Introduction to AI COM727 Technical Report (AE2)

Your student number + Name:	Q16155807 – Om Prakash Narayan Gunalan
Your team member's name if applicable (for the implementation of the prototype)	Pavan Ram Perumal Alan Mathew Arukala Kalyan Undadi Swapna Kuncham

Li Bot

Table of Contents

Table of Figures

Introduction

Artificial Intelligence (AI) is an evolving technology that has the potential to revolutionize the way we interact with the world. AI enables machines to learn and take decisions based on data and patterns, making them more efficient and accurate than ever. AI has already proven to be a powerful tool in fields as diverse as healthcare, finance, tourism and transportation.

However, AI also poses several risks to our society. AI-based systems are vulnerable to bias and data manipulation, leading to inaccurate or unfair decisions. Additionally, AI has the potential to be used for surveillance and to invade our privacy. Furthermore, the development of AI technology raises ethical, legal and professional issues, as it is difficult to ensure that these systems are not abused or used for unethical purposes. In particular, the development of an AI-based library chatbot raises several ethical and legal issues.

For example, the chatbot must ensure that it is not unfairly biased against certain groups or individuals and must comply with relevant laws and regulations. Additionally, the chatbot must be designed in a way that respects the privacy of its users and does not infringe on copyright law. Finally, the library chatbot must be designed with the utmost professionalism

Need for your prototype

A chatbot can also be used to automate library processes, such as circulation and cataloguing, which can save time and resources. The chatbot can also be used to provide personalized recommendations, such as suggesting books or resources related to a student's interests or course of study. This will ensure the availability of resources and automation of common queries.

Statement of the problem

Lack of easy methods to derive information about the books 24/7, easy access of customer support, low efficiency in solving reiterating queries are some problems that are present.

Aims and Objectives

The aim of this chatbot is to provide a way to access basic information about books and opening times and answer reiterating queries.

Proposed Solution

To create a chat bot to answer basic queries about library so that it can be used in future to help people 24/7.

Prototype Design

We have found that most chat bots are traditionally entirely rule based we want to change it in a way so that they are partially rule based and rest is based on ML based algorithm this makes it more robust in nature. Then we are designing it in a structure so that it reusable for most use cases. We have created out data set based on our university.

Prototype Development and AI Algorithms used

We have started by just creating a chat and then we inserted our use case to it. We used Python and bootstrap based frameworks and packages to develop this project. **Flask** is used to convert the algorithms into an web app. **Keras** is used to build the neural network of the project and save it. **Numpy** is used to

convert the data into numpy array so that this can be processed by the neural network, **Nltk** is used to procces the datset into various formats so that they can be utilized the main use of the nltk library is to use the lemmatize function which converts the paragraph based answers into tokens and categorizes it, **JSON** is used to read the data set as is in json format ,**Pickle** is used to save the dataset post the processing. This is further discussed in the upcoming paragraphs.

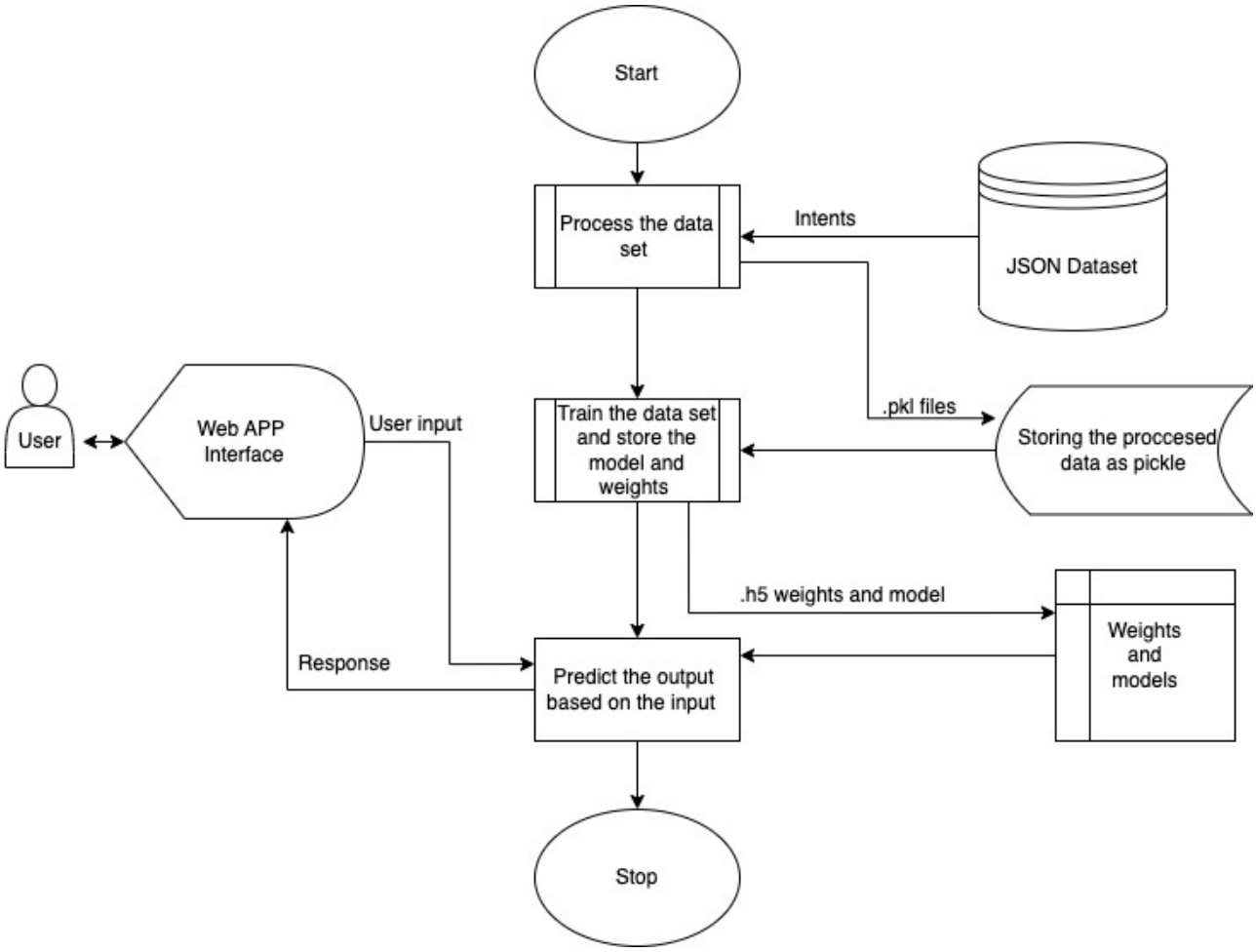


Figure 1 Flow chat of working diagram

We have used Stochastic gradient descent as this is one of the most commonly used type of ML model, We have set the parameters in this function in way so that it provides the results optimally this is also nesterov accelerated.

```
# Create model - 3 layers. First layer 128 neurons, second layer 64 neurons and 3rd output layer
# equal to number of intents to predict output intent with softmax
model = Sequential()
model.add(Dense(128, input_shape=(len(train_x[0]),), activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(len(train_y[0]), activation='softmax'))

# Compile model. Stochastic gradient descent with Nesterov accelerated gradient gives good result
sgd = SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])

#fitting and saving the model
hist = model.fit(np.array(train_x), np.array(train_y), epochs=200, batch_size=5, verbose=1)
model.save('chatbot_model.h5', hist)
```

Figure 2 Neural network

There are 3 layers in this model with drop out layers in between all layers. First layer consists of **128 neurons** with input shape and an activation funtions that is **relu.** Second layer consists of **64 neurons** and an activation layer which is **relu.** Third layer consists of output and activation function which is **softmax**. After each layer there is a dropout layer with a rate of 0.5. The purpose of the dropout layer is to reduce the risk of overfitting by randomly disabling a certain number of neurons so that the model is less dependent on any single neuron and instead leverages the collective input from all of the neurons.

The data processing is a part where we are using nltk library in which lies the lemmatizer function that is used to split it into tokens, then these are stored as pickle files and it is further appended into training data. The lemmatizer function is used to reduce a word to its base form. It is used to reduce the complexity of the text and make it easier to work with. It is also used to reduce the text to its root form, making it easier to compare words, which is useful for language processing tasks.

```
for intent in intents['intents']:
    for pattern in intent['patterns']:

        w = nltk.word_tokenize(pattern)
        words.extend(w)

        documents.append((w, intent['tag']))


        if intent['tag'] not in classes:
            classes.append(intent['tag'])

words = [lemmatizer.lemmatize(w.lower()) for w in words if w not in ignore_words]
words = sorted(list(set(words)))

classes = sorted(list(set(classes)))

print (len(documents), "documents")

print (len(classes), "classes", classes)

print (len(words), "unique lemmatized words", words)

pickle.dump(words,open('words.pkl','wb'))
pickle.dump(classes,open('classes.pkl','wb'))
```

Figure 3 Data proccessing

Next, we have used Flask to create a web app with the help of bootstrap. The UI part of the program is handled using bootstrap template and the backend is handled using Flask, there we have a module to get response from the user and then process the input with the use of process.py. The web app includes a form with a text box and a submit button, when the user enters the text and clicks the submit button, the text is sent to the Flask web app which processes the input and sends the response back to the web app.

```python
from flask import Flask, render_template, jsonify, request
import processor

app = Flask(__name__)
app.config['SECRET_KEY'] = 'enter-a-very-secretive-key-3479373'

Om Prakash Narayan
@app.route('/', methods=["GET", "POST"])
def index():
    return render_template('index.html', **locals())
Om Prakash Narayan
@app.route('/chatbot', methods=["GET", "POST"])
def chatbotResponse():

    if request.method == 'POST':
        the_question = request.form['question']

        response = processor.chatbot_response(the_question)

    return jsonify({"response": response })
if __name__ == '__main__':
    app.run(host='0.0.0.0', port='8888', debug=False)
```

Figure 4 Flask app

The above image explains the functionalities of flask app and its implementation. The chat board response definition helps us. Following this its processing is done in the procces.py file which is basically, the input Is categorized based on the trained model which is comparing the similarity of the sentence with others, and then the data that is present as an response on the classified category is returned in random if there are multiple responses present.

```
def predict_class(sentence, model):
    # filter out predictions below a threshold
    p = bow(sentence, words, show_details=False)
    res = model.predict(np.array([p]))[0]
    ERROR_THRESHOLD = 0.25
    results = [[i,r] for i,r in enumerate(res) if r>ERROR_THRESHOLD]
    # sort by strength of probability
    results.sort(key=lambda x: x[1], reverse=True)
    return_list = []
    for r in results:
        return_list.append({"intent": classes[r[0]], "probability": str(r[1])})
    return return_list

 Om Prakash Narayan
def getResponse(ints, intents_json):
    tag = ints[0]['intent']
    list_of_intents = intents_json['intents']
    for i in list_of_intents:
        if(i['tag']== tag):
            result = random.choice(i['responses'])
            break
        else:
            result = "You must ask the right questions"
    return result

 Om Prakash Narayan
def chatbot_response(msg):
    ints = predict_class(msg, model)
    res = getResponse(ints, intents)
    return res
```

Figure 5 Proccess.py code

Coming to the execution we can see that we have several files in which chatbot.py and app.py are how we execute the programs. Before starting to run the code, we need to setup the environment using the ReadMe.md file

which is present in the root directory of the project. Initially we need to install all the packages mentioned in the file. At the time of execution, we have python version 3.10.8 installed in our pc, when setting up the environment we need to have the same version of python to ensure there are no version control based issues. Chatbot.py is the file where we can train the chat bot using the data set by just running it. Which should be done currently in every system when trying to setup the new environment for this project.

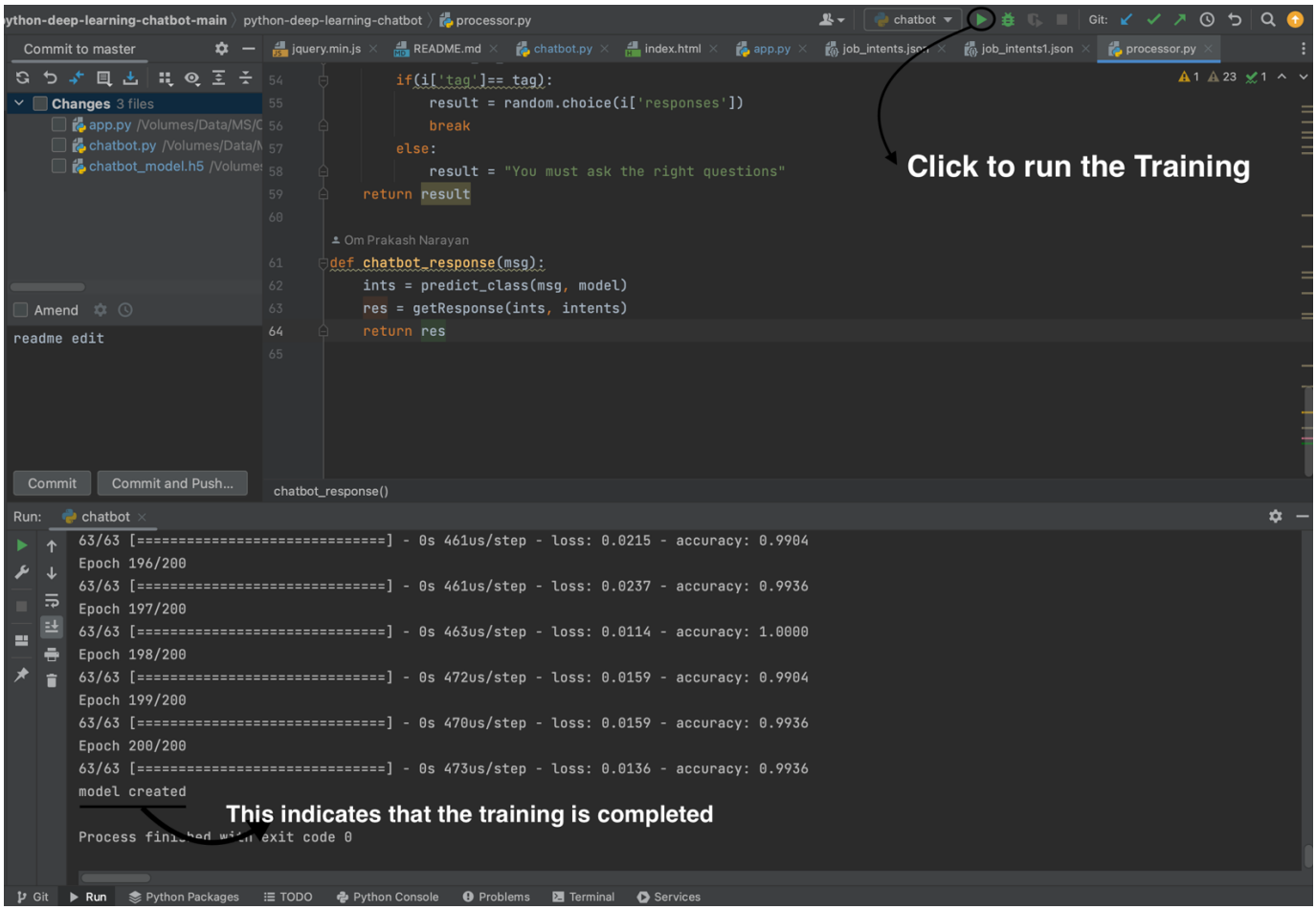
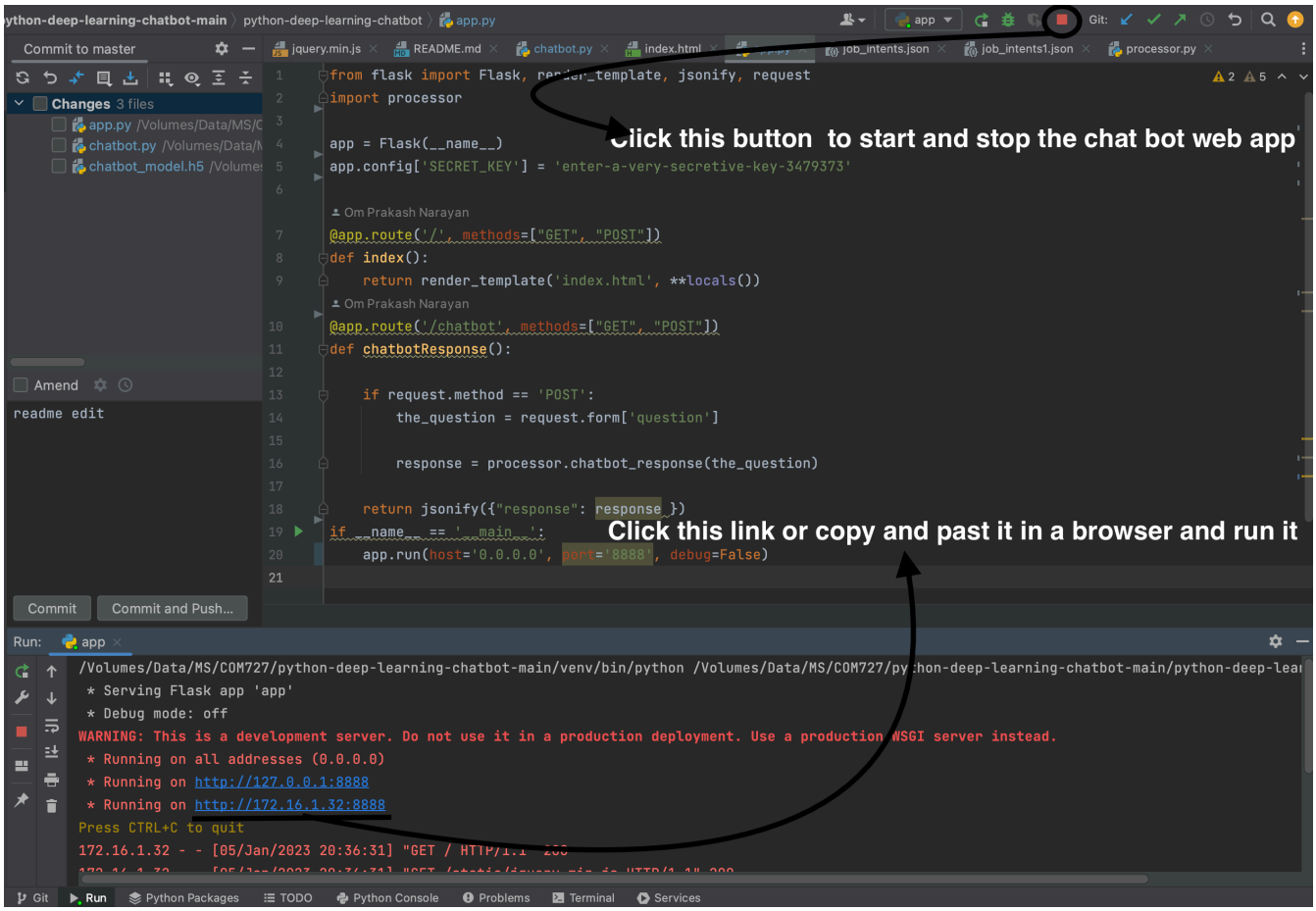


Figure 6 Training the chat bot

To run the chatbot post training we need to run app.py which in return opens the chat bot in the browser if the is not opening we need click on the link that is present in the terminal.



We need to copy and paste the link in the browser that helps us to see the chat bot webpage.

Then we need to input our query in the chat box and click send which gives us our response. Which is explained using the above image.

Evaluation

We have done **manual testing** on this prototype which is basically testing using the questions that are frequently asked in library this gave us a result which helped us to ensure that the prototype satisfies the problem statement.

Limitation

We have faced many issues in terms of the data set as we could not find any data set satisfies our specific use case to, we ended up creating our own data set, in terms of complex scenarios chatbot seems to lack.

Conclusion

In terms of the prototype, we have we can say that this satisfies the basic use case its created for. We have got an accuracy of 98.7 %, In terms of future improvement, we need to add more dataset to overcome its limitations. We can also deploy this protype in a cloud-based setup which will ensure that this is accessible for the exact use case its created for.

Reference List (Harvard Style)

Artificial Neural Network Based University Chatbot System | IEEE ... (no date). Available at: https://ieeexplore.ieee.org/document/8973095/ (Accessed: January 8, 2023).

Deep learning techniques for implementation of Chatbots (no date). Available at: https://www.researchgate.net/publication/341809413_Deep_Learning_Techniques_for_Implementation_of_Chatbots (Accessed: January 8, 2023).

Dialog prediction in institute admission : A deep learning way (no date). Available at: https://www.researchgate.net/profile/Pooja-Raundale/publication/352671710_Dialog_prediction_in_Institute_Admission_A_Deep_Learning_Way/links/6298523a416ec50bdb0366fa/Dialog-prediction-in-Institute-Admission-A-Deep-Learning-Way.pdf (Accessed: January 8, 2023).

EDUBOT-a chatbot for education in covid-19 pandemic and VQAbot ... (no date). Available at: https://www.researchgate.net/publication/354797635_EDUBOT-A_Chatbot_For_Education_in_Covid-19_Pandemic_and_VQAbot_Comparison (Accessed: January 8, 2023).

(1970) *Programming challenges of chatbot: Current and future prospective: Semantic scholar*, *2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*. Available at: https://www.semanticscholar.org/paper/Programming-challenges-of-chatbot%3A-Current-and-Rahman-Mamun/c987d7617ade93e5c9ce8cec3832d38dd9de5de9 (Accessed: January 8, 2023).