Intelligent Assistant With Face Recognition

Overview

Intelligent assistants identify individuals by their features, such as their name. If a new person arrives, they must be added to the database. Subsequently, voice assistance is provided, the system hears the user's query and responds according to the model.

Communication Model

Text to speech and speech to text

Speech interaction, enabling users to both input and receive information via speech.

Users can provide input through speech, which the script captures using the microphone.

The function intelligently adapts its recognition method based on internet connectivity.

When internet access is available, it utilizes Google's speech recognition service for accurate transcription. In the absence of internet connectivity, it seamlessly switches to a local speech recognition model. There is function to verify internet connectivity by sending a request to Google's homepage. This ensures reliable performance by adjusting the recognition mode according to the availability of an internet connection.

NLP Model

This model implements a basic neural network model for intent classification based on user input. Here's a summary of its functionality:

Data Load:

First loads data from a JSON file containing intents and associated patterns.

Intents: Each intent represents a category of user queries or statements.

It contains:

Tag: Unique identifier for the intent.

Patterns: Possible variations of user input that correspond to this intent.

Responses: Possible responses that the chatbot can provide when recognizing this intent.

It tokenizes the patterns into words and lemmatizes them to convert them to their base form.

Unique words and intents (classes) are extracted and stored for further processing.

Training Data Generation:

The script creates training data by constructing a bag of words representation for each pattern.

Each pattern is represented as a binary vector indicating the presence of words from the vocabulary.

Output labels are one-hot encoded, with a '1' corresponding to the class (intent) and '0' for others.

Model Definition:

A neural network model is defined using Keras, consisting of three layers:

two dense layers with ReLU activation and dropout, and an output layer with softmax activation.

The model is compiled using stochastic gradient descent (SGD) optimizer with categorical cross-entropy loss.

Model Training:

The model is trained on the generated training data using the fit method.

Training parameters include the number of epochs, batch size, and verbosity level.

The trained model is saved to a file (model.keras) for future use.

Model Prediction:

Loading Pre-trained Model:

The pre-trained neural network model is loaded from the file named 'model.keras'. Additionally, the data containing intents and associated responses are loaded from the 'data.json' file.

Tokenization and Lemmatization:

The cleanup sentence function tokenizes and lemmatizes the input sentence. Lemmatization is the process of reducing words to their base or root form.

Bag of Words (BoW) Representation:

The bow function converts the tokenized sentence into a bag-of-words representation using the previously loaded words.

Prediction of Intent:

The predict_class function predicts the intent of the user input using the trained model. It calculates the probability of each intent class and returns the intents with probabilities above a certain threshold (ERROR_THRESHOLD).

Selection of Response:

The getResponse function selects a random response from the set of responses associated with the predicted intent.

Chatbot Response:

The chatbot response function takes a user message as input, predicts the intent, and returns a suitable response.

Face Detection And Face Recognition Model

The Python script implements a real-time face recognition system using the OpenCV and face recognition libraries. Here's a brief description:

Functionality:

The script continuously captures video frames from the webcam, detects faces in each frame using a pre-trained Haar cascade classifier, and Recognize the detected faces with pre-registered faces using facial recognition algorithms.

Components:

setAndGetName Class: Manages the name associated with the recognized face.

EncodeFaceData Function: Encodes face data for known faces, which is used as the training dataset.

FaceRecognition Function: Performs real-time face recognition by comparing detected faces with the encoded faces in the training dataset.

getName Function: Retrieves the name associated with the recognized face.

Final Output: The recognized name is displayed on the video frame in real-time.