IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Developing A Corpus Of Plagiarised Short Answers

[1] Sharmishtha Chavan, [2] Laxmi Bewoor
[1]Post Graduate Student, [2]Associate Professor
[1] Department of Computer Engineering, [2] Department of Computer Engineering,
[1]Vishwakarma Institute of Information Technology, Pune, India

Abstract: Plagiarism is widely acknowledged to be an enormous and increasing problem for education institutions (McCabe 2005; Judge 2008). A wide range of solutions, including several commercial systems, are proposed to assist the educator within the task of identifying plagiarised work, or even to detect them automatically. Direct comparison of those systems is formed difficult by the issues in obtaining genuine samples of plagiarised student work. We describe our initial experiences with constructing a corpus consisting of answers to short questions during which plagiarism has been simulated. This corpus is meant to represent sorts of plagiarism that aren't included in existing corpora and can be a useful addition to the set of resources available for the evaluation of plagiarism detection systems.

Index Terms **– Plagiarism, Plagiarism detection, AmazoneSagamaker.**

I. INTRODUCTION

Plagiarism is understood as illegal use of others' a part of work or whole work as one's own. Plagiarism diminishes one's innovative thinking, creativeness and improvement of data and also it's considered as illegal act during a moral society. In a survey that was conducted on plagiarism in the academia by the University of California in Berkley, it was shown that the percentage of plagiarism has increased by 74.4% within four years period (1993 – 1997) [1] and in another study by Butakov and Scherbinin concludes that quite 90.0% of high school students are involved in plagiarism. Plagiarism is one of the growing issues in academia. Academic staff faces difficulties in marking students' assignments with higher degree of judgment and waste their valuable time for plagiarism detection. The paper focuses on building an efficient, simple and fast tool for plagiarism detection on text based electronic assignments to attenuate this issue and to assist the tutorial staff in conducting proper evaluation of assignments.

II. PROBLEM DEFINITION

Plagiarism is one of the growing issues in academia and is always a concern in Universities and other academic institutions. The situation is becoming even worse with the supply of ample resources on the online. This paper focuses on creating an efficient and fast tool for plagiarism detection for text based electronic assignments.

III. IMPLEMENTATION

This is developed using the tri-gram sequence matching technique with the help of a scripting language .Figure 1 depicts the steps followed in the development and such steps are discussed in development and such steps are discussed in details under the subsections in this section.

The electronic assignments were pre-processed before they're sent through a clustering algorithm. We use clustering here as an approach to expedite the plagiarism detection process and claim together of the contributions. Therefore, we have developed two versions of this, one that performs clustering and the other that does not. Later, we compared the effect of clustering on the detection latency by performing the tests with both the versions this module [2]. The step that follows clustering is that the tri-gram construction and analysis. The tri-gram analysis is employed for measuring pairwise similarities among the assignments tested and therefore the results are presented to the user as percentage scores representing similarities.

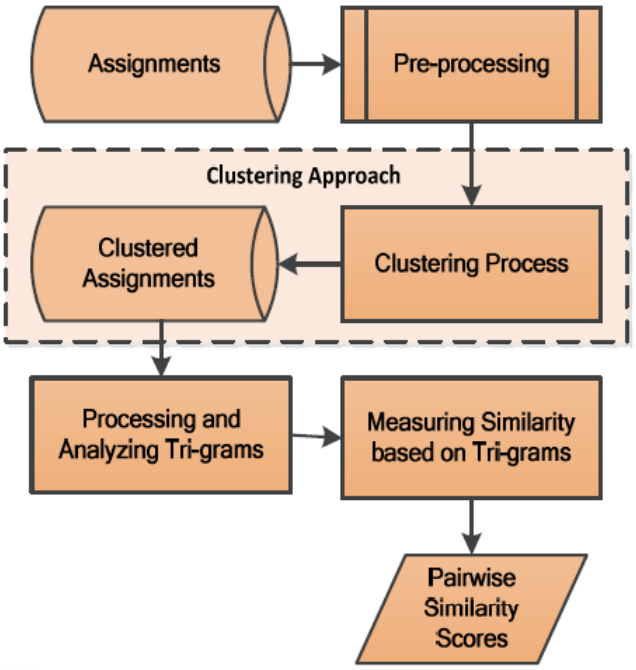

Fig. 1 Tri-gram sequence matching technique

A. Data Collection and File Conversion

Electronic text based assignments are collected as three isolated datasets. They were different in format and have been converted into plain text format to maintain all documents in the same format for fair treatment.

B. Pre-processing

The pre-processing step we performed, we consider as a crucial step for the plagiarism detection process. The purpose of pre-processing is to make suitable data which is to be input into the detection processes and to extend the effectiveness of the plagiarism detection tool. The corpus consists of a mixture of lower case and upper case characters. All upper case characters in the corpus have been transformed into lower case characters to eliminate case sensitivity from the corpus. All the diagrams, pictures and pictures within the corpus are removed. The delimiters within the corpus and stop words are identified and faraway from the corpus.

C. Tri-gram Construction

A tri-gram consists of three consecutive words sequence in each line. The tri-gram sequences are formed from the pre-processed assignments. Tri-grams are constructed by extracting collection of tri-gram sequences from the corpus. Figure 2 depicts the formation of tri-grams for an example sentence.

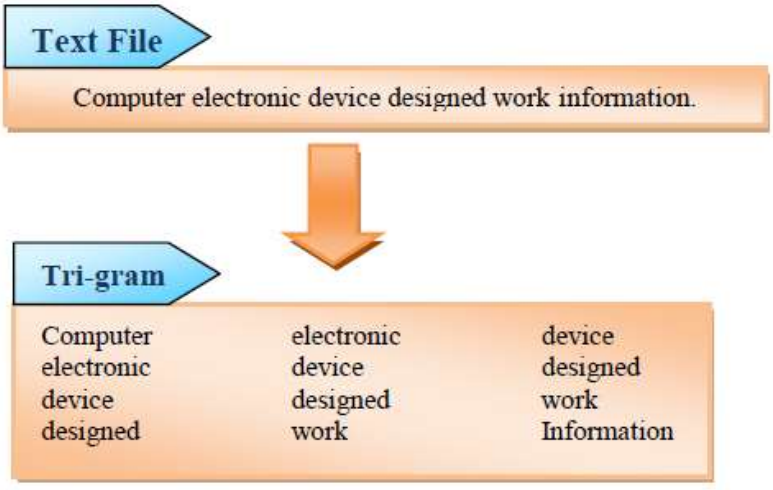

Fig. 2 Construction of tri-gram

D. Similarity Measure:

For similarity measure here we use two feature first is containment and second is Longest Common Subsequence.

Containment:

Let's say we are given a non-plagiarized source text and a solution text which will or might not be plagiarized. We must first extract the words from both text documents so as to make a corpus. Next, we count up the intersection of n-grams (sequential word groupings of n words) between the sources and answer texts. This will be checked out like counting the amount of words (1-grams) that both texts have in

common. Lastly, we normalize the worth by dividing by the entire number of n-grams within the answer text. The calculation for calculating containment is as follows:

$$\frac{\sum count(\overline{\text{ngram}}_A) \cap count(\overline{\text{ngram}}_S)}{\sum count(\overline{\text{ngram}}_A)}$$

Fig. 3 Calculation of containment

Longest Common Subsequence:

The longest common subsequence (LCS) is just the longest sequence of words left to right, present in both texts. Much like containment, LCS, may be a feature that computes the similarity of two texts. We can compute the LCS by iterating through each of the texts and counting up common words recursively; however this is often computationally exhaustive and for this reason we'll prefer to use a dynamic approach[3].

We will start with an example: Let's find the LCS between: a solution text — "ABCD" and a source text — "BD". Right away we will see that our answer text shares common characters. We start with an empty matrix; initializing the primary row and column with zeroes as shown within the first figure:

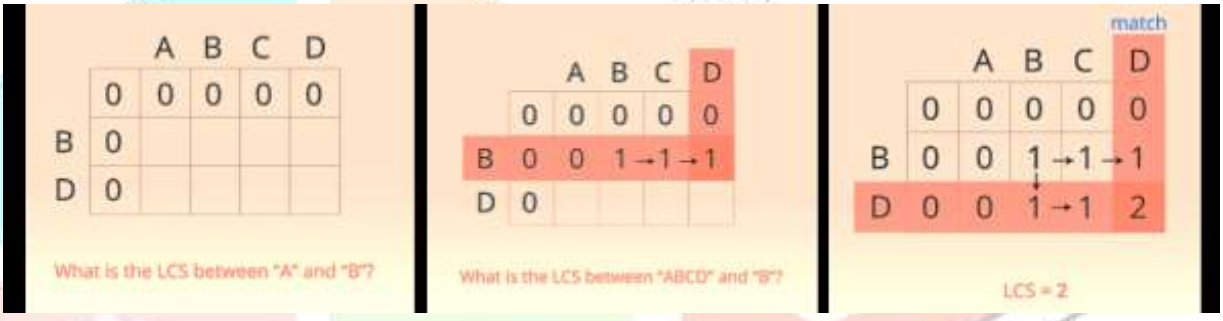

Fig. 4 Example of Longest Common Subsequence

Next, we iterate through each column, checking if there's a match. If there's a match, we add 1 and propagate down that row. After, we iterate through subsequent row, bring the 1 down from the previous row, and propagate. This provides us a worth of two - as we expected. Lastly we normalize this value by dividing by the length of the solution text. In our case, we dive 2 by 4 and acquire an LCS of 0.5. This is a rather high LCS indicating a high probability of plagiarism.

SageMaker:

For this project, I used SageMaker; Amazon's full stack machine learning service for building, training, and deploying predefined and custom machine learning models.

Amazon SageMaker could even be a completely managed service that gives every developer and data scientist with the power to form , train, and deploy machine learning (ML) models quickly. SageMaker removes the work from each step of the machine learning process to make it easier to develop high quality models.

Traditional ML development could even be a complicated , expensive, iterative process made even harder because there are not any integrated tools for the whole machine learning workflow. you would like to stitch together tools and workflows, which is time-consuming and error-prone. SageMaker solves this challenge by providing all of the components used for machine learning during a single toolset so models get to production faster with much less effort and at lower cost.

IV. RESULT

The dataset contains files with short questions and answers that have different levels of plagiarism. Original answers were included within the dataset. The goal of the project was to make a machine learning system that would detect if any given text is plagiarised or not. It clothed that the primary evaluated algorithm Random Forest delivered 100% accuracy in plagiarism detection. It means the machine learning system was ready to detect all plagiarism correctly.

```
accuracy: 1.0
              precision    recall  f1-score   suppo

         Non       1.00      1.00      1.00
           P       1.00      1.00      1.00

   micro avg       1.00      1.00      1.00
   macro avg       1.00      1.00      1.00
weighted avg       1.00      1.00      1.00


Predicted class labels:
[1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 0 1 0 1 1 0 0]

True class labels:
[1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 0 1 0 1 1 0 0]
```

Fig. 5 Results of random forest

V. CONCLUSION

Plagiarism of text has become a standard occurrence today with difficult to detect forms like paraphrasing and summarizing being frequently practiced. Hence, there's a requirement to style effective mechanisms for automatic plagiarism detection. During this work, a paraphrase recognition approach has been wont to detect the occurrence of plagiarism in source and suspicious passages.

VI. ACKNOWLEDGMENT

I would prefer to take this chance to increase my gratitude to Vishwakarma Institute of knowledge Technology, Pune, under Savitribai Phule Pune University for providing the platform for my work.

REFERENCES

[1] Chitra, A. & Rajkumar, Anupriya. (2015). Plagiarism Detection Using Machine Learning-Based Paraphrase Recognizer. Journal of Intelligent Systems. 10.1515/jisys-2014-0146.

[2] Vani, K., and Deepa Gupta. "Text plagiarism classification using syntax based linguistic features." Expert Systems with Applications 88 (2017): 448-464.

[3] E. Hunt et al., "Machine Learning Models for Paraphrase Identification and its Applications on Plagiarism Detection," 2019 IEEE International Conference on Big Knowledge (ICBK), Beijing, China, 2019, pp. 97-104, doi: 10.1109/ICBK.2019.00021.

[4] Patel, Ahmed & Bakhtiyari, Kaveh & Taghavi, Mona. (2011). Evaluation of cheating detection methods in academic writings. Library Hi Tech.

[5] Bin-Habtoor, A. & Zaher, Mahmoud. (2012). A Survey on Plagiarism Detection Systems. International Journal of Computer Theory and Engineering. 185-188. 10.7763/IJCTE.2012.V4.447.

[6] Kravchenko, Dmitry. (2018). Paraphrase Detection Using Machine Translation and Textual Similarity Algorithms. 277-292. 10.1007/978-3-319-71746-3_22.