Plagiarism Checker

Soha Ashraf
Faculty of Informatics and Computer Science- AI Major
The British University in Egypt
Cairo, Egypt
soha205001@bue.edu.eg

Mohammed Amein
Faculty of Informatics and Computer Science- AI Major
The British University in Egypt
Cairo, Egypt
mohammed211320@bue.edu.eg

Mohammed Essam
Faculty of Informatics and Computer Science- AI Major
The British University in Egypt
Cairo, Egypt
mohammed217764@bue.edu.eg

Omar Tarek
Faculty of Informatics and Computer Science- AI Major
The British University in Egypt
Cairo, Egypt
omar213109@bue.edu.eg

***Abstract*— Plagiarism checker or text similarities is a common problem in natural language processing (NLP) research area. It helps to retain credibility and prevent criminal consequences by identifying and removing copied text. Plagiarism checker which uses cosine similarity and an additional similarity measure to assess the semantic similarity between texts. To evaluate the similarity of documents, the NLP Plagiarism Check use a commonly accepted metric for comparing vectors called cosine similarity. Additionally, a secondary similarity measure, such as Jaccard similarity or Word Mover's Distance (WMD), will be incorporated to enhance the accuracy and effectiveness of plagiarism checker. It involves training the model to measure the similarities of the documents.**

Keywords—plagiarism detection, cosine similarity, Jaccard similarity, levenshtien distance, word mover's distance, word embedding.

I. INTRODUCTION

The issue of text similarity and plagiarism detection is well known in the field of natural language processing research. Optimizing results and reducing time spent on analysis of documents is one of the key challenges in this area. To detect plagiarism, a variety of approaches have been developed, mostly involving text parsing by various algorithms and thresholds for the number of words that are similar. [1]

Exploring different aspects of plagiarism, its various types and the methodologies of plagiarism detection. Also to demonstrate the different textual features of documents as a condition for the quantification of documents and the similarity measures for plagiarism detection.

There are two types of plagiarism: Literal plagiarism, which is more sophisticated and difficult to detect, and it involves exact or modified copies of the text and Intelligent Plagiarism. Intelligent Plagiarism, where the meaning of the original text is preserved but presented in a modified or disguised form. It includes Text Manipulation, Translation, and Idea Adoption. The complexity and nuances involved in detecting and dealing effectively with plagiarism are highlighted in these categories.

The semantic similarity is the comparison of two texts to calculate how close or similar these two texts it also known as lexical similarity. To measure the text similarity, there are several ways to measure by using:

- **String-based** plagiarism detection operates on sequences of characters and their compositions. It encompasses two subclasses:
    1. **Character-based detection**, which aids in identifying typographical errors by analyzing individual characters.
    2. **Term-based or Token-based detection**, which is valuable for detecting rearrangements of terms by breaking strings into substrings.
- **Corpus-based**, which relies on information gathered from a large corpus to estimate similarity between concepts.
- **Knowledge-based,** which measures semantic similarity by considering rules of inference, logical propositions, and semantic networks such as taxonomies and ontologies in its representation schema.
- **Hybrid similarity,** which aims to the usage of the advantages of the previous classes to create a better approach such as string- based, knowledge-based and corpus-based similarity.

For measuring the text similarity, it is important to determine the distance between two texts. The query document (DQ) and source documents (D) are split into smaller parts known as fingerprints of length K, which are then processed to generate hash values using a hash function. With each document's vector consisting of a set of its unique fingerprints, these hash values are arranged and compared to other documents. For the identification of the similarity ratio between two texts, a text vector similarity is helpful. To specify the ratio of similarity between documents, some distance measures are used to measure distances between two vectors that represent those documents. Combining these techniques and setting thresholds allows to find and calculate similarity between texts and aiming for plagiarism detection. In Natural Language Processing (NLP), the detection of plagiarism is becoming a major research area to detect instances of text reuse, transformation, or reproduction across different forms. It is important to Preprocessing texts before using NLP techniques, as they improve the accuracy of detecting plagiarism.

Text preprocessing is a critical step in preparing data for machine learning algorithms, particularly for Natural Language Processing (NLP) tasks. It involves cleaning and transforming text to facilitate better performance of algorithms by converting it into a more consumable format. Key steps in text preprocessing include:

1. Text normalization: Converting characters representing letters from different languages into characters from the language being used, ensuring uniform encoding across all texts.
2. Stop Word and Special Character Removal: Removing commonly occurring words, known as stop words, and irrelevant special characters like punctuation marks, which can slow down document processing.
3. Stemming and Lemmatization: Stemming involves heuristic removal of word endings, while lemmatization analyzes words morphologically to return their base or dictionary form.
4. Tokenization: Breaking down sentences into individual words or tokens to facilitate word-by-word processing.
5. Text Representation: Numerically representing the feature vector of text, enabling computational processing.

These preprocessing steps are essential for optimizing NLP tasks and improving the accuracy of machine learning models applied to text data. [2] [3].

II. LITERATURE REVIEW

Plagiarism Detection in the Bengali Language: A Text Similarity-Based Approach

In this paper, the authors made plagiarism detection tools for other language which is Bengali as the most exist plagiarism detection tool in English text. the authors collected Bengali literature from the National Digital Library of India and developed a corpus. The authors achieved in extracting text ranging from **72.10%: 79.89%** accuracy rate through the utilization of optical character recognition (OCR). By using the Levenshtein Distance algorithm, plagiarism detection was performed with successful result.

The curated collection of processed and organized texts, serving various purposes such as data mining, natural language processing, and plagiarism detection is referred to as a corpus. The aim is to create a specific corpus for plagiarism detection and ensure its potential use in other applications. In the proposed methodology, the authors outline for generating a corpus of Bengali literature a systematic approach. It begins with a collection of Bengali literature, comprising 200 books selected from the National Digital Library of India, which includes both old and new works. To extract Bengali texts from copies of scanned PDF, the OCR technology is employed in the second step. To eliminate non-Bengali characters, extraneous whitespaces and invalid characters, text cleanup is performed in the third step. In the final step, the result of the third step, which is text cleanup, is stored in the database and the corpus generation process is completed. In the implementation phase, the authors preprocessing the used dataset, then performed OCR using Tesseract, which is an open source to recognize any language as it supports more than 100 languages and Bengali is one of them. It is also for using the pre-trained model. Text similarity between two documents is the proposed algorithm. To measure the minimum number of the necessary edits to changing or transforming one string to another one, by using Levenshtien distance or as referred to as Edit Distance. This algorithm can be applied by using three editing operations which are: insertion, replacement, or removal. With dynamic programming or recursive solution, the Levenshtien distance can be implemented.

In the implementation phase, the application of plagiarism checker first takes the input from the user. Then the algorithm tokenizes the input and deletes the stopwords for preprocessing the data and to improve the accuracy. The input will be comprised with every page of the whole book to find the similarity and store it in the database. After the comparison, the system will retrieve the information of the pages where similarity scores met predefined thresholds and indicate the possible cases of plagiarism.

The authors noted the successful results of their system. They found that the plagiarism detection accuracy correlates with the volume of the input text provided the more text, the better detection.

Plagiarism Detection using Word Mover's Distance (WMD):

Word Mover Distance (WMD) is a novel approach to measuring document similarity that has gained significant attention in the field of natural language processing and information retrieval. Introduced by Kusner(2015)[1], WMD leverages word embeddings to quantify the semantic similarity between documents. Unlike traditional metrics that solely rely on word overlap or co-occurrence, WMD takes into account the distributional semantics of words and their relationships within the embedding space. By considering the optimal transportation cost required to move word embeddings from one document to another, WMD captures the semantic distance between their respective word distributions. its ability to utilize the high quality of the word2vec embedding. This innovative approach has shown very good results in plagiarism detection, text summarization, and document clustering.

Scalable and language-independent embedding-based approach for plagiarism detection considering obfuscation type: no training phase

This paper presents a scalable language-independent approach for plagiarism detection. The method utilizes text embedding vectors to compare document similarity. the paper employed the cosine similarity algorithm to compare sentence representations. Each sentence is represented as an average of word embedding vectors as in the equation

$$S_i = \frac{\sum_{j=1}^{n} w_{ij}}{n},$$

where Si is the vector representation for sentence i, wij is the word vector for the jth word of sentence i, and n is the number of words in the sentence.

The resulting vector is then used for cosine similarity comparison between source and other documents. The highest cosine similarity above a predefined threshold nominates a plagiarism candidate.

The paper utilized pre-trained word embeddings for English from GoogleNews, trained word vectors using the word2vec algorithm for Persian, and utilized pre-trained Arabic word vectors from fastText for Arabic. Evaluation on English, Persian, and Arabic datasets employed three parameter tuning methods: offline threshold tuning, offline threshold tuning with respect to obfuscation type, and automatic language-independent parameter tuning with respect to obfuscation type (online tuning). Results showed online tuning performed comparably or better than offline approaches in some cases, outperforming offline tuning on English and Arabic corpora and performing as well as offline tuning for Persian. The paper achieved competitive performance in text alignment, ranking third in English for summary obfuscation type and fourth for no-obfuscation cases. Overall, the approach demonstrated promising results across various languages and obfuscation types without requiring training data.

A. Maintaining the Integrity of the Specifications

Identify applicable funding agency here. If none, delete this text box.

III. Prepare Your Paper Before Styling

A. Abbreviations and Acronyms

B. Units

C. Equations

D. Some Common Mistakes

IV. Using the Template

A. Authors and Affiliations

1) For papers with more than six authors: Add author

2) For papers with less than six authors: To change the

a) extra authors.

B. Identify the Headings

C. Figures and Tables

References

(PDF) NLP based Deep Learning Approach for plagiarism detection, https://www.researchgate.net/publication/347682908_NLP_based_Deep_Learning_Approach_for_Plagiarism_Detection (accessed Mar.24, 2024).

M. A. El-Rashidy, R. G. Mohamed, N. A. El-Fishawy, and M. A. Shouman, "Reliable plagiarism detection system based on Deep Learning Approaches - neural computing and applications," SpringerLink, https://link.springer.com/article/10.1007/s00521-022-07486-w (accessed Mar. 24, 2024).

Rahutomo, F., Kitasuka, T., & Aritsugi, M. (2012, October). Semantic cosine similarity. In The 7th international student conference on advanced science and technology ICAST (Vol. 4, No. 1, p. 1).

Ghosh, S., Ghosh, A., Ghosh, B., & Roy, A. (2022). Plagiarism Detection in the Bengali Language: A Text Similarity-Based Approach. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2203.13430

[5] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In Proceedings of the 32nd International Conference on Machine Learning, volume 37 of Proceedings of Machine Learning Research, pages 957–966, Lille, France.

[6] https://www.researchgate.net/publication/337087795_Scalable_and_language-independent_embedding-based_approach_for_plagiarism_detection_considering_obfuscation_type_no_training_phase