Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Semester: (Spring, Year: 2025), B.Sc. in CSE (Day)

AI GuBbot

Course Title: Artificial Intelligence Lab
Course Code: CSE-316
Section: 221-D7

Students Details

Name	ID
Md. Jahid Hasan Jitu	221002248
MD RANA MIAH	221002217

Submission Date: 10/05/2025
Course Teacher's Name: Md. Sabbir Hosen Mamun

[For teachers use only: Don't write anything inside this box]

Lab Project Status	
Marks:	**Signature:**
Comments:	**Date:**

Contents

Chapter 1

Introduction

AI Chatbot using python programming language

1.1 Overview

Chatbots are now indispensable tools in automation, e-commerce, and modern customer service because they enable companies to instantly respond to user inquiries. The goal of this project is to create a chatbot with Python that uses neural networks and natural language processing (NLP). The chatbot uses a trained machine learning model to identify and react to various inputs, and it uses a structured intents.json file to classify user interactions.

1.2 Motivation

The goal of this project is to develop a useful and perceptive chatbot that can:

- Automate customer service to cut down on response times and deliver prompt responses to frequently asked questions.
- Boost User Engagement: By providing dynamic responses, websites and applications can improve user interaction.
- Investigate AI and NLP: Get practical experience with NLP, deep learning, and machine learning methods.
- Scalability and Adaptability: Create a chatbot that is simple to modify for various sectors, including healthcare, education, and retail.
- Developers can work with real-world AI applications while learning the basics of chatbot development by following this project.

1.3 Problem Definition

1.3.1 Components

The chatbot system is a complex network of important segments:

- Intents File (intents.json) – Saves tags, patterns, and responses beforehand so that developers can recognize them without any need to train the system.
- Natural Language Processing (NLP) Module – Conducts tokenization, lemmatization, and text pre-processing procedures.
- Machine Learning Model – This is actually a neural network that has been trained to perform the classification of the user intents through TensorFlow/Keras.
- User Interface – A front end GUI or command-line tool which is used by the users to interact with the chatbot.
- Response Generator – It collects user inputs, compares them with the already-stored responses, and then gives users the responses that closely match their inputs.

1.3.2 Program Flow

The chatbot demonstrates a structured procedure:

- User Input Processing
  - The chatbot sends message to the user in the form of text.
  - The input is cleaned, tokenized, and lemmatized for standardization.
- Intent Classification
  - The clean input is transferred in the form of a bag of words.
  - The clean input is converted into a bag of words style.
  - The neural network forecasts the most matching intent among the predefined categories.
  - The neural network predicts the most relevant intent from the predefined categories.
- Response Selection
  - The chatbot selects the respective answer from the intents.json file according to the indicated intent.
  - The chatbot returns the response to the user.
  - The chatbot chooses the response from the intents.json file which corresponds to the predicted intent.

  - The chatbot sends the response back to the user.

- Continuous Learning Improvement
  - Programmers are allowed to retrain the bot by entering additional training data.
  - Result improvement can be achieved through the inclusion of more training data.
  - Developers can update the intents.json file to improve response accuracy.
  - Additional training data can boost the chatbot's understanding of different inputs.

1.3.3 Security Measures:

Security is the most important issue. Just to make sure that it will be safe the follow security measures should be rolled out:

- Input Validation – Removing malicious code structures prior to executing user input can avoid injecting illicit substances.
- Rate Limiting – To prevent spam or denial of service, users' excessive requests will be restricted. Otherwise, they will be limited.
- Data Privacy – Minimizing data and ensuring encryption is key to achieve data security. Keep privacy at the forefront of your activities.
- Model Robustness – To tackle confounded inputs in a great manner, the model should be tutored on this subject and provide a bit of stuff that could go wrong in order to prevent mischaracterizations of the sentence.
- Error Handling – Runtime error "errors" are mostly caused by bugs. Therefore, force the business logic by, for example, making it handle the rest of the system, informing the client, and so on.
- Arranging the work in such a manner that these constituents are integrated leads to a clear program flow, and at the same time security is fully implemented in the chatbot, that way of working can be used as an example in practical scenarios.

1.4 Objectives

Instant messaging-based communication has been a part of every generation, but now with the advanced versions of AI technology, we are experiencing chatbot projects whose primary objective is to develop an intelligent conversational agent capable of understanding and responding to user queries using Natural Language Processing (NLP) and Machine Learning.The specific objectives include:

- Develop a Structured Intent Recognition System – The most straightforward approach to making spontaneous behavior more human-like is to use grammar in the rules. To proceed, customize your chatbot by creating intents.json files that organize users' inputs and then generate responses. Ensure the intended response is correct by encoding the user input into predetermined tags.
- Implement NLP Techniques for Text Processing – The Library produced reacts according to its constructed quality, for example, it tokenizes and returns the stemmed words as a list. What the JSON file categories cannot do in one shot, you can make the program do step by step with tokenization, lemmatization. [2]
- Build and Train a Neural Network Model – In the problem of classifying handwritten digits, there are still residual errors that the classifier makes, and the visualizations created show that the problem of overfitting is minor. The trai By achieving these targets, a chatbot shall become an effective, flexible, and secure conversational AI system that can automate interactions across various domains.

1.5 Application

The chatbot can be deployed in a wide range of applications across various industries, which are essentially meant to increase user interaction, automate the responses, and make the efficiency better. In customer service, the chatbot can be deployed on websites and mobile apps to handle FAQs, process orders and offer real-time support, thereby, minimizing the workload of human agents. In e-commerce, it can be a perfect guide through showing the users the products they are looking for thus, answering their inquiries about availability, and helping them through the buying process. Chatbot is also a perfect tool for educational institutions that can be used for student support, helping with enrollment information, course details, and academic queries. In addition to this, the health sector is also served by the chatbot.

Chapter 2

Design/Development/Implementation of the Project

2.1 Introduction

The chatbots have evolved as an essential part of user interaction today, where they play an important role in automating communication and enhancing the user experience. The chatbots are being used increasingly in a variety of field- from customer service and online shopping to healthcare services- as they are able to give quick, consistent, and round-the-clock help. This specific work is presenting the development of a chatbot with Natural Language Processing (NLP) and Neural Networks in Python. Despite its simplicity, the model is smart enough to solve complicated tasks.The main aim lies in the construction of a conversational bot which can recognise the user's question in natural language, can categorize it into defined label (intents) and generate responses using a hard-coded reply. With the help of techniques like tokenization, lemmatization, and the bag of words model, the chatbot can well understand the user's query. Such methods are quite effective for the model to respond to the users efficiently and thus meeting the users' demands. [1]

2.2 Project Details

The chatbot is the result of Python programming and a combination of different libraries such as NLTK, NumPy, TensorFlow, and Keras to construct a conversational agent that works. The whole idea of the project is encapsulated in an intent.json file, a kind of bot brain which has data in itself. This piece consists of many tags each representing a user's intention and the conversation patterns that can be followed by such person and their replies. Data preprocessing is the first step in the development of this project, which will be carried out with NLP techniques such as tokenization and lemmatization, to change the user input from its raw form into something that the machine learning model can digest. When a source is fed to a bag-of-words model, it is converted into numerical vectors, which serve as the input for a sequential neural network then. For improving learning and preventing overfitting, the neural network is assembled with dense and dropouts layers connected to a softmax output layer that makes the classification of

inputs to the intents as a piece of cake. After being trained, the model is exported to a different script for running the real-time user interface. It's when the user types a query that the chatbot takes over to catch the intent, and based on the user's request, the bot fetches the corresponding response from the intents file. The system also provides some functions such as cleaning the input, producing predictions, and fetching the right responses, all of which guarantee that the conversation is smooth and correct.

2.3 Implementation

The first step that the chatbot's implementation requires is to make several necessary changes, beginning with the creation of an intents.json file, which outlines various user intents, example input patterns, and their corresponding responses. Then, a Python script is created with the purpose of preprocessing the data by using Natural Language Processing (NLP) techniques, such as tokenization and lemmatization, with the help of the NLTK library. After that, the cleaned data is tokenized using the bag of words model, which then is going to be used as the input for training a neural network built with TensorFlow and Keras. The neural network uses the dense layers powered by the ReLU activation and the dropout layers to better the learning process and to diminish the overfitting. The output layer is activated using the softmax function to classify the user input into one of the categories defined as intents. Once the training is completed, the model is saved, and afterward a separate Python script is used to load the model at runtime and handle chat. The inline script checks what the user input is, uses the trained model to predict the intent, and then matches a response back to the intents file. [3]

Tools and libraries

The development of the chatbot project relies on a set of powerful tools and libraries that simplify the implementation of Natural Language Processing (NLP) and Machine Learning tasks:

- Python
- NLTK (Natural Language Toolkit)
- NumPy
- TensorFlow
- Keras
- JSON
- Pickle

2.4 Functionality Details

The chatbot system includes several core functionalities that work together to enable seamless and intelligent interaction between users and the application. These functionalities are:

- Intent Recognition
- Natural Language Processing (NLP)
- Bag of Words Conversion
- Response Generation
- Model Training and Prediction
- User Interaction Interface
- Expandability

Intent Recognition

The chatbot identifies the user's intent by analyzing input text and matching it with patterns defined in the intents.json file. This is done using a trained neural network that predicts the most likely intent based on the input

Natural Language Processing (NLP)

Through tokenization and lemmatization using the NLTK library, the chatbot processes raw user input to a standard format. This allows the system to recognize different variations of similar words and maintain consistency in intent recognition.

Bag of Words Conversion

The user's input is converted into a bag-of-words vector, a numerical representation that the neural network can process for classification

Response Generation

Based on the predicted intent, the chatbot selects an appropriate response from a predefined list in the intents file. This ensures accurate and context-aware replies.

Model Training and Prediction

A deep learning model built with TensorFlow/Keras is trained on the intent patterns and used to classify future inputs. This enables the chatbot to learn from training data and improve accuracy.

User Interaction Interface

A simple interface (web-based) allows users to input their queries and receive real-time responses from the chatbot.

Expandability

New intents, patterns, and responses can easily be added to the intents.json file, allowing the chatbot to grow and adapt to new domains without changing the core logic.

Chapter 3

Performance Evaluation

3.1 Simulation Environment/ Simulation Procedure

The simulation process specifies the process of that has been followed to validate the Chatbot and is in line with people after it is created. The process makes sure that in a real-time conversation each aspect of the system, for instance, from user input processing to response generation, runs well.

3.1.1 Prepare the Environment

Begin by setting up the Python environment and installing all necessary libraries such as nltk, numpy, tensorflow, keras, and pickle. Download essential NLTK packages like punkt and wordnet for tokenization and lemmatization.

3.1.2 Run the Training Script

Execute the training.py script, which loads the intents.json file, processes the input patterns, creates the bag-of-words model, and trains the neural network using TensorFlow/Keras. Once training is complete, the model and associated data (e.g., words, classes) are saved using Pickle.

3.1.3 Start the Chatbot Application

Launch the main chatbot script (e.g., chatbot.py), which loads the trained model, intents file, and other necessary components like the lemmatizer and tokenizer.

3.1.4 User Input Simulation

Begin interacting with the chatbot through a web-based UI. Enter sample inputs such as greetings (“hello”, “hi”), product inquiries (“do you have laptops?”), or farewells (“bye”).

3.1.5 Observe Responses

The chatbot processes the input, predicts the intent using the trained model, and returns a predefined response from the intents.json file. Monitor how accurately the chatbot understands and responds to various user queries.

3.1.6 Test Edge Cases

Test the chatbot with untrained or ambiguous inputs to check how well it handles unknown or unexpected phrases. If necessary, refine the intents file and retrain the model for improved accuracy.

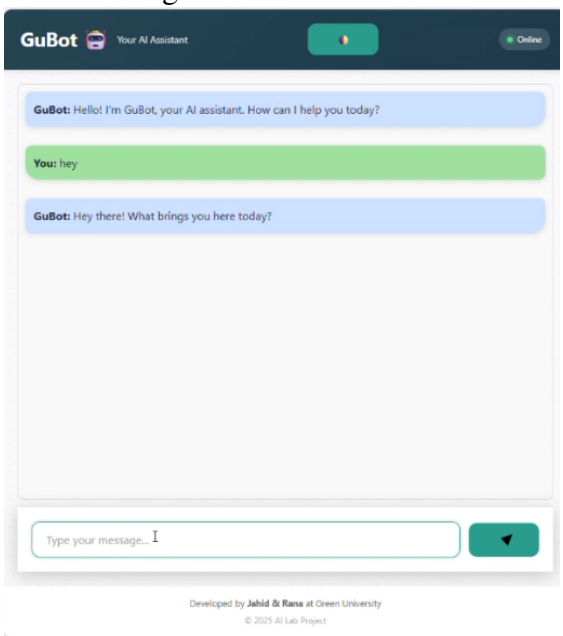
3.1.7 Evaluate Performance

Evaluate the chatbot's performance based on response relevance, speed, and consistency. Fine-tune the model or training data as needed to enhance its capabilities.

3.2 Results Analysis/Testing

The chatbot system has been created and put to the trial through the means of the multiple simulation sessions validated its capability to comprehend and respond to user inputs according to the intents embedded. The chatbot, throughout the testing stage, has been capable of detecting and reacting to the different groups mainly greetings, course inquiries,creators inquiries, and farewells in an accurate manner. Overall, the system achieved its goal of demonstrating how a basic chatbot can be built using natural language processing and deep learning. [4]

Figure 3.1: Test Result



3.3 Results Overall Discussion

The results of the chatbot project generally indicate that a practical and smart conversational agent can be generated through simple NLP and machine learning methods in Python. Thus, the chatbot can understand the intent of the user and supply suitable replies depending on the dataset in the intents.json file it has. Consequently, the process of tokenization, lemmatization, and the bag-of-words model was used, which in turn, the system simplifies the user's input and passes it into a deep learning model for accurate classification into one of the predefined categories.

3.4 Necessary Links

Here is the github links and project output video link

3.4.1 Github Link

GithubLink:https://github.com/JahidC0deSpace/ai$_{c}hatbot_{w}ith_{a}nn$

3.4.2 Drive Link

DriveLink:https://drive.google.com/file/d/1OFPaXU7MIC6OeRv$_{n}i0K6v2-xnLOK9FQ/view?usp=drive_{l}ink$

Chapter 4

Conclusion

4.1 Discussion

At its core, this AI conversational interface example project demonstrates a basic rule-oriented and intention-fueled conversational model using Natural Language Processing (NLP) and neural networks.Adequacy of the use case is evidenced by the fact that both (intents.json file) data and machine learning engineering (combined with) gave birth to user knowledgeable bot, which was able to understand the questions and respond accordingly. A key strength of the system lies in its simplicity and modular design. The use of NLTK for text preprocessing, along with a basic bag-of-words model and a neural network built using TensorFlow and Keras, allows for effective intent classification without requiring large-scale datasets. The chatbot performs well in recognizing trained inputs and returning appropriate predefined responses. [5]

4.2 Scope of Future Work

While the current chatbot project successfully demonstrates the basic principles of intent recognition and response generation, there is significant scope for future enhancement and development. One key area of improvement is the incorporation of dynamic response generation using advanced Natural Language Generation (NLG) techniques, allowing the chatbot to generate more natural and flexible replies rather than relying solely on hardcoded responses. Moreover, you can also enhance the model by employing another type of NLP model (e.g., Transformer-based architectures such as BERT or GPT), which would help the model to better understand the topic as well as generate better answers. The combination of speech recognition and voice synthesis would be more favorable in accomplishing its use over voice-based platforms such as smart assistants.

References

[1] Fankar Armash Aslam, Hawa Nabeel Mohammed, and Prashant S Lokhande. Efficient way of web development using python and flask. *International Journal of Advanced Research in Computer Science*, 6(2), 2015.

[2] Nitin Hardeniya, Jacob Perkins, Deepti Chopra, Nisheeth Joshi, and Iti Mathur. *Natural language processing: python and NLTK*. Packt Publishing Ltd, 2016.

[3] Mohammad Robihul Mufid, Arif Basofi, M Udin Harun Al Rasyid, Indhi Farhandika Rochimansyah, et al. Design an mvc model using python for flask framework development. In *2019 International Electronics Symposium (IES)*, pages 214–219. IEEE, 2019.

[4] Why Python. Python. *Python releases for windows*, 24, 2021.

[5] Sentdex. Neural networks from scratch - p.1 intro and neuron code. https://www.youtube.com/watch?v=1lwddP0KUEg, 2019. Accessed: 2025-03-10.