Deep Learning: College Enquiry chat-bot

M.SATHYAM REDDY
Asst Professor
Computer Science and Engineering
Narasaraopeta Engineering College
(Autonomous)
Narasaraopet, Andhra Pradesh

Surya Teja Penugonda
Student
Computer Science and Engineering
Narasaraopeta Engineering College
(Autonomous)
Narasaraopet, Andhra Pradesh
psuryateja01@gmail.com

Angalakurthy Achyuth
Student
Computer Science and Engineering
Narasaraopeta Engineering College
(Autonomous)
Narasaraopet, Andhra Pradesh
achyuth986@gmail.com

Tubati Uma Sankar Aditya
Student
Computer Science and Engineering
Narasaraopeta Engineering College
(Autonomous)
Narasaraopet, Andhra Pradesh
adityaaditya4043@gmail.com

Abstract—
Every year, newcomers and their parents turn to our college website seeking clarification on various matters. Similarly, existing students frequently visit the website to resolve their queries. Recognizing this recurring need, we have devised an innovative solution – an 'Intelligent Enquiry Bot' to seamlessly integrate with our official college website. This bot is meticulously crafted to adeptly handle inquiries from freshers, parents, students, and faculty members alike. This web-based application delivers precise responses to user inquiries by harnessing the power of Natural Language Processing (NLP) and DL networks. Our paper elucidates the development of this sophisticated chat-bot, engineered with NLP and DL technologies, tailored specifically to address fundamental college-related queries and, notably, admission related concerns.
Keywords—component, formatting, style, styling, insert (key words)

I. Introduction

A chat-bot is a clever computer program that communicates with people. A chat-bot operates in a manner akin to that of human chatters. Its main responsibility is to assist consumers by answering their inquiries, figuring out what they want, and guiding them toward the solution they want. These days, different chat-bots are in charge of handling a variety of business-related duties to enhance client experiences in a wide range of industries [6], including banking, insurance, e-commerce, healthcare, and many more.

Natural Language Processing (NLP)[4] is a technique used by Deep Learning chat-bots to map user inputs to specific intents. In order to send a prepared answer, it will Categorize messages
The chat-bot develops into an intelligent software component with the ability to process, understand, and respond in natural language. Typically, when creating a chatbot, we utilize unique data
The primary goal of utilizing natural language processing (NLP) to build a chat-bot is to construct one that needs little or no human involvement.

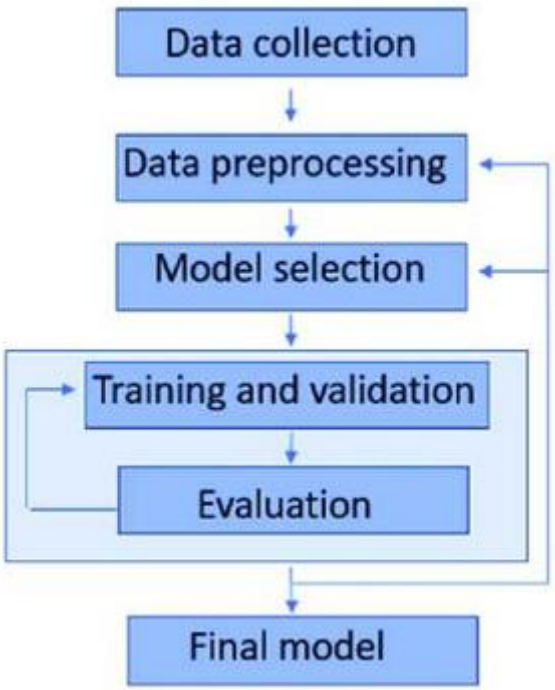


Fig1 Flow Chart

II. LITERATURE SURVEY

He focussed with the use of deep learning and natural language processing methods including CNN, GRU, and LSTM, this project aims to develop an interactive AI agent for customer service. Through sequence-to-sequence learning, it seeks to automatically produce answers to consumer inquiries Proposed System [1].

In the realm of aquaculture, the chat-bot in the suggested system acts as a helper to promote user involvement. It makes use of natural language processing (NLP) and IOT methods to comprehend customer inquiries and deliver pertinent data or help concerning the aquaculture system. chat-bots can be used by users to ask questions regarding system status, water quality parameters, or recommendations for sustaining ideal conditions. Through chat-bot integration, users may quickly and easily obtain up to-date information and assistance, improving the aquaculture [2] monitoring system's overall usability and efficacy.

The text emphasizes the significance of user interfaces, particularly chat-bot-based conversational interfaces, in software applications. By giving prompt and helpful answers to questions about a range of college-related topics, including placements, academics, and admissions, it highlights the efficacy and efficiency of chat-bots in improving user experience on college websites. These chatbots provide smart and engaging user interactions by utilizing AI and NLP algorithms [4]. The creation of a flexible chat-bot for a range of business requirements is overseen by Madana Mohana. Customers and organizations can communicate more easily thanks to it, which boosts customer happiness and streamlines operations. Its simple design encourages mutual success by making it easier for people to communicate and understand one another, which eventually increases everyone's level of pleasure [5]. Apart from its diagnostic functionalities, CUDoctor also makes medical expertise more accessible by enabling smooth communication between users and healthcare providers. It guarantees proper interpretation of user inputs, resulting in more accurate diagnoses, by utilizing fuzzy logic and natural language processing. In addition, the system's reach is increased by the integration of Twilio API and Telegram Bot API, which allow for SMS and messaging platform interactions. All things considered, CUDoctor is a major development in telehealth technology, providing a practical and efficient means of diagnosing illnesses and providing medical assistance [7]. The goal of this project is to create a chat-bot that can properly read user inquiries and react to them about colleges by applying algorithms. Students spend less time looking for information because the web application, called chat-bot, responds to their inquiries quickly. The solution makes sure that data management and retrieval are done efficiently by integrating MySQL with CodeIgniter (a PHP framework). It also has an online board where users may view pertinent announcements and papers, which improves accessibility and ease of usage. Important terms: MySQL, CodeIgniter, chatbot, Knowledge Database.[8]

It was suggested to create an AI-powered online support group that automatically clusters patient data using deep learning NLP. We took and examined patient behaviors, demographics, choices, feelings, clinical aspects, and social contact. Online support groups get patient-centered information about all medical diseases and ailments, and a bot known as One bot responds to the posts. OneBot offers excellent user support along with an intuitive UI that leverages natural language. The monitor unit can be calculated by the software, which is compatible with Windows and Android operating systems. It makes use of the inherent qualities of the Internet, luding real-time communication, Internet access, data exchange, and transmission.[11]

III METHODOLOGY

Research Objective: The primary objective of this research is to employ Deep learning and NLP[2] for creating the chatbot which will be helpful for college administration and students

A)Data Collection and Processing:

I have gathered the dataset from kaggle which contains 500 lines of data which is present in the form of tags, questions, and responses with file name intents. Json

```
#Used in Tensorflow Model
import numpy as np
import tensorflow.compat.v1 as tf
tf.disable_v2_behavior()
import tflearn
import random

#Usde to for Contextualisation and Other NLP Tasks.
import nltk
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
from nltk.stem.lancaster import LancasterStemmer
stemmer = LancasterStemmer()

#Other
import json
import pickle
import warnings
warnings.filterwarnings("ignore")
```

Fig-2 Importing Modules and packages

Explanation of some more modules imported above are

1)Numpy: A core Python library for scientific computing is called NumPy. It offers support for matrices and multidimensional arrays, as well as a number of mathematical operations that may be performed on these arrays. Frequently used as an alias for NumPy, "np" makes it simpler to refer to NumPy objects and functions within code.

2) TensorFlow (tf): Google created this open-source machine learning framework. It is frequently utilized for many different tasks, including neural network construction and training. The import statement imports TensorFlow 1.x using the alias "tf" and imports TensorFlow.compat.v1. TensorFlow 2.x compatibility is ensured with the use of compat.v1.

3)TensorFlow Learn (tflearn): A high-level TensorFlow-based API, TensorFlow Learn (TFLearn) is also referred to as TFLearn. By offering a greater degree of abstraction, it

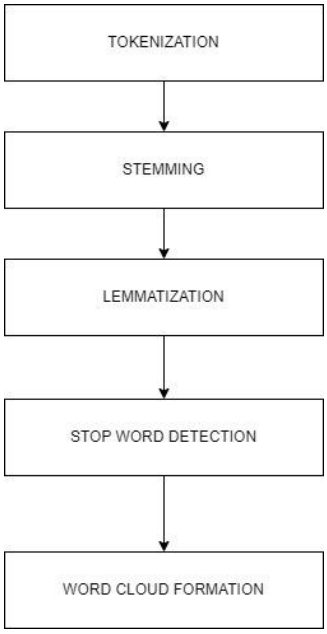
seeks to make the process of creating and training neural networks more straightforward. For high-level TensorFlow APIs, users are advised to utilize Keras or another alternative as it has been deprecated since TensorFlow 2.0.

4) Natural Language Toolkit, or NLTK (nltk), is a popular framework for developing Python applications that interact with human language data. With its set of text processing tools, it offers user-friendly interfaces to more than 50 corpora and lexical resources, including WordNet, for activities like tokenization, stemming, tagging, parsing, and semantic reasoning.

5)WordNetLemmatizer: Words are lemmatized, or reduced to their most basic or canonical form, using the WordNetLemmatizer class. Lemmatizing "running" for instance would produce "run". NLTK offers an interface to access WordNet, an English lexical database.
6) Additional Imports: The standard Python libraries pickle, json, and warnings are used to handle warnings, serialize Python objects, and operate with JSON data, respectively. These modules offer features for reading and writing JSON files, serializing and deserializing Python objects, and managing the way warning messages Are handled while a program is running.

2)Preprocessing:

In the context of natural language processing (NLP), preprocessing refers to a set of procedures used to clean and convert unprocessed text data into a format better suited for modeling or analysis. These actions are essential for raising the caliber of the data and boosting the efficiency of later NLP assignments.
Preprocessing, where raw textual data is systematically cleaned and transformed to improve its usability for various analytical purposes, is a key step in the pipeline of natural language processing (NLP) operations. A number of procedures, including tokenization, lowercasing, stemming or lemmatization, stopword elimination, and normalization, are performed during this preliminary stage. Lowercasing unifies the text by making all characters lowercase, whereas tokenization splits the content into smaller units or tokens, such as words or characters.



1) TOKENIZATION
The act of tokenizing [4]a document involves dividing it up into smaller units, like words or phrases. It's an essential phase in natural language processing that enables computers to comprehend and interpret language spoken by people. In order to facilitate processing and analysis, a text might be divided into its constituent parts. Tokenization would break down a sentence like "The quick brown fox jumps over the lazy dog" into individual words like "the," "quick," "brown," "fox," "jumps," "over," "the," "lazy," and "dog." Numerous NLP tasks, such as text classification, sentiment analysis, and machine translation, are based on this method.

```
words = []
classes = []
documents = []
ignore_words = ['?', '_', '!', '-']
print("Looping through the Intents to Convert them to words, classes, documents and ignore_words.....")
for intent in intents['intents']:
    for pattern in intent['patterns']:
        # tokenize each word in the sentence
        w = nltk.word_tokenize(pattern)
        # add to our words list
        words.extend(w)
        # add to documents in our corpus
        documents.append((w, intent['tag']))
        # add to our classes list
        if intent['tag'] not in classes:
            classes.append(intent['tag'])
```

Fig-2 Code for Tokenization

```
Looping through the Intents to Convert them to words
Words:
Hi
Hello
How
are
you
?
how
was
your
day
Goodbye
Bye
See
you
later
Talk
to
you
later
Who
created
you
?
...
and
personal
life
?
```

Tokenization Output

2) STEMMING

A natural language processing method called stemming is used to break down words into their most basic form, or stem. The main objective of stemming is to standardize words such that, for analysis or retrieval purposes, various forms of the same word—such as plurals or verb tenses—are regarded as interchangeable. Stemming is the process of removing word affixes while attempting to retain as much of the original meaning as feasible. Prefixes, suffixes, and infixes may be eliminated during this procedure, leaving the word in a more basic form. Particularly helpful for activities like text mining is stemming.information retrieval in computational linguistics, where it lowers the vocabulary size and unifies word variants into a single representation to enhance the accuracy and efficiency of text processing. Although stemming algorithms are often straightforward and efficient, their performance can fluctuate between languages and settings and they may not always yield linguistically acceptable stems. Therefore, when applying stemming algorithms to real-world language processing tasks, it is crucial to take into account their unique requirements and limits.

```
print("Stemming, Lowering and Removing Duplicates.....")
words = [stemmer.stem(w.lower()) for w in words if w not in ignore_words]
words = sorted(list(set(words)))

# remove duplicates
classes = sorted(list(set(classes)))

print (len(documents), "documents")
print (len(classes), "classes", classes)
print (len(words), "unique stemmed words", words)
```

Fig-3 Stemming Code

```
Stemmed Words:
hi
hello
how
ar
you
how
was
yo
day
goodby
bye
see
you
lat
talk
to
you
lat
who
cre
you
who
is
yo
...
academ
and
person
lif
```

Stemming Output

Lowering

One typical preprocessing step in deep learning (DL) and natural language processing (NLP) tasks is lowercasing or lowering words. This method entails changing every character in a word to lowercase. Lowercasing words serves to maintain consistency and uniformity in text data because distinct documents, sentences, or tokens may have varying capitalization.Lowercasing unifies words with disparate capitalizations into a unified representation, which helps to minimize the vocabulary size and complexity of the text data. Furthermore, by lowering the dimensionality of input characteristics and lessening the effect of case sensitivity in text classification, sentiment analysis, machine translation, and other NLP tasks, lowercasing can enhance the performance of DL models.

```
lowered_words = [w.lower() for w in words if w not in ignore_words]
print("Lowered Words:")
for word in lowered_words:
    print(word)
✓ 0.0s
```

Lowering code

```
Lowered Words:
hi
hello
how
are
you
how
was
your
day
goodbye
bye
see
you
later
talk
to
you
later
who
created
you
who
is
your
...
academics
and
personal
life
```

Lowering Output

Removing duplicates

One frequent data preprocessing step used in several domains, such as natural language processing (NLP) and deep learning (DL), is the removal of duplicates. This procedure entails finding and removing duplicates of the same data point from a dataset.

Eliminating duplicates reduces redundancy and potential biases in the dataset by guaranteeing that each unique piece of information is represented only once in the context of text data in natural language processing. where the performance of the model might be impacted by duplicate instances, which can distort the training process.

Eliminating duplicates is crucial in deep learning (DL) to maximize computational resources and boost training effectiveness. The model gets exposed to a wider variety of examples by removing duplicates, which improves its capacity to generalize to new data.

```
duplicated_words = sorted(list(set([w for w in words if words.count(w) > 1])))
print("Duplicated Words:")
for word in duplicated_words:
    print(word)
```

Removing Duplicates Code

```
's
,
.
?
Are
Can
Do
Does
Greek
How
I
Is
Tell
What
Where
Which
Who
WiFi
a
about
abroad
academic
access
accessible
...
would
writing
you
your
```

3) STOP WORD DETECTION:

In deep learning-based natural language processing (NLP) is the process of locating and eliminating stop words from textual data. Stop words are often occurring words in a language that are frequently filtered out during the text preprocessing stage because they usually don't add much to the text's understanding and don't carry much significance. English stop words include "the," "is," "and," "of," and so forth.

```
import re
from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))
print("Removing unnecessary spaces, special symbols, numbers, and stopwords")
words = [re.sub(r'[^a-zA-Z]', ' ', w).strip() for w in words if w.lower() not in stop_words]
words = sorted(list(set(filter(None, words))))
words = [word for word in words if not any(char.isdigit() for char in word)]
classes = sorted(list(set(classes)))
print(len(words), "words obtained after unnecessary things")
print("words are",words)
print(classes)
print(documents)
```

Fig-4 Stop word detection

4)WORD CLOUD FORMATION

A word cloud is a graphic representation of text data in which each word's size reflects how frequently or how important it occurs in the dataset. Word clouds do this by showing words in varied sizes, with the words that occur most frequently looking larger or bolder. This helps to illustrate the relative significance of various keywords or concepts in the text. It is simple to quickly recognize the main ideas, subjects, or trends in the text thanks to this graphical depiction. Word clouds are frequently used to swiftly and intuitively summarize vast amounts of text and derive relevant insights in domains including data analysis, text mining, and information visualization.

Fig-5 word cloud Formationn

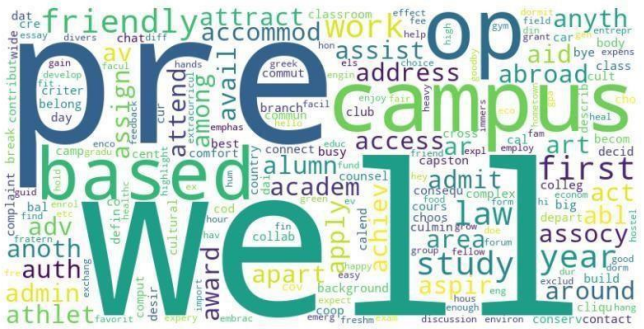


Fig-5 word cloud Formation

5)N-Gram analysis

In natural language processing, n-gram analysis is a technique used to examine the frequency and arrangement of consecutive sequences of n elements in a given text. Words, characters, or other language units can be among these things.

For instance, each word in the sentence "The quick brown fox jumps over the lazy dog" would be considered a unigram (1-gram), pairs of adjacent words would be bigrams (2-grams), and sequences of three consecutive words would be trigrams (3-grams).

Language modeling, text production, information retrieval, stylometry, and spelling correction are among the applications of n-gram analysis. It helps to comprehend linguistic phenomena and enables a variety of applications for natural language processing by offering insights into the structure and patterns of text data.

Bi gram analysis

A bigram, sometimes referred to as a 2-gram, is a group of two neighboring items in a text or dataset that are tokens, usually words or characters.

Bigrams are frequently used in natural language

processing (NLP) for a variety of tasks, including text analysis, feature extraction, and language modeling.

In conclusion, bigrams are essential to natural language processing (NLP) because they can represent the relationships and context of neighboring components in a sequence, which allows for a variety of modeling and text analysis tasks.

```
#generating bigram
n = 2
bigrams = ngrams(words, n)
# Count the occurrences of each bigram
bigram_counts = Counter(bigrams)

# Print the bigram frequencies
for bigram, count in bigram_counts.items():
    print(f"{bigram}: {count} times")
```

Bi Gram code

```
('abl', 'abroad'): 1 times
('abroad', 'academ'): 1 times
('academ', 'access'): 1 times
('access', 'accommod'): 1 times
('accommod', 'achiev'): 1 times
('achiev', 'act'): 1 times
('act', 'address'): 1 times
('address', 'admin'): 1 times
('admin', 'admit'): 1 times
('admit', 'adv'): 1 times
('adv', 'aid'): 1 times
('aid', 'alumn'): 1 times
('alumn', 'among'): 1 times
('among', 'anoth'): 1 times
('anoth', 'anyth'): 1 times
('anyth', 'apart'): 1 times
('apart', 'apply'): 1 times
('apply', 'ar'): 1 times
('ar', 'area'): 1 times
('area', 'around'): 1 times
('around', 'art'): 1 times
('art', 'aspir'): 1 times
('aspir', 'assign'): 1 times
('assign', 'assist'): 1 times
('assist', 'assocy'): 1 times
```

Trigram analysis:

Trigram analysis, often called 3-gram analysis, is a technique used in computational linguistics and natural language processing (NLP) to look at word or character sequences that include three consecutive elements in a text or dataset. By taking into account the links between three successive parts, trigrams offer even more context than bigrams (pairs of adjacent words) or unigrams (single words).

To sum up, trigram analysis benefits natural language processing (NLP) activities by offering a more comprehensive contextual understanding by analyzing the sequences of three adjacent items. This allows for more advanced text analysis, modeling, and interpretation.

```
#generating trigram
n = 3

# Generate trigrams
trigrams = ngrams(words, n)

# Count the occurrences of each trigram
trigram_counts = Counter(trigrams)

# Print the trigram frequencies
for trigram, count in trigram_counts.items():
    print(f"{trigram}: {count} times")
```

```
('Hi', 'Hello', 'Hey'): 1 times
('Hello', 'Hey', 'Good'): 1 times
('Hey', 'Good', 'day'): 1 times
('Good', 'day', 'How'): 1 times
('day', 'How', 'are'): 1 times
('How', 'are', 'you'): 1 times
('are', 'you', '?'): 2 times
('you', '?', 'how'): 1 times
('?', 'how', 'was'): 1 times
('how', 'was', 'your'): 1 times
('was', 'your', 'day'): 1 times
('your', 'day', 'Goodbye'): 1 times
('day', 'Goodbye', 'Bye'): 1 times
('Goodbye', 'Bye', 'See'): 1 times
('Bye', 'See', 'you'): 1 times
('See', 'you', 'later'): 1 times
('you', 'later', 'Talk'): 1 times
('later', 'Talk', 'to'): 1 times
('Talk', 'to', 'you'): 1 times
('to', 'you', 'later'): 1 times
('you', 'later', 'Who'): 1 times
('later', 'Who', 'created'): 1 times
('Who', 'created', 'you'): 1 times
('created', 'you', '?'): 1 times
('you', '?', 'Who'): 2 times
...
('balancing', 'academics', 'and'): 1 times
('academics', 'and', 'personal'): 1 times
('and', 'personal', 'life'): 1 times
('personal', 'life', '?'): 1 times
```

Bag of words representation

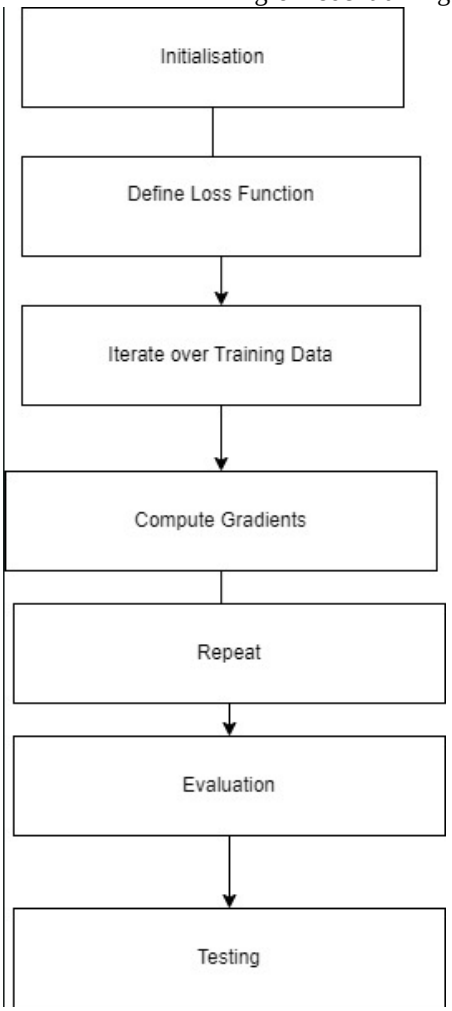
A key method in natural language processing (NLP) is the Bag-of-Words (BoW) representation, which transforms textual data into a numerical format appropriate for machine learning algorithms. A document is represented as a vector in the BoW model, with each element denoting the frequency or presence of a particular word in the document. The BoW method just considers the presence of individual words in the document, ignoring the word order.

B) Model Training :
When training a model with the Stochastic Gradient Descent (SGD) technique, the model's parameters are optimized through an iterative process. These parameters are initially initialized either with pre-trained weights or randomly. After that, the training data set is split up into smaller batches in order to maximize memory use and computational efficiency This procedure is carried out across a number of epochs, during which the model's performance is progressively enhanced via parameter updates that are directed by the SGD algorithm. To keep track of its development and avoid overfitting, the model's performance is assessed on a different validation data set during training.

```
print("Here i am using stochiastic Gradient Descent ALgorithm")
model_filename = 'model.tflearn'
print("Training the model...")

model.fit(train_x, train_y, n_epoch=1000, batch_size=8, show_metric=True)
print("Saving the trained model...")
model.save(model_filename)
print("Model saved successfully!")
```

Fig-6 model training



steps involved in above flow chart

Initialization: Apply specialized initialization procedures or randomize the model parameters to start. For neural networks, these parameters include weights and biases.
Iterate over Training Data: To maintain randomness and stop the model from picking up patterns depending on the sequence in which the data samples are presented, shuffle the training data. Next, cycle over the training dataset using individual samples or mini-batches. With SGD, a single training example is used for each iteration to update the model's parameters.

Determine Gradients: For every training instance or mini-batch:

Determine the model's predictions based on the input data. Utilizing the selected loss function, calculate the loss by contrasting the model's predictions with the actual targets. Using backpropagation, calculate the gradients of the loss function with respect to the model parameters. This entails calculating the gradient of the loss function with respect to each parameter and propagating the mistake backward through the network.
Repeat: Until a stopping criterion is satisfied, keep iterating through the training dataset, calculating gradients, and adjusting model parameters. A maximum number of iterations, convergence depending on validation performance, or other user-specified parameters could be used as this criterion.

Assessment and Validation: To keep track of the model's development and avoid overfitting, periodically assess its performance on a different validation set. Based on the outcomes of the validation process, modify hyperparameters like regularization and learning rate.

Testing: Lastly, assess the trained model using a different test set that wasn't utilized for validation or training. In this step, the effectiveness of the model in real-world scenarios is estimated and its generalization performance is evaluated.

These steps describe how to use stochastic gradient descent to train a model.

C)Model Testing:
A critical step in the Deep learning process, model testing determines how well a trained model performs. Using a different data set from the one used for training and validation is essential to this procedure. An objective assessment of the model's functionality and capacity to generalize to new data is provided by this test data set.
Several assessment criteria are used in testing to determine how well the model performs in addressing the particular issue that it was trained to solve. When it comes to categorization activities, measures like accuracy[7] The entire data set divided into two parts the first part of data set will be used for training and second half of data set will be used for testing of data. And the values obtained are mentioned below

Table-1-Accuracy and Loss Values

	Accuracy	Loss
Training	98.79	0.014
Testing	98.04	0.02

The Table-1 represents the accuracy and loss percentage obtained during the training and testing of model

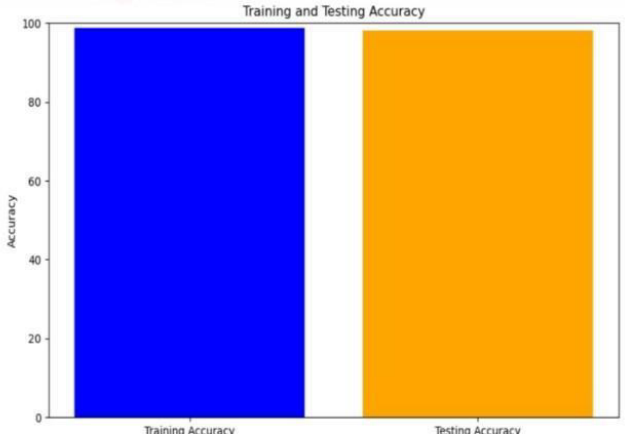


Fig-7 Training and testing Accuracy

The bar graph represents the accuracy obtained during the training and testing of the model

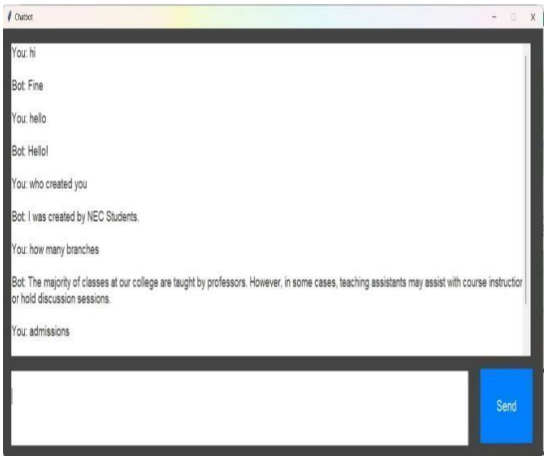
From the above we can say that the accuracy and loss of training and testing will be inversely proportional to each other

Fig-8 Training and testing loss

Fig-8 bar chart represents the loss obtained during Training and testing.Loss obtained during testing was some what greater compared to loss obtained during Training

Results and Discussion

There are a number of concrete advantages to using a chatbot to answer college-related questions. First off, it may significantly improve the user experience by giving prompt answers to questions and making college information easily accessible to potential students. The quick access to information has the potential to greatly raise satisfaction and engagement levels. Additionally, the chat-bot can manage a large number of inquiries at once by automating the inquiry process, which improves operational efficiency and lessens the workload for administrative workers. The chat-bot 24/7 accessibility, which guarantees that users may access



Conclusion

In summary, the creation of a chat-bot for college inquiry purposes that makes use of natural language processing (NLP) and deep learning (DL) is a noteworthy development in automating and improving the effectiveness of the college admissions process.

With the help of this paper, we have effectively shown that utilizing cutting-edge technologies to create an intuitive user interface that can comprehend and reply to Prospective students can obtain pertinent information about the college, courses offered, admissions processes, deadlines, campus amenities, and more instantly with the help of the chat-bot, which is a useful tool. The chatbot helps administrative personnel focus on more complicated tasks by optimizing the inquiry process, thereby reducing workload and increasing user satisfaction

References

[1] Aleedy, M., Shaiba, H. and Bezbradica, M., 2019. Generating and analyzing chat-bot responses using natural language processing. International Journal of Advanced Computer Science and Applications, 10(9).

[2] Al-Rasyid, M.U.H., ukaridhoto, S., Dzulqornain, M.I. and Rifai, A., 2020. Integration of IoT and chat-bot for aquaculture with natural language processing. TELKOMNIKA (Telecommunication Computing Electronics and Control), 18(2), pp.640-648.

[3] Echeazarra, L., Pereira, J. & Saracho, R. TensioBot: a chat-bot Assistant for Self-Managed in-House Blood Pressure Checking. J Med Syst 45, 54 (2021). https://doi.org/10.1007/s10916-021-01730-x

[4] Gbenga, L.O., Oluwafunto, O.T. and Oluwatobi, A.H., 2020. An improved rapid response model for university admission enquiry system using chat-bot. Int. J. Comput, 38(1), pp.123-131

[5] Lalwani, T., Bhalotia, S., Pal, A., Rathod, V. and Bisen, S., 2018. Implementation of a chat-bot System using AI and NLP. International International Journal of Innovative Research in Computer Science Technology (IJIRCST) Volume-6, Issue3.

[6] Madana Mohana, R., Pitty, N. and Lalitha Surya Kumari, P., 2021. Customer support chat-bot

using machine learning. In Intelligent Data Engineering and Analytics: Frontiers in Intelligent Computing: Theory and Applications (FICTA 2020), Volume 2 (pp. 445-451). Springer Singapore.

[7] Malvin, D. and Rangkuti, A.H., 2022. WhatsApp chat-bot Customer Service Using Natural Language Processing and Support Vector Machine. International Journal of Emerging Technology and Advanced Engineering, pp.130136.

[8] Omoregbe, N.A., Ndaman, I.O., Misra, S., Abayomi- Alli, O.O., Damasevi ˇ cius,R. and Dogra, A., 2020. Text messaging-based medical diagnosis using natural45 language processing and fuzzy logic. Journal of Healthcare Engineering, 2020,pp.1-14

[9] [Ms.Ch.Lavanya Susanna, R.Pratyusha, P.Swathi, P.Rishi Krishna, V.Sai Pradeep, "College Enquiry chat-bot",International Research Journal of Engineering and Technology (IRJET), e-ISSN: 2395- 0056, p-ISSN: 2395- 0072, Volume: 07 Issue: 3 Mar 2020 pp 784- 788 Issue03,March-2019, pp 109-112

[10] R. B. Mathew, S. Varghese, S. E. Joy and S. S. Alex, "chat-bot for Disease Prediction and Treatment Recommendation using Machine Learning," 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 2019, pp. 851856, doi:0.1109/ICOEI.2019.8862707. keywords: {Diseases;Natural language Processing;Medical iagnostic imaging;Hospitals;Medical chat-bot;Machine Learning;Disease Prediction;Treatment;KNN},

[11] Siddique, S. and Chow, J.C., 2021. Deep learning in healthcare communication. Encyclopedia, 1(1), pp.220-239.

[12] J. Weizenbaum, "Elizaa computer program for the study of natural language communication between man and machine," Communications of the ACM, vol. 9, no. 1, pp. 36–45, 1966.

[13] A. M. Neves, F. A. Barros, and C. Hodges, "Iaiml: A mechanism to treat intentionality in aiml chatterbots," in Tools with Artificial Intelligence, 2006. ICTAI'06. 18th IEEE International Conference on. IEEE, 2006, pp. 225–231.

[14] N. Thomas, "An e-business chatbot using aiml and lsa," in Advances in Computing, Communications and Informatics (ICACCI), 2016 International Conference on. IEEE, 2016, pp. 2740–2742.

[15] S. Reshmi and K. Balakrishnan, "Implementation of an inquisitive chatbot for database supported knowledge bases," Sadhan ¯ a¯, vol. 41, no. 10, pp. 1173–1178, 2016