A group of people sitting around a table

Description automatically generated with medium confidence

.Shape

Description automatically generated with medium confidence

Student Name: Mohammed Albardawil

Student ID Number: P2661204

Natural Language Processing Coursework

Executive Summary

2021/2022

This report summarizes the implemented solutions for four chatbots. The first is one which is already provided by PyTorch is the RNN based chatbot. Other implementations include Non-AI Technique, MLP, and the so-called LSTM. MLP and LSTM are simply RNN that has notable differences in their architecture, which are summarized below. All the three models RNN, MLP and LSTM are Deep Learning based AI approaches. The Non-AI algorithm used is called Rule-Based approach.

Regarding the Rule-Based chatbot, its structure is so simple compared to the other chatbots built using AI techniques. It consists of two main parts: the predefined rules as grammatically declared as dictionary of strings of messages and responses. The second part is the match\_rules() function that maps the message with its associated response. However, to ensure more accuracy, the rules dictionary must be extended to have a lot of rules as the more questions the user ask the more accurate answer may be received from the chatbot. These rules use the Regex library of Python, which is simple allows for creating regular expressions.

To bring more accurate and high-performance solutions to process texts for chatbot-related applications, this should be overcome by the state-of-the-art Deep Learning AI algorithms. As stated earlier, the first chatbot in this project is already developed using Recurrent Neural Networks (RNNs). RNNs are the most basic Deep Learning Machine Learning models that other models are built on it with some modifications to adapt specific applications. RNNs are commonly used for applications that have time series or sequential data like language translation, speech recognition, sentiment analysis, and much more useful applications that fall into the field of Natural Language Processing (NLP). RNNs process is having output that is dependent on its input sequential data with hidden state. The basic RNN mode may have one to two fully connected layers. However, the implemented RNN in this project does have only one fully connected layer.

The second implemented model is called Multilayer Perceptron (MLP). MLP is simply RNN with at least three fully connected layers and can also be called Multilayered RNNs. Each of MLP layers contain weighted sum of a set of nonlinear systems resulted from previous connected input. Beside that, MLP use single length inputs unlike RNNs which use variable length inputs.

The third Deep Learning model implemented is Long-Short Term Memory (LSTM), which is another form of RNN models. The only difference between LSTM and RNN is that RNN and all other RNN based models use Gated Recurrent Units (GRU). LSTM by itself has three gates, input, forget and output gates. GRU has only two gates in which the input and forget gates are considered as one gate.

All of the three model-based chatbots have been evaluated using two algorithms: the Cosine Similarity and Bilingual Evaluation Understudy Score (BLEU). Both algorithms give results based on the similarity between the reference and test sentences.

There has been 6 training executed for all the three models based on fine-tuning by changing hyperparameters. The main hyperparameters were tuned are number of training iterations, number of encoding and decoding layers, and type of Luong attention layer (dot product and general). In addition, graphs have been plotted for showing all losses vs training iterations for all models plus summary table for training with their modified parameters and another summary table for all conversations made with the chatbots along with similarity results evaluated by both BLEU score and Cosine Similarity algorithms. All models tested with the [Cornell Movie-Dialogs Corpus](https://www.cs.cornell.edu/~cristian/Cornell_Movie-Dialogs_Corpus.html) dataset, and the whole system was extensively implemented using PyTorch with help of NLTK for corpus management.