**NLP Final Project**

Kevin Lin

Email :ltsungha@syr.edu

All of the writing in this document is my original work, except for the code excerpts and any quoted material for which I have provided a citation

Introduction

In this project I forked the Github chatbot code from [KMahima03](https://github.com/kkkkkktw/NLPfinal/commits?author=KMahima03). This python notebook creates feature of human like chatting robot that can make logical response to user input. For example, when user say hello, the chatbot will greet them back and ask how can it help. In the following file, we import a lot of python library for NLP. Since we have labeled responses data. Using NLTK to tokenize the data and applying Keras API port works on Tensorflow. After that we use about 4 kind of layer (Embedding, LSTM, Flatten, Dense) to create a LSTM model. Finally we design a human behavioral input to test our model.

Code Walkthrough

**Block 1**

From the block one we import some important python libraries, TensorFlow is the main frameworks we used in this project and also import numpy helping us do the calculation.

**Block 2**

It is the data we are going to use in this project. We stored it as a JSON file since it has best compatibility and very easy to read.

**Block 3 to block 5**

Convert our previous data into Pandas DataFrame so we can more properly use in our training process, like MS excel pandas DataFrame could handle 2-dimensional data (rows and columns). At the end of the block we can see that we have each sentences with tag.

**Block 6 to block 7**

To train our model we have to do text preprocessing. In this part we must have to convert unstructured data to structured data. As we did in our class. First, we need to remove punctuation and do tokenization. Second, we delete stop words and do stemming and lemmatization. However in this case the author just use keras.preprocessing.text library to do all the steps automatically. Moreover, to properly feed the model, padding is necessary. The author also applied keras library to do this automatically.

**Block 8 to Block 9**

Make definition of vocabulary and check the input shape, we could see it has 6 dimension and 56 unique words.

**Block 10**

We construct a LSTM model include 4 kind of layer each one is embedding, LSTM, Flatten and Dense layer. Because LSTM contain gate and cell state, it has better performance handling long term memory than traditional RNN. That is perfect match for chatbot training. We use vocabulary as input and set return sequences as true. Therefore, we could return all hidden state of last LSTM layer. For the Dense layer, the author uses SoftMax activation function classifier.

**Block 11 to Block 12**

Compiling and train the model that we built with 200 epochs. The author apply sparse\_categorical\_crossentropy as loss function. This function is fit in label input data that is what we have.

**Block 13**

The figure shows that about after 130 epochs, our training accuracy become stable and loss.

**Block 14**

The author did the demonstration of chatting with our trained bot. As a result, our bot can distinguish greeting, question, farewell pretty well, responding logical answer to user.

Conclusion

Chatting bot is very commonly use in our daily life. For example, when we want to search for customer support in Amazon store, the first answering your questions is always a bot. That may reduce cost of customer supports rather than hiring bunch of people standing by online. In this project we built a sequence2sequence LSTM model and train it. As a result, we can see if we ask something that is exist in our json data set. Our bet can respond quite logically and almost perfect. However if we say something that is not included in our data set like “Say something funny”, the bot will automatically say goodbye and end the conversation. In my opinion this bot just has really simple function and not really useful enough, but we can expand our data set feeding into the model, letting our model handle much more chatting scenario. Even more we can change the training method. For instance, OpenAI which make an overnight sensation after its release a few weeks ago, fine tune with GPT3.5 with supervised learning. It has really powerful function not only can handle casual chat but also can do translation, teaching… . We may apply similarly training process to make our model have better performance. First, try using GPT or BERT rather than only LSTM. Second, we can label the answer that GPT generated weather it is human prefer or not. Even more, we can use reinforcement learning techniques adjust parameters to get best rewards value. Although it is really a tremendous amount of work , it’s still a feasible way.