**Approach**

I have done following steps in order to detect and correct incorrect usage of articles. Accuracy of my model is 35.94% on sentence\_test.txt file.

* Removed special characters and non-English alphabetic words from sentence\_train.txt file
* Compared correction\_train file with sentence\_train file to find out incorrect articles in sentence\_train file and correct them before building the model
* Read each sentence in sentence\_train file and:
  + Found out articles used in that sentence
  + Assigned article used count for each word in that sentence
* Created a dictionary called “words” where key is each word in sentence\_train and value is a Python list of size 3 where 0th index corresponds to number of sentences in which that word has occurred with ‘a’, 1st index corresponds to number of sentences in which that word has occurred with ‘an’ and 2nd index corresponds to number of sentences in which that word has occurred with ‘the’.
* Read sentence\_test file and:
  + If the word is missing in ‘words’ dictionary then skip it
  + If the word is present then get its corresponding value i.e. list of article counts and multiply these lists column-wise. Find out the max value from these 3 values for each word and assign that article as most probable article.
* Then check this article with corresponding index in correction\_test file and if there is a match then increment correct\_prediction count by 1
* Else get the article from original sentence in sentence\_test file and compare it with the article that we predicted
* Correct Prediction / Total Prediction will give us the accuracy.
* Then read sentence\_private\_file.txt and ran the model on it. I have generated the output of this model on “sentence\_private\_test.txt” in “my\_submission.txt” file in the format [[‘null’, ‘null’, ‘null’, [correct\_article, confidence\_score], ‘null’], [….], [….]].
* Also tried with getting the count of Part of Speech of all words in the sentence before the article and after the article. However this did not help much as same POS was returned as most occurring before each article and similarly same POS was returned as most occurring after each article.

**Output**

* Correct predictions: 4313
* Total Predictions: 12000
* Accuracy: 35.9416666667