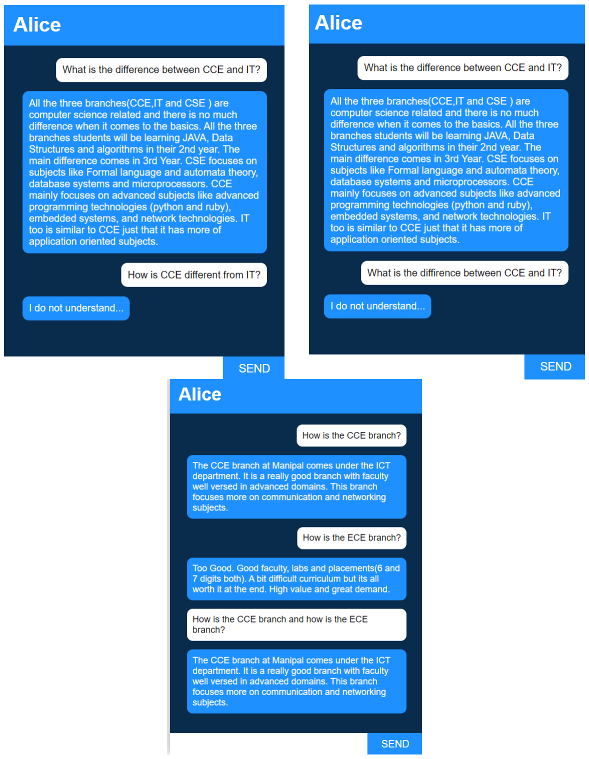
CCE students get equal opportunities in government sector.

How are the placements for CCE and what is the scope in the government sector?

placement is good, all eligible students are getting placed in good companies.

cce students get equal opportunities in government sector.

FIGURE 10. Interface of Hercules



What is the difference between CCE and IT?

All the three branches(CCE,IT and CSE ) are computer science related and there is no much difference when it comes to the basics. All the three branches students will be learning JAVA, Data Structures and algorithms in their 2nd year. The main difference comes in 3rd Year. CSE focuses on subjects like Formal language and automata theory, database systems and microprocessors. CCE mainly focuses on advanced subjects like advanced programming technologies (python and ruby), embedded systems, and network technologies. IT too is similar to CCE just that it has more of application oriented subjects.

How is CCE different from IT?

I do not understand...

What is the difference between CCE and IT?

All the three branches(CCE,IT and CSE ) are computer science related and there is no much difference when it comes to the basics. All the three branches students will be learning JAVA, Data Structures and algorithms in their 2nd year. The main difference comes in 3rd Year. CSE focuses on subjects like Formal language and automata theory, database systems and microprocessors. CCE mainly focuses on advanced subjects like advanced programming technologies (python and ruby), embedded systems, and network technologies. IT too is similar to CCE just that it has more of application oriented subjects.

What is the diffirence between CCE and IT?

I do not understand...

How is the CCE branch?

The CCE branch at Manipal comes under the ICT department. It is a really good branch with faculty well versed in advanced domains. This branch focuses more on communication and networking subjects.

How is the ECE branch?

Too Good. Good faculty, labs and placements(6 and 7 digits both). A bit difficult curriculum but its all worth it at the end. High value and great demand.

How is the CCE branch and how is the ECE branch?

The CCE branch at Manipal comes under the ICT department. It is a really good branch with faculty well versed in advanced domains. This branch focuses more on communication and networking subjects.

FIGURE 11. Interface of Alice

Considering Sam as the base chat-bot, it can be observed that Alice (pattern matching) and Big Mouth (TF-IDF vectorization) might not be as effective as neural network-based models, and they require a deeper understanding of the technologies to obtain an acceptable model.

### 6) Time Complexity of Chatbots

The time complexity of chatbots implemented using neural networks (NN) and natural language processing (NLP) can vary depending on the specific architecture, algorithms, and models employed. Let's break down the time complexity for different components:

- Natural Language Processing (NLP): O(n)
- Neural Networks (NN): O(e * n * h)
- Response Generation: O(1)

## VI. CONCLUSION

Engineering colleges follow an online admission process that involves a counseling process for engineering stream selection. During the counseling phase, students and parents have many queries regarding the branches offered by the college and many other such queries. These questions can be answered by visiting the college or over a phone call. The volume of the queries can be overwhelming, and due to this, there might be some miscommunication of information or many times, the officials might be busy on other calls. Students might have to rely on unofficial sources like Quora to get information. Furthermore, the students have to navigate through the entire website for data which can be tedious.

In this paper, five chatbot models were created using neural networks, TF-IDF vectorization, and pattern matching. In neural network-related models, pre-processing steps like converting to lowercase, stemming, lemmatization, tokenization, removing stop words, and creating a "bag of words" are applied to the training data before passing it through the neural network. A query is taken from the user; pre-processing steps are used to it, and it is passed through the model, which returns the list of probabilities that the query belongs to a certain intent.

Hercules performs best among the five chat-bots discussed in the project because it has sequential modeling designed to prevent overfitting training data. Furthermore, it is the only chat-bot with any optimizer applied to it, improving its performance. Therefore, it can be concluded that a chatbot similar to Hercules can be implemented in real-time for university/institute counseling. This will be very helpful for the students because it can provide official and accurate results. Furthermore, it can give 24x7 assistance, and the information provided will be uniform. Lastly, students will be able to rely on reliable information sources to resolve their queries.

Future work for integrating ChatGPT into counseling could focus on enhancing emotional intelligence, enabling dynamic learning and adaptation, exploring multimodal interactions, addressing privacy concerns, ensuring cultural sensitivity, integrating with counseling resources, implementing a continuous user feedback mechanism, prioritizing accessibility features, optimizing scalability for concurrent interactions, and conducting rigorous evaluation studies for efficacy validation. These developments aim to make ChatGPT a more personalized, responsive, and inclusive tool in the counseling process.

## REFERENCES

[1] T. Lalwani, S. Bhalotia, A. Pal, V. Rathod, and S. Bisen, "Implementation of a chatbot system using ai and nlp," *International Journal of Innovative Research in Computer Science & Technology (IJIRCST) Volume-6, Issue-3*, 2018.

[2] J. Thukrul, A. Srivastava, and G. Thakkar, "Doctorbot-an informative and interactive chatbot for covid-19," *International Research Journal of Engineering and Technology (IRJET)*, vol. 7, no. 07, pp. 2395–0072, 2020.

[3] S. Maher, S. Kayte, and S. Nimbhore, "Chatbots & its techniques using ai: an review," *International Journal for Research in Applied Science and Engineering Technology*, vol. 8, no. 12, pp. 503–508, 2020.

[4] M. Aleedy, H. Shaiba, and M. Bezbradica, "Generating and analyzing chatbot responses using natural language processing," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 9, 2019.

[5] P. Qi, Y. Zhang, Y. Zhang, J. Bolton, and C. D. Manning, "Stanza: A python natural language processing toolkit for many human languages," *arXiv preprint arXiv:2003.07082*, 2020.

[6] M. M. H. Dihyat and J. Hough, "Can rule-based chatbots outperform neural models without pre-training in small data situations?: A preliminary comparison of aiml and seq2seq," in *Proceedings of the 25th Workshop on the Semantics and Pragmatics of Dialogue*, 2021.

[7] A. Chandan, M. Chattopadhyay, and S. Sahoo, "Implementing chat-bot in educational institutes," *IJRAR Journal*, vol. 6, no. 2, pp. 44–47, 2019.

[8] D. Davis and J. Smith, "The potential of chatbots in counseling," *J. Counsel.*, vol. 5, no. 2, pp. 123–136, Apr. 2019.

[9] R. Johnson and S. Lee, "Chatbots providing emotional support to engineering students," in *Proc. IEEE Eng. Educ. Conf.*, Austin, TX, USA, 2020, pp. 45–50.

[10] A. Patel, B. Author2, and C. Author3, *Personalized Career Guidance for Engineering Students*. Springer, 2021.

[11] Y. Chang and W. Wang, "Privacy concerns in counseling chatbots," in *Proc. IEEE Int. Conf. Comput. Commun.*, Paris, France, 2018, pp. 234–239.

[12] H. Yang and Q. Liu, "User acceptance of counseling chatbots," *J. Comput.*, vol. 8, no. 3, pp. 210–225, May 2019. [Online]. Available: https://www.example-url.com

[13] B. R. Ranoliya, N. Raghuwanshi, and S. Singh, "Chatbot for university related faqs," in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2017, pp. 1525–1530.

[14] V. Sharma, M. Goyal, and D. Malik, "An intelligent behaviour shown by chatbot system," *International Journal of New Technology and Research*, vol. 3, no. 4, p. 263312, 2017.

[15] E. Adamopoulou and L. Moussiades, "Chatbots: History, technology, and applications," *Machine Learning with Applications*, vol. 2, p. 100006, 2020.

[16] S. Khan and M. R. Rabbani, "Artificial intelligence and nlp-based chatbot for islamic banking and finance," *International Journal of Information Retrieval Research (IJIRR)*, vol. 11, no. 3, pp. 65–77, 2021.

[17] C. Curry and J. D. O'Shea, "The implementation of a story telling chatbot," *Advances in Smart Systems Research*, vol. 1, no. 1, p. 45, 2012.

[18] V. Tiwari, L. K. Verma, P. Sharma, R. Jain, and P. Nagrath, "Neural network and nlp based chatbot for answering covid-19 queries," *International Journal of Intelligent Engineering Informatics*, vol. 9, no. 2, pp. 161–175, 2021.

[19] S. S. Ranavare and R. Kamath, "Artificial intelligence based chatbot for placement activity at college using dialogflow," *Our Heritage*, vol. 68, no. 30, pp. 4806–4814, 2020.

[20] L. Fryer and R. Carpenter, "Bots as language learning tools," *Language Learning & Technology*, vol. 10, no. 3, pp. 8–14, 2006.

[21] S. Verma, L. Sahni, and M. Sharma, "Comparative analysis of chatbots," in *Proceedings of the International Conference on Innovative Computing & Communications (ICICC)*, 2020.

[22] A. N. Mathew, V. Rohini, and J. Paulose, "Nlp-based personal learning assistant for school education," *Int. J. Electr. Comput. Eng*, vol. 11, no. 5, pp. 4522–4530, 2021.

[23] M. Mittal, G. Battineni, D. Singh, T. Nagarwal, and P. Yadav, "Web-based chatbot for frequently asked queries (faq) in hospitals," *Journal of Taibah University Medical Sciences*, vol. 16, no. 5, pp. 740–746, 2021.

[24] Q. N. Nguyen, A. Sidorova, and R. Torres, "User interactions with chatbot interfaces vs. menu-based interfaces: An empirical study," *Computers in Human Behavior*, vol. 128, p. 107093, 2022.

[25] S. Han and M. K. Lee, "Faq chatbot and inclusive learning in massive open online courses," *Computers & Education*, vol. 179, p. 104395, 2022.

[26] A. A. Qaffas, "Improvement of chatbots semantics using wit. ai and word sequence kernel: Education chatbot as a case study," *International Journal of Modern Education and Computer Science*, vol. 11, no. 3, p. 16, 2019.

[27] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[28] S. Qaiser and R. Ali, "Text mining: use of tf-idf to examine the relevance of words to documents," *International Journal of Computer Applications*, vol. 181, no. 1, pp. 25–29, 2018.

[29] I. Ahmed and S. Singh, "Aiml based voice enabled artificial intelligent chatterbot," *International Journal of u-and e-Service, Science and Technology*, vol. 8, no. 2, pp. 375–384, 2015.

[30] R. Rani and D. Lobiyal, "Automatic construction of generic stop words list for hindi text," *Procedia computer science*, vol. 132, pp. 362–370, 2018.

[31] D. Ofer, N. Brandes, and M. Linial, "The language of proteins: Nlp, machine learning & protein sequences," *Computational and Structural Biotechnology Journal*, vol. 19, pp. 1750–1758, 2021.

[32] G. Sperlí, "A cultural heritage framework using a deep learning based chatbot for supporting tourist journey," *Expert Systems with Applications*, vol. 183, p. 115277, 2021.

[33] S. D. Nithyanandam, S. Kasinathan, D. Radhakrishnan, and J. Jebapandian, "Nlp for chatbot application: Tools and techniques used for chatbot application, nlp techniques for chatbot, implementation," in *Deep Natural Language Processing and AI Applications for Industry 5.0*. IGI Global, 2021, pp. 142–168.

[34] A. Jaglan, D. Trehan, U. Megha, and P. Singhal, "Covid-19 trend analysis using machine learning techniques," *Int J Sci Eng Res*, vol. 11, no. 12, pp. 1162–1167, 2020.

[35] M. A. Al Muid, M. M. Reza, R. B. Kalim, N. Ahmed, M. T. Habib, and M. S. Rahman, "Edubot: An unsupervised domain-specific chatbot for educational institutions," in *Artificial Intelligence and Industrial Applications: Artificial Intelligence Techniques for Cyber-Physical, Digital Twin Systems and Engineering Applications*. Springer, 2021, pp. 166–174.

[36] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

SUCHETA KOLEKAR Dr. Sucheta V. Kolekar is currently working as an Associate Professor in the Department of Information and Communication Technology, MIT, Manipal Academy of Higher Education, Manipal, Karnataka, India. She has around 15 years of experience in the field of teaching and research. She holds a PhD in the area of Adaptive E-Learning from Manipal Academy of Higher Education, Manipal, Karnataka, India. Her primary research interests include E-learning, Web Usage Mining, Human Computer Interaction, Serious Game Development and Cloud Computing. She has published more than 20 papers in national and international journals/conference proceedings. She has received E-learning Excellence Award in 2017 by Academic Conferences International for her research work in Adaptive E-learning. She along with her student team have designed and developed novel browser extension to capture the usage data of online courses which are provided by Coursera. Dr. Sucheta Kolekar is one of the inventor for the patent called "Smart Sole based diabetic foot ulcer prediction system" which is granted by Chennai Patent Office, India. She handles additional responsibility at the institute to promote and enhance innovation and entrepreneurship culture.

GIRIJA ATTIGERI Dr. Girija Attigeri has 18 years of teaching and research experience in reputed institutes of Karnataka. She is currently working as Associate Professor in the Department of Information and Communication Technology, Manipal Institute of Technology, Manipal. Her research interests span big data analytics, Artificial Intelligence, Machine Learning Deep Learning and Semantic Web. She has 16+ publications in reputed international conferences and journals. She has conducted several seminars and workshops on her big data and machine learning. She is working on several projects related to data analytics in health care, education and agriculture. Dr. Girija has received B.E. and M. Tech. degrees from the Visvesvaraya Technological University, Karnataka, India. She has received his Ph.D. from the Manipal Institute of Technology Manipal Academy of Higher Education, Manipal.

ANKIT AGRAWAL Ankit Agrawal is working professional is the IT industry. He completed his bachelors degree in Computer and Communication Engineering from Manipal Institute of Technology, Manipal where he got the opportunity to write a research paper and create a project that will solve a real world problem for colleges and it's applicants. He has always been intrigued with the concept of Artificial Intelligence and was is grateful to the college and his mentors to provide him with the freedom and guidance to implement the project and use his capabilities to the maximum.