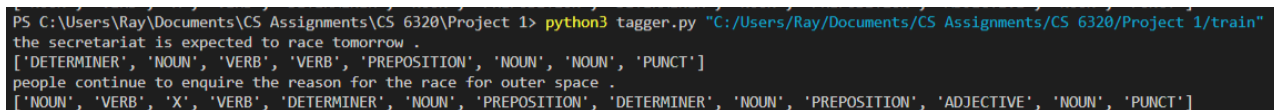


## Project 1 Short Report

For the Part-of-Speech tagging (POST) Project, we were given 500 documents to use for training a Hidden Markov Model (HMM) with the Viterbi algorithm and comparing it with the Recurrent Neural Network (RNN) using the Keras library. For testing, we were asked to evaluate two sentences; “the secretariat is expected to race tomorrow .” and “people continue to enquire the reason for the race for outer space .”. The interesting thing to note is that in the second sentence, “enquire” (british spelling of ‘inquire’) does not exist in any of the training documents, so the goal was to see how to handle such a situation with both models.

To begin with, the Hidden Markov Model was interesting because it had to be implemented without using a data science library. Here are the results that I got for the model on each of the two sentences:



```
PS C:\Users\Ray\Documents\CS Assignments\CS 6320\Project 1> python3 tagger.py "C:/Users/Ray/Documents/CS Assignments/CS 6320/Project 1/train"
the secretariat is expected to race tomorrow .
['DETERMINER', 'NOUN', 'VERB', 'VERB', 'PREPOSITION', 'NOUN', 'NOUN', 'PUNCT']
people continue to enquire the reason for the race for outer space .
['NOUN', 'VERB', 'X', 'VERB', 'DETERMINER', 'NOUN', 'PREPOSITION', 'DETERMINER', 'NOUN', 'PREPOSITION', 'ADJECTIVE', 'NOUN', 'PUNCT']
```

*Figure 1: HMM Model Output*

Using the viterbi algorithm yielded very consistent results, getting most of the words correct, as indicated with the the first test sentence. What was particularly interesting to me though was that “race tomorrow” was interpreted as “NOUN NOUN”, even though “race” was a verb in this context. This is probably because there were more instances of race as a noun in the training documents than there were as a verb. Thus, the HMM model can be considered “rigid” because it does not know how to predict uses of words it does not enough experience of. This was also the case when it came to the prediction of the foreign word, “enquire”. My program assumes that the the probability of an unknown word having a particular tag is equal for all of the tags. Thus, the model struggles to identify the word, and instead defaults to the highest probability for the rest of the known words in the sentence.

The Recurrent Neural Network (RNN) on the other hand, was a lot more about fitting a POST problem in such a way that the Keras library could understand as a model, in order to both train and test it compared to foreign input. Here are the results that I got when I trained the model for 40 epochs:

Figure 2: RNN Model Output

Please note that the [PAD] tokens is the model predicting that the sentence has ended, and is not the length of the largest sized sentence found within the training documents, in this case 180 word tokens. During the training, the model produced really high accuracies, somewhere in the range of 99.5% to 99.6%, which would imply that the model is extremely accurate. However, because of the [PAD] tokens being easy to predict, and not every sentence being equal to the largest size, some significantly smaller, most of that accuracy involves predicting the end of the sentence, which is much easier than the tags of the actual sentence tokens in the model. This is also why when you run the program multiple times, you get slightly different results for the output, since the model has to be trained again.

forward propagation and backwards propagation that occurs in the RNN, constantly updating the value of the weights for each sentence to be able to prioritize for the next sentence.

Overall, this project was a lot of trial, error, and debugging, especially getting the input to fit into the Keras model, but it was really interesting to see how each model worked and how to handle unknown situations. It would be interesting to see how different companies, such as Amazon, Microsoft, and Google implements Part-of-Speech tagging for their search engines and how they increase accuracy without sacrificing precision for their models.