Clemente, Christian Justine G.

Yu, Julian

4-CSA

**Language Modeling Implementation**

**1.) Steps on how to prepare text on developing a word-based model.**

The text was *The Republic by Plato,* the text was prepared by downloading the text from the internet and loading it in a python file.

However, before creating a model, one must take first into account what problem will the model be solving. In this case, this is considered a *generation* task because the model must generate a text, based from the data which this is trained for rather than a *discriminative* task, which falls under classification, etc.

**2.) How was the text cleaned?**

The text was cleaned by:

* Replacing - - with a space.
* Tokenizing words by using a whitespace delimiter.
* Removing punctation(s) from each token.
* Removing tokens that are not alphabetic.
* Turning all the tokens into lower case.

**3.) How was the data trained?**

The data was trained by using a Recurrent Neural Network(RNN) consisting of Bi-Directional Long-Short Term Memory Cells.

**4.) Explain the LSTM modules.**

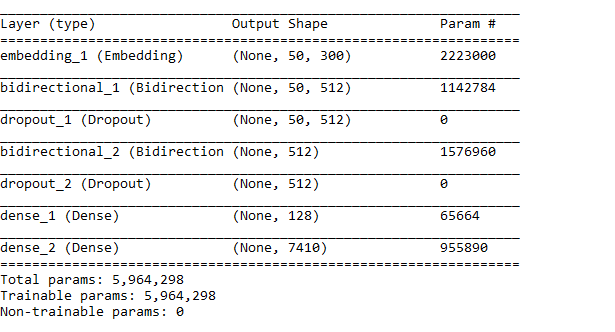
The LSTM cells are the solution to the recurrent neural network’s vanishing gradient problem. Simply put, the LSTM cells solve like mini-neural networks that keep track of what to remember and what to forget. The data that needs to be remembered are passed through hidden states.

LSTM Cells are based on *Markov Assumptions*, so it can be considered that a layer of RNN with LSTM cells are behaving like *Hidden Markov Models*.

**5.) Tuning Model**

Tuning the model is the most challenging part, specifically in a generative task. The data doesn’t have a testing set in which we can use a metric to define the performance of a model. Since this is the part where a metric depends on what the user does, then we’ll settle for the metric of accuracy. Our predefined model consists of 2 LSTM layers, both of them Bi-directional, the only difference between the 2 LSTM layers is the 1st LSTM layer has the parameter return\_sequences=true, this parameter means that the LSTM layer *returns the output of each hidden state.* The 2nd layer, the return\_sequences=false because it will cause an error in the neural network when it is ticked to true because of the dimensionality of the data.

We tried increasing both cells of the LSTM layers, to 256 layers, and adding a Dropout cell after each LSTM layer with a 0.2 value. The simple hidden layer after the 2 LSTM layers are also increased to 128 number of neurons. We also increased the dimensions of our Word Embeddings, from 100 to 300 dimensions. Our model looks like this:



With only 50 epochs, we have achieved the results of 46.23% with our fine-tuned model. A 6.92% slight difference compared to our predefined model which is trained on 100 epochs. This shows that by increasing the number of LSTM cells and dimensionality of word embeddings, we can increase the performance of our model to generate text. However, increasing the number of cells may cause overfitting the model, that’s why we added a dropout function of 0.2 to each of the LSTM layers.

