Plagiarism Detection Using Deep Based Feature Combined with SynmDict

Ashish Varghese Muttumana, Harsh Goel, Yash Teotia, and Piyush Bhardwaj

Abstract Plagiarism in colleges is a significant issue, staying as a point for logical works for a considerable length of time. We can watch plagiarism happen in different fields like writing, scholastic, science, and music inconceivably. It very well may be likewise conceivable that one day we will get our task work in another production without legitimate reference. Plagiarism discovery systems are there, which are ordered into character-based strategy, basic based technique, characterization or group-based strategy, cross language-based methods, citation-based methods, semantic-based methods, and syntax-based methods. Different devices are accessible utilizing the above plagiarism strategies. Our tests show the viability of "deep features" in the undertaking of grouping task program entries as copy, partial-copy, and non-copy by bunching systems. Here, we have created a database containing sets of synonyms in the tabular form; it covers a variety of words containing a total of 100,000 words. This dataset helps to create an instantaneous feature for the specific dataset.

Keywords Plagiarism · SynmDict · Natural language processing · Deep learning · Deep feature · Deep deep-based feature · Machine learning

A. V. Muttumana (✉) · H. Goel · Y. Teotia · P. Bhardwaj
Bhagwan Parshuram Institute of Technology (affiliated to Guru Gobind Singh Indraprastha University, Delhi 110078), Rohini, Delhi 110089, India
e-mail: ashishvarghesem@bpitindia.com

H. Goel
e-mail: harshgoel@bpitindia.com

Y. Teotia
e-mail: yashteotia@gmail.com

P. Bhardwaj
e-mail: piyushbhardwaj@bpitindia.com

A. Abraham et al. (eds.), *Proceedings of 3rd International Conference on Computing Informatics and Networks*, Lecture Notes in Networks and Systems 167,
https://doi.org/10.1007/978-981-15-9712-1_5

1 Introduction

Plagiarism is viewed as scholastic dishonesty and disregard to journalistic ethics. It is liable to punishments, suspension, ejection from school or work, significant fines, and even detainment. As of late, instances of "extreme plagiarism" have been founded in the scholarly world [1]. Throughout history, it was discovered that some outstanding works were copied from past works that did not receive much fame, and the original author was deprived of credits. Thus, we require proper plagiarism detection software or systems which can detect plagiarism to the minute level hence providing the original author with the credit one deserves. In this paper, we have discussed the use of SynmDict[1] created by us in the project. We show that the state-of-the-art strategies can be effectively consolidated utilizing SynmDict via machine learning for an all the more dominant and flexible plagiarism identification apparatus. We additionally show that highlights can be developed from related fields of research and that these can help in ordering plagiarism [2]. We use SynmDict to make new information which includes these highlights that would be explicit for an informational index that exists just for that occasion. At the point when we enter two content document source and doubt, then profound element makes a component which is the mix of the considerable number of equivalent words that are in source and doubt. The size of this presented highlight is significantly less when contrasted with the total volume of SynmDict, which thus spare substantially more execution time in contrast with different calculations present. We have used the data collected from "PAN, a series of scientific events and shared tasks on digital text forensics and stylometry." The structure of paper contains five sections. The first covers the introduction, the second discusses the state-of-the-art methods used, the third elaborates our methodology, the fourth shows the results that we obtained, and the fifth section covers the conclusion and future scope.

2 State of Art

We have used the different text-based features in the method from state of the art.

2.1 Feature Engineering

Feature engineering is exceptionally vital while making machine learning models. Accuracy of predicted values of machine learning models highly depends on the feature vectors that have been chosen for the model; therefore, the real aim is to engineer such features that will help our machine learning pipeline [3].

[1] Database containing sets of synonyms in tabular form, having a total of 100,000 words.

2.2 Jaccard Similarity

The Jaccard similarity index checks the correlation between elements of two or more sets and gives an idea of how similar or different they are. The highest similarity is 100 percent while lowest is 0. For two sets A and B, it is calculated by an intersection B divided by A plus B minus A intersection B. It is used in a lot of plagiarism softwares as well as computer-vision software [4].

$$J(X, Y) = |X \cap Y| \div |XUY| = |X \cap Y| \div |X| + |Y| - |X \cap Y| \quad (1)$$

2.3 Dependency Parser

A dependency parser keeps a tab on the syntactic construction of a sentence, establishing a link between keywords and words which can modify these keywords. Dependency is a relationship that shows that a component, or set of components, requires other model components for their determination or implementation. The component is reliant upon the autonomous component, called the provider. At least two components right now called tuples [5].

2.4 N-Gram

N-gram is a set of words in sequence; it is usually collected from text or speech. It is beneficial for the detection of plagiarism as it provides a specific context which can be used for detection. There are different types of n-gram, depending on the value of n [6].

3 Methodology

3.1 Preprocessing

Stemming. It is a process of cutting off a part of the word either from the start or from the end to get to the root of the original word. For stemming in our research paper, we have used porter stemmer, which is very popular in stemming of words.

Tokenization. It is a process of breaking the text into smaller parts known as tokens; thus, all the.txt files in our dataset were broken into some important words known as tokens.

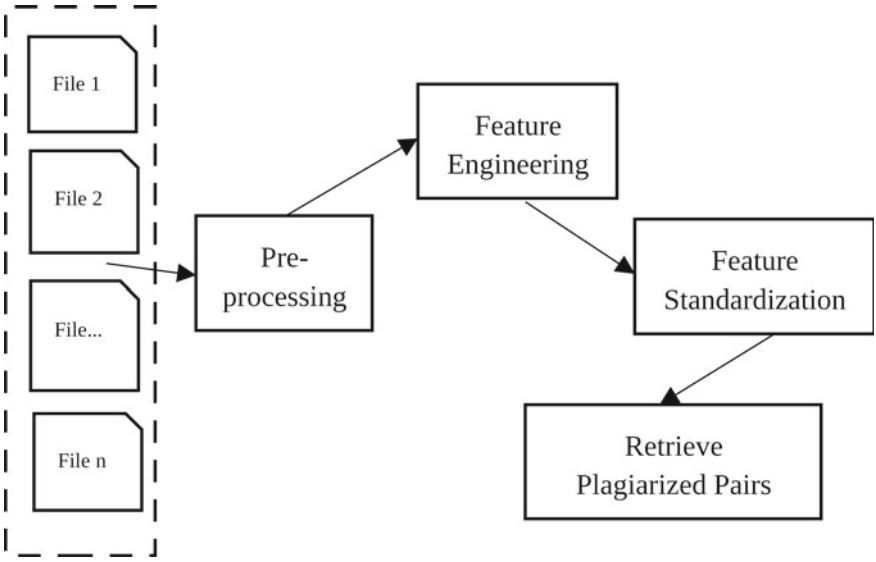

Fig. 1 Overview of the pipeline using an unsupervised learning-based approach

Stop-words. These are the words whose frequency is pretty high and does not add a sentiment value to the word; thus, all .txt files in our dataset eliminated all the stop-words, giving us the words that were of significant value.

We used regular expression and removed all the characters except for [a-zA-Z]. We converted all the words to the lowercase. Split the sentences to a set of words. We then removed all the stop-words. Stem the rest of the left words. Then, save these words in a data-frame. Repeat steps for rest of the text files (Fig. 1).

3.2 LSTM

LSTM is a neural system that is a piece of the repetitive neural system family. It is utilized in succession to arrangement expectation issues, for example, temperature-determining, hand-composing acknowledgement, and so on. We have to initially take a view at how neural systems (RNN and LSTM's) are prepared: first of all the forward propagation, then the mistake concerning the yield is estimated with which we figure angle in which this propagation proceeds. However, for different data sources the slope in deep neural systems are not steady. Along these lines, prior slopes are the result of later angles, and they will either increment or decline exponentially and in this way it cannot be stable. This is known as the exploding/vanishing tendency issue; this is the explanation why individuals use LSTM over RNN [7].

3.3 Text-Based Features

Difference in Length of Text (DLT). DLT captures the difference between the lengths of each text.

Similarity as Measured by Difference (SMD). SMD measures the number of lines of common terms in the original text extracted using different text.

Similarity in String Literals (SSL). SSL measures the similarity between the two sets of literals, one from each text file.

Jaccard Similarity and LCS (Longest Common Subsequence). Jaccard similarity and LCS are also used as text-based features.

3.4 SynmDict

It is a Dataset created which contains sets of synonyms in the tabular form; it covers almost 100,000 words.

3.5 Deep Based Feature

Deep Feature. A deep feature is a reliable reaction of a node or layer inside a various leveled model to info that gives a reaction that applies to the models' last yield. One feature is considered "deeper" than another relying upon how early in the decision tree or other framework, the reaction is enacted [8].

Deep based Feature. We use SynmDict to create new data features. These features would be specific for a data set that exists only for that instance. When we enter two text file sources and suspicious files, then deep feature creates a feature, which is the combination of all the synonyms that are in source and suspicious file. The size of this introduced feature table is much less as compared to the total size of SynmDict, which in turn saves much more execution time in comparison with other algorithms. This method is termed as "deep based features." The "deeper" layers can react and make their feature filters for increasingly complicated patterns in the input, for example, language blunder, evacuation of non-relevant words (such as the, a that, an, and so on), word tallying, or varieties of features processed earlier.

In Fig. 2, after the feature table, is made utilizing feature engineering and coordinating methodologies, information is sent to the RNN machine, which in turn uses LSTM strategy after classifier-based calculation giving the outcome as the group of various sorts of copy and no-copy information found in these two records and arranges as no-copy (0), partial-copy (1), copy (2).

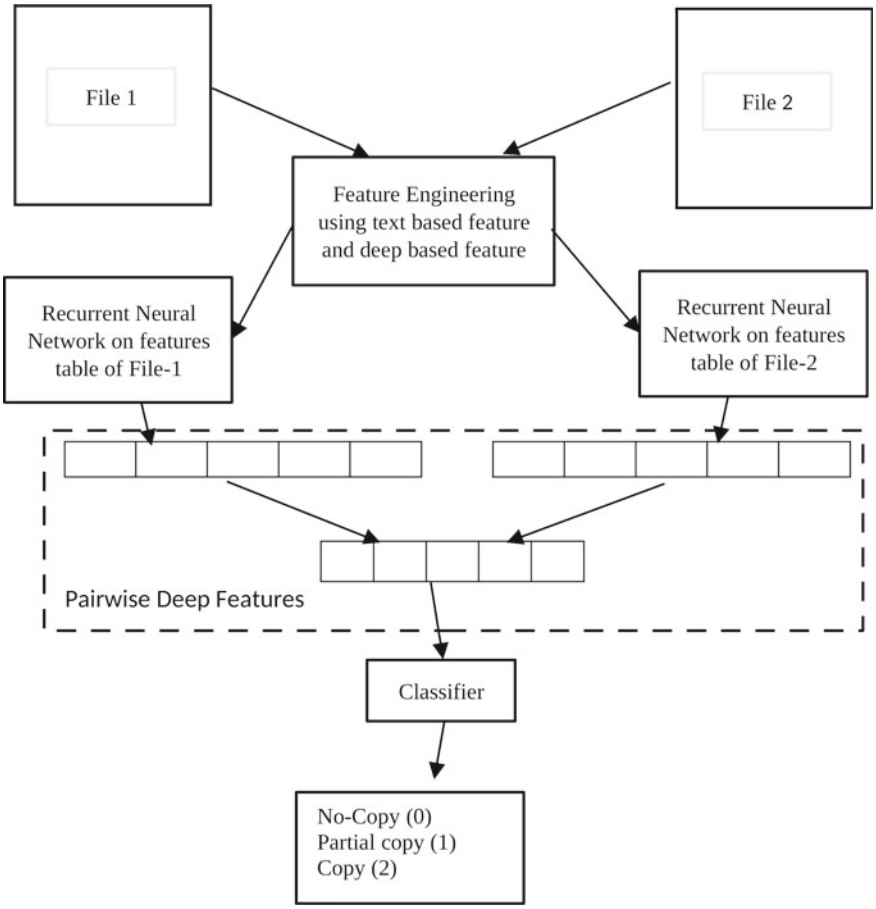

Fig. 2 RNN model approach using text and deep based features, which gives a percentage of no-copy, partial-copy, copy data from the source, and suspicious text files

4 Results

We have calculated the precision and recall value, which is further used to calculate the $F1$ score [9].

$$F1 = 2 * (\text{precision} * \text{recall})/(\text{precision} + \text{recall}) \tag{2}$$

These are the outcomes when text-based and deep based features are utilized. From the table, we can determine that when we use the text-based feature, it gives an $F1$ score of 0.42, and it gives an $F1$ score of 0.55. In any case, when we utilize both the features together, there is an exceptional change in the precision and recall value, which this way influences the $F1$ score. Here, we see that the $F1$ score gets

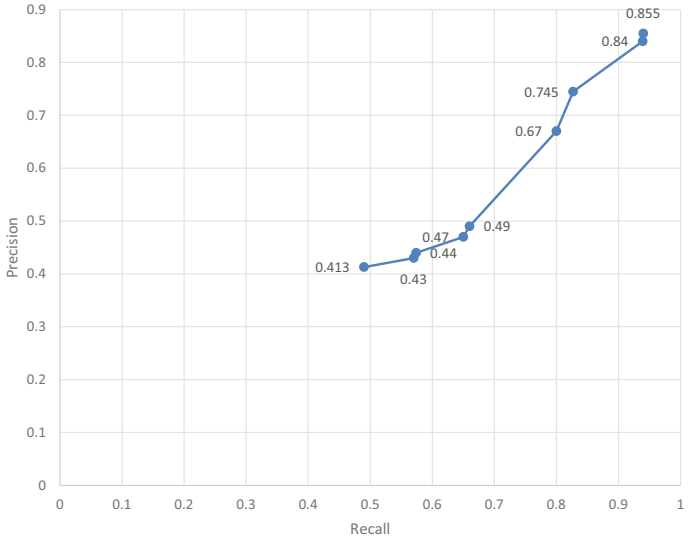

Fig. 3 Precision versus recall with each test

0.71, practically twofold than the individual strategies. In this manner, in the present state of the art, both deep feature and text feature is utilized.

4.1 SynmDict

Here, we have actualized content and profoundly based highlights using SynmDict in the RNN model. RNN machine uses the LSTM technique after classifier-based estimation, which gives the result as the gathering of different sorts of copy and no-copy data found in these two records and orchestrates as no-copy (0), partial-copy (1), copy (2). From the given table, we see that we achieved a pretty good improvement in the precision and recall value, which in turn provides a better $F1$ score of 0.91 in "no-copy" class.

4.2 Graphical Representation

Figure 3 shows different values of precision and recall values plotted in a line graph. It gives a piece of detail information about the improvements made by us during the timeline by conducting the different tests on the data provided by PAN[2] with each time improved SynmDict.

[2]PAN is a series of scientific events and shared tasks on digital text forensics and stylometry.

5 Conclusion and Future Works

In this paper, we used the database SynmDict, which is an acronym for synonym dictionary. It helps to create new instantaneous and data-specific features, which increases the precision and recall values. Only a particular part of the SynmDict is used for a dataset depending on the words in the dataset, thus increasing the speed while checking for plagiarism.

Multidimensional database. We can improve the SynmDict and increase the dimension of the table, which will increase accuracy as synonyms that are not words with the same meaning but similar meaning, and it changes with context. Multidimensional database may increase the efficiency drastically.

Tree-structured LSTM. Utilization of tree-structured LSTM, recursive recurrent neural networks can encourage us to get familiar with some portrayal to grow such detection frameworks [10].

References

1. Blum SD (2011) My word! Plagiarism and college culture. Cornell University Press
2. Kalleberg RB (2015) Towards detecting textual plagiarism using machine learning methods (Master's thesis, Universitetet i Agder; University of Agder)
3. Bengio Y, Courville A, Vincent P (2013) Representation learning: A review and new perspectives. IEEE Trans Pattern Anal Mach Intell 35(8):1798–1828
4. Niwattanakul S, Singthongchai J, Naenudorn E, Wanapu S (2013) Using of Jaccard coefficient for keywords similarity. Proc Int Multi-Conf Eng Comput Sci 1(6):380–384
5. Nivre J (2005) Dependency grammar and dependency parsing. MSI Rep 5133(1959):1–32
6. Younes N, Reips UD (2019) Guideline for improving the reliability of Google Ngram studies: Evidence from religious terms. PloS One 14(3)
7. Sherstinsky A (2020) Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Phys D Nonlinear Phenom 132306
8. Jaderberg M, Vedaldi A, Zisserman A (2014) Deep features for text spotting. In: European conference on computer vision. Springer, Cham, pp 512–528
9. Sokolova M, Japkowicz N, Szpakowicz S (2006) Beyond accuracy, Fscore and ROC: a family of discriminant measures for performance evaluation. In: Australasian joint conference on artificial intelligence. Springer, Berlin, Heidelberg, pp 1015–1021
10. Tai KS, Socher R, Manning CD (2015) Improved semantic representations from tree-structured long short-term memory networks. arXiv:1503.00075