CHATBOT USING PYTHON

PHASE 4: DEVELOPMENT PART 2

INTRODUCTION

In this phase, we will build upon the work done in previous phases of our chatbot development project. Phase 4 focuses on selecting a machine learning algorithm, training the model, and evaluating its performance. The key objectives include:

1. **Selecting a Machine Learning Algorithm:** We will choose a machine learning algorithm suitable for our chatbot project's requirements. In this document, we'll demonstrate an example using the Multinomial Naive Bayes algorithm for "Intent Recognition."
2. **Training the Model:** We will train the selected machine learning model. In the context of chatbots, "Intent Recognition" is essential for understanding user queries and generating appropriate responses.
3. **Performance Evaluation:** We will evaluate the model's performance to ensure it effectively recognizes user intents.

Now, let's proceed with the code and explanations for each of these steps.

1. Selecting a Machine Learning Algorithm

For our chatbot project, we aim to recognize user intents. "Intent Recognition" is a critical component of chatbot systems as it allows the chatbot to understand the user's purpose and respond accordingly. To accomplish this, we select the Multinomial Naive Bayes algorithm, which is commonly used for text classification tasks like intent recognition.

2. Training the Model

To train our intent recognition model, we need labeled data. We will use a dataset consisting of user questions (input) and associated intent labels (output). Each user question will be associated with one or more possible intents. We'll use a Term Frequency-Inverse Document Frequency (TF-IDF) vectorization method to convert text data into numerical format. The Multinomial Naive Bayes algorithm will be trained on this TF-IDF representation of the user questions and their corresponding intents.

3. Performance Evaluation

Evaluation is crucial to ensure that our intent recognition model is performing effectively. We will split the dataset into training and testing sets, train the model on the training data, and then use the testing data to evaluate its performance. The key evaluation metrics include accuracy and a classification report. Accuracy measures the percentage of correctly recognized intents, while the classification report provides more detailed insights into model performance, including precision, recall, and F1-score for each intent class.

Code Implementation

Here's the Python code for Phase 4, including the selection of the Multinomial Naive Bayes algorithm, training the model, and evaluating its performance:

```python
# Import necessary libraries
import os
import pandas as pd
from transformers import pipeline
from flask import Flask, request, jsonify
from pyngrok import ngrok
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, classification_report

# Set your OpenAI GPT-3 API key as an environment variable
os.environ['API_KEY'] = 'API_KEY'

# Load and preprocess the provided dataset for intent recognition
file_path = '/content/drive/My Drive/colab/dialogs.txt'
data = pd.read_csv(file_path, sep='\t', names=['Question', 'Answer'])
X = data['Question']
y = data['Answer']

# Initialize intent recognition model
tfidf_vectorizer = TfidfVectorizer()
X_tfidf = tfidf_vectorizer.fit_transform(X)
intent_model = MultinomialNB()
intent_model.fit(X_tfidf, y)

# Initialize GPT-3 pipeline for response generation
generator = pipeline('text-generation', model='EleutherAI/gpt-neo-2.7B')

# Create a Flask web app
app = Flask(__name__)

@app.route('/chat', methods=['POST'])
def chat():
    if 'user_message' in request.form:
        user_message = request.form['user_message']

        # Recognize the intent of the user message
        user_message_tfidf = tfidf_vectorizer.transform([user_message])
        recognized_intent = intent_model.predict(user_message_tfidf)[0]

        # Generate a response based on the recognized intent using GPT-3
        bot_response = generator(user_message, max_length=50, do_sample=True)[0]['generated_text']

        return jsonify({'intent': recognized_intent, 'bot_response': bot_response})
    else:
        return jsonify({'error': 'Invalid request'})

if __name__ == '__main':
    # Set up Ngrok to create a public URL for the Flask app
    public_url = ngrok.connect(addr='5000')
    print(' * ngrok tunnel "' + public_url.public_url + '" -> "http://127.0.0.1:5000"')

    # Start the Flask app
    app.run(host='0.0.0.0', port=5000)

# Evaluate the intent recognition model's performance
# Split the data into train and test sets, train the model, and then evaluate
X_train, X_test, y_train, y_test = train_test_split(X_tfidf, y, test_size=0.2, random_state=42)
intent_model = MultinomialNB()
intent_model.fit(X_train, y_train)
y_pred = intent_model.predict(X_test)

# Calculate and print accuracy and classification report
accuracy = accuracy_score(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)

print(f'Intent Recognition Model Accuracy: {accuracy}')
print('Intent Recognition Model Classification Report:')
print(classification_rep)
```

CONCLUSION

Phase 4 focused on selecting and training a Multinomial Naïve Bayes model for intent recognition. We have successfully completed the core development phases, and the chatbot is ready to interact with users and provide responses based on recognized intents. Our next steps involve deploying the chatbot, conducting user testing, and exploring opportunities for further enhancement.

BY:

SHWETHA V

ROOBINI M

GOVINDALALITHA T M

HARINI PRIYAA S