PLAGIARISM SCAN REPORT

0%
Plagiarised

100%
Unique

Date	2021-12-12
Words	585
Characters	3682

Content Checked For Plagiarism

The work has been divided into various components.
Intent Defining
Data Preparation
Model Training
Model Testing with real-time query
Creating the front-end and integrating it with the backend

3.1 Intent Defining
"Intent" refers to the intention of a user communicating with a chatbot, or the intention behind each message received by the chatbot from a specific user. Depending on the domain of the chatbot solution, these intentions may differ from one to the next. As a result, it's critical to give the chatbot with appropriate intentions that are relevant to its domain. The chatbot uses a domain knowledge base to answer questions and performs a range of additional tasks. In order to continue interacting with the user, the chatbot must first understand what the user is saying or planning to do. As a result, the chatbot must be able to decipher the intentions underlying user communications (in order to determine the user's intent). A few simple intents and messages that correlate to those intentions have been produced, and replies have been mapped to each intent type. Those are stored in several json files.

3.2 Data Preparation
Because this is a simple chatbot, there is no need to gather or download any significant datasets. To train the model, a unique dataset is developed. Before this dataset can be trained, the intentions must be understood.The needed data is taken from the json file that has been loaded in the previous step. The data is then split into two parts: one contains all of the training data (which are example messages for each intent category), and the other contains all of the target labels that match to each training data.
LabelEncoder is used to transform the target labels into a format that the model can comprehend.The user-built Tokenizer class vectorizes the text data corpus. By default, this class eliminates all punctuation, converting the texts into space-separated word sequences, which are then divided into token lists. They'll be indexed or vectorized after that. Another option is introduced to handle words that are out of vocabulary words (tokens) at the moment of inference.Another user-created method is included to ensure that all of the training text sequences have the same size.

3.3 Model Training
The suggested model's neural network architecture is then specified, with Keras' "Sequential" model class being employed. Softmax was employed as the activation function.

The model has trained via extensive training. It is preferable to dump all needed files after training so that they may be used during inference. So that the trained model, the fitted tokenizer and the encoder object are all saved.

3.4 Model Testing with real-time query
When a new user message is received, the chatbot computes the similarity of the new text sequence to the training data. Taking into account the confidence scores obtained for each category, it assigns the user message to the intent with the greatest confidence level.

3.5. Creating the backend and front-end and integrating it with the backend
Now that the Deep Learning component has been done, the front-end is built with HTML, CSS, and Bootstrap, while the backend is built with Flask. As a result, the end- to-end project is now completed and is given the shape of a web application.

4. Conclusion
This is a chore that will be mechanised for 24 hours a day, eliminating people's loneliness and providing them with a talking agent to converse with. The performance may be enhanced further by employing emoji-based responses, additional deep learning models, or a different pre-built chatbot framework.

Matched Source

No plagiarism found