Semantic Similarity Based Framework Using Recurrent Neural Networks for Plagiarism Detection

Arihant Jain

Department of Computer Science Engineering, Manipal University Jaipur, Rajasthan, India

---***---

Abstract - *Plagiarism is defined as "the act of using another person's words or ideas without giving credit to that person" by the Merriam-Webster dictionary. In the field of academics, it is one of the worst offenses. The act of plagiarizing someone's work can lead to not just discreditation but also legal action. In this day and age, the vast quantity of information requires a robust and efficient system to quickly determine whether a document has plagiarized content or not. We also need a system to compress and store documents in a format that will allow us to quickly compare new documents with old ones. Due to the complexity of natural language, it is not possible to make a reliable tool with a direct comparison method. We require a tool that utilizes semantic similarity to determine the possibility of plagiarism. This project uses Long Short-Term Memory (LSTM) units to convert GloVe word vectors into a score per sentence to facilitate both efficient storage and operations.*

Key Words: **Plagiarism, Long Short-Term Memory, GloVe, Natural Language, Machine Learning, Recurrent Neural Networks**

1. INTRODUCTION

Since the advent of the internet, we have access to more and more information every day. Not only does the amount of information at our fingertips increase every day, but the speed at which we can access it is also improving daily. Never have we had petabytes of data just a simple search away. Despite the untold benefits of this situation, we also face a growing problem of plagiarism. Some may call plagiarism the worst offense in an academic institution. It undermines the hard work of others and is both unethical and illegal.

As such, we must develop methods of allowing an unsupervised system to check whether a submitted document is plagiarized from a list of documents. Furthermore, if it is plagiarized, the system should also be able to determine which sections of the document are plagiarized. If a sentence's vocabulary has been changed yet its meaning remains the same, it's still considered to be plagiarized and thus, a simple one to one comparison of sentences will not be sufficient. Due to the large amounts of data that would be required to check the document against, it is physically impossible for a human operator to perform these checks. My proposed model shall combine the powers of both Recurrent Neural Networks as well as Deep Neural Networks in order to make accurate estimations of whether plagiarism exists in each pair of documents.

1.1 Problem Statement

This application presents a recurrent adaptation of the neural network for labeled data comprised of pairs of variable-length sequences. This model is applied to assess semantic similarity between sentences, where we exceed state of the art, outperforming carefully handcrafted features and proposed neural network systems of greater complexity. This model provides word embedding vectors supplemented with synonymic information to the network artifacts. This is a framework-oriented application for maintaining the plagiarism of any artifacts or documents.

1.2 Conceptual Overview

Unlike humans, computers are incapable of understanding anything other than binary. When we look at a pair of documents, we are able to understand their meaning and easily discern whether they contain plagiarism. Computers, however, are incapable of performing this task. In order for a computer to be able to understand a text document, it needs to be converted into a vector of numbers such that the computer can use those numbers to "understand" the text. This where the concept of word vectors comes up. Word vectors are a *(1, n)* dimensional array, where *n* is the number of extracted features, that a computer uses to identify and draw relationships between words. These word vectors are at the core of any processing a neural network performs on a sequence of words. The word vectors utilized in this project were provided courtesy of Stanford University.

Once we have these word vectors, we can utilize them to build an array of word vectors corresponding to every word in a sentence, and every sentence in a document. With this three-dimensional array, our model can now begin learning to discern the existence of plagiarism. This process however is lengthy and time consuming to execute for every document at runtime. Hence, we may convert documents into a vector of scores for each sentence by teaching the recurrent neural network to generate unique scores depending on the words in a sentence. This drastically reduces storage size and improves processing time significantly. Armed with these scores, a deep neural network is able to compare two such vectors and determine the presence of plagiarism in them.

2. LITERATURE REVIEW

A. A Multi-Level Plagiarism Detection System Based on Deep Learning Algorithms [1]

This method proposed by El Mostafa HAMBI and Faouzia Benabbou attempts to detect plagiarism at not only the word level, but also the paragraph level by applying certain pre-processing steps. This method however requires a large amount of pre-processing to apply the model and this process is executed at runtime. The Long Short-Term Memory (LSTM) representations are also combined into a single matrix before it is operated on to determine plagiarism.

B. Semantic Similarity Between Sentences [2]

This paper, written by Pantulkar Sravanthi and Dr. B. Srinivasu, provides an objective comparison between several methods to compare similarity between variable length text sequences. The metrics provided are useful in creating a basis for the methods proposed in this paper yet are insufficient for usage as is due to the difference in methodology. The methods described in this paper are also relying on the usage of WordNet to pre-process the input data which is not present here.

C. Semantic Plagiarism Detection System for English Texts [3]

The methods described by Anupama Nair, Asmita Nair, Gayatri Nair, Pratiksha Prabhu and Prof. Sagar Kulkarni in their paper utilizes several complex pre-processing and operating steps to produce a plagiarism report. Not only is the text preparation process lengthy, but the subsequent steps also use WordNet, Named Entity Recognition and Latent Semantic Analysis to obtain the plagiarism report.

3. PROPOSED WORK

The proposed method is based on a recurrent neural network (RNN) comprising of both LSTM and dense layers that will generate score vectors for each document that can be easily compared using a deep neural network (DNN). This method will only require a raw text form of a suspicious document, which will be pre-processed and then checked against pre-vectorized set of source documents. This model is intended to compare documents in a single storage location instead of the internet. The workflow of the proposed model is shown in Figure 1 and each stage of the process is elaborated upon below.

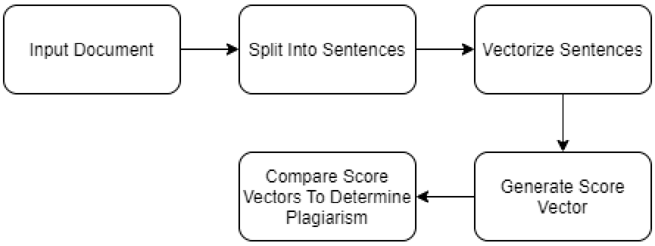

Figure 1: Proposed Model Workflow

3.1 Input Document

A text document that is suspected of plagiarism is inputted into the system. The model will compare this suspicious document with all source documents present in its database.

3.2 Split into Sentences

The raw text data is then split into sentences automatically using the punkt tokenizer. All punctuations in the sentences are replaced with whitespace and " 's " is separated into its own word.

3.3 Vectorize Sentences

Each sentence is further separated into words and then replaced with the corresponding word vector. The word vectors are then stacked and concatenated with zero vectors to achieve a common matrix shape of *(m, l, 300)* where m is number of examples and l is the length of the longest sentence.

3.4 Generate Score Vector

Each sentence vector is passed to the RNN, and element-wise squaring and summation is carried out on the output to generate the score. The scores of all sentences are then concatenated to form the score vector representative of the sentence.

3.5 Compare Score Vectors to Determine Plagiarism

The score vector of the suspicious document is then concatenated with that of the source document and passed through a DNN which determines whether there is plagiarized content in the suspicious document or not. This process is repeated with every source document and any documents that have been plagiarized from are displayed at the end.

4. IMPLEMENTATION

The first order of business was to implement the RNN for score generation. The STSBenchmark dataset was used to train this network. The original dataset contains pairs of variable length text sequences and a similarity score on a scale of 1.0-5.0. This score was rescaled to a 0.0-1.0 scale and then compared against the cosine similarity of the word vectors of the variable length sequences. The cosine similarity was not added as a part of the network itself, rather, it was added into the loss and accuracy functions to preserve the vector output from the RNN. The network architecture consists of an LSTM cell followed by a 20-unit layer of dense neurons with another layer of 10 dense units and an output neuron. The metrics and architecture are shown below.

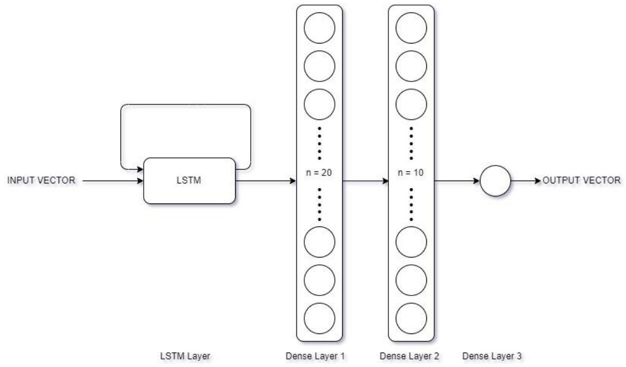

Figure 2: RNN Architecture

The output from this network is used to generate a single vector to represent the document inputted. These vectors can be stored and used for the plagiarism check whenever required and thus we can make the process of a plagiarism check significantly faster. The next step is to give a DNN 2 such score vectors so that it may decide the presence of plagiarism in them. The DNN is a simple 10-unit hidden layer with one output neuron and 20% chance of dropout. The dataset used here is PAN-PC-11 corpus. This dataset contains both source and suspicious documents with xml files to store metadata. This DNN takes as input 2 score vectors and predicts a binary label where 0 refers to no plagiarism and 1 refers to the presence of plagiarism. The architecture of the network is given below.

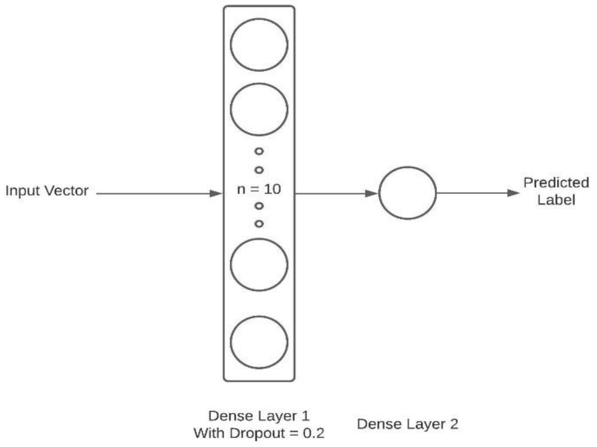

Figure 3: DNN Architecture

5. RESULTS

The training accuracy of the RNN was calculated to be 80.15%, with dev accuracy and test accuracy at 79.69% and 79.49% respectively. The training of the DNN yielded similar results with training accuracy as 79.79% and test accuracy as 80.64%. The results obtained from the training of the two neural networks are shown below.

Table 1: RNN Training Metrics

RNN Training Metrics	
Data Type	**Accuracy**
Training	80.15
Dev	79.69
Test	79.49

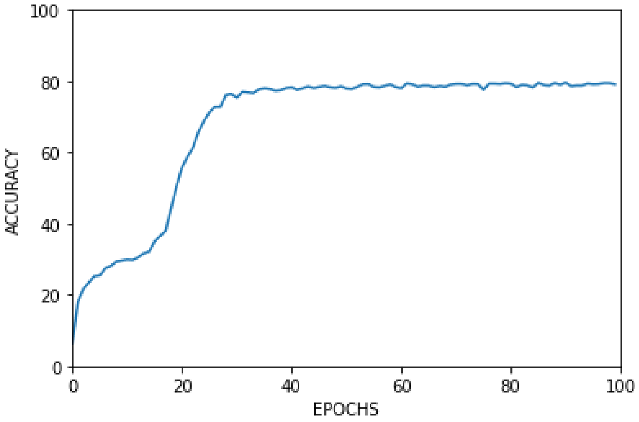

Figure 5: RNN Training Accuracy

Table 2: DNN Training Metrics

DNN Training Metrics	
Data Type	**Accuracy**
Training	79.79%
Test	80.64%

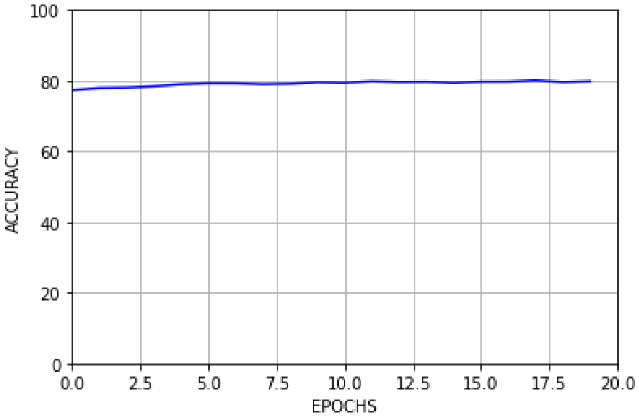

Figure 6: DNN Training Accuracy

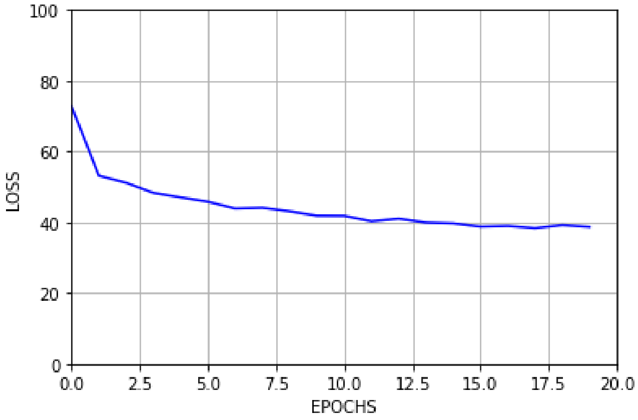

Figure 7: DNN Training Loss

6. CONCLUSION

This project shows the potential of using numeric vectors as a means of both representing and storing text documents for the purposes of plagiarism detection. While the RNN is working well, it may perform even better on a more accurate and robust dataset. The DNN may also be replaced by a Convolutional Neural Network (CNN) and instead of providing the CNN with score vectors we can generate a similarity matrix from the score vectors. This similarity matrix may be used to determine precisely where the plagiarism exists in the text. This idea however requires further research and exploration to be implemented.

7. REFERENCES

[1] El Mustafa HAMBI and Faouzia Benabbou, "A Multi-Level Plagiarism Detection System Based on Deep Learning Algorithms", IJCSNS International Journal of Computer Science and Network Security, VOL.19 No.10, October 2019, Pg. 110-117

[2] Pantulkar Sravanthi and Dr. B. Srinivasu, "Semantic Similarity between Sentences", International Research Journal of Engineering and Technology (IRJET), VOL. 4, ISSUE: 01, Jan 2017, Pg. 156-161

[3] Anupama Nair, Asmita Nair, Gayatri Nair, Pratiksha Prabhu and Prof. Sagar Kulkarni, "Semantic Plagiarism Detection System for English Texts", International Research Journal of Engineering and Technology (IRJET), VOL. 7, ISSUE: 05, May 2020, Pg. 532-537