ISSN: 0067-2904

Using Retrieved Sources for Semantic and Lexical Plagiarism Detection

Ayoub Ali M. Saeed[1*], Alaa Yaseen Taqa[2]
[1]*Department of Mathematics, College of Basic Education, University of Mosul, Mosul, Iraq*
[2]*Department of Computer sciences, College of Education for Pure Sciences, University of Mosul, Mosul, Iraq*

Received: 21/2/2022 Accepted: 17/9/2022 Published: 30/6/2023

Abstract

Plagiarism is described as using someone else's ideas or work without their permission. Using lexical and semantic text similarity notions, this paper presents a plagiarism detection system for examining suspicious texts against available sources on the Web. The user can upload suspicious files in pdf or docx formats. The system will search three popular search engines for the source text (Google, Bing, and Yahoo) and try to identify the top five results for each search engine on the first retrieved page. The corpus is made up of the downloaded files and scraped web page text of the search engines' results. The corpus text and suspicious documents will then be encoded as vectors. For lexical plagiarism detection, the system will leverage Jaccard similarity and Term Frequency-Inverse Document Frequency (TFIDF) techniques, while for semantic plagiarism detection, Doc2Vec and Sentence Bidirectional Encoder Representations from Transformers (SBERT) intelligent text representation models will be used. Following that, the system compares the suspicious text to the corpus text. Finally, a generated plagiarism report will show the total plagiarism ratio, the plagiarism ratio from each source, and other details.

Keywords: Plagiarism, Plagiarism detection, Term Frequency-Inverse Document Frequency, Doc2vec, Sentence Bidirectional Encoder Representations from Transformers.

استعمال المصادر المسترجعة للكشف عن الانتحال الأكاديمي

أيوب علي محمد سعيد[1*] , الاء ياسين طاقة[2]
[1]قسم الرياضيات، كلية التربية الأساسية، جامعة الموصل، الموصل، العراق
[2]قسم علوم الحاسوب، كلية التربية للعلوم الصرفة، جامعة الموصل، الموصل، العراق

الخلاصة

يوصف الانتحال او السرقة العلمية بأنه استعمال أفكار او اعمال شخص آخر دون إذنه. باستعمال مفاهيم تشابه النص المعجمي والدلالي ، تقدم هذه الورقة نظامًا للكشف عن الانتحال لفحص النصوص المشبوهة مقابل المصادر المتاحة على الويب. يمكن للمستخدم تحميل الملفات المشبوهة بصيغة pdf أو .docx سيقوم النظام بالبحث في ثلاثة محركات بحث شائعة عن النص المصدر وهي Google , Bing وYahoo ومحاولة تحديد أفضل خمس نتائج لكل محرك بحث في الصفحة المسترجعة الأولى. تتكون قاعدة بيانات من الملفات التي تم تنزيلها ونص صفحة الويب المستخلصة لنتائج محركات البحث. سيتم بعد ذلك ترميز نص المجموعة والوثائق المشبوهة كمتجهات. للكشف عن الانتحال المعجمي ، سيستفيد النظام من تشابه Jaccard وتقنيات

*Email: ayobali-1980@uomosul.edu.iq

TFIDF، بينما سيتم استعمال نماذج Doc2Vec و SBERT الذكية والخاص بترميز النص للكشف عن الانتحال الدلالي. بعد ذلك ، يقارن النظام النص المشبوه بالنصوص الموجودة في قاعدة البيانات. أخيرًا ، سيعرض تقرير الانتحال الذي تم إنشاؤه إجمالي نسبة الانتحال ونسبة الانتحال من كل مصدر وتفاصيل أخرى.

1. Introduction

Plagiarism is a type of deception in which someone steals someone else's work and then lies about it. It is not just about copying words; it is also about stealing other people's phrases, ideas, and work. While the exponential growth of the Internet has made plagiarism easier than ever, it has also made it easier to verify or uncover plagiarism [1].

Plagiarism may be classified into several categories. The most frequent kinds are classified based on their intensity as follows:

1. Significant plagiarism: In academia, this is the most common type. The plagiarist rewrites the original text in this case and substitutes synonyms for the terms.
2. Minimal plagiarism: occurs when some other text is inserted into the original text and the patterns of the original text are altered.
3. Complete plagiarism: All text is taken from other source texts with no changes, and the author claims that it is his work [2, 3].

Plagiarism can involve several kinds of modifications in the texts, such as the following:

1. Lexical modifications: the terms of the text can be added, deleted, reordered, or replaced.
2. Synthetic modifications: include changing passive to active sentences and vice versa.
3. Semantic modifications: This type includes paraphrasing as well as semantic and word alterations [2, 4].

Plagiarism Detection (PD) is defined as detecting cases of plagiarism within a text or document. The text similarity metric determines how similar two texts are. The notion of text similarity is used by many organizations to detect plagiarism. The challenge of detecting plagiarism in texts can be described as a problem of computing text similarity. The PD is one of the common applications of text similarity and the measure of similarity, among texts is a viewpoint for classifying approaches of PD [5].

PD can be done as: [6]

1. Manual detection: needs enormous work and extra memory. When a vast number of texts must be compared, it has become impractical and almost impossible.
2. Software-based detection: This enables the comparison of a large number of documents, making detection easier so that the findings are considerably more acceptable [3].

Furthermore, PD methods are classified as follows:

1. Extrinsic PD: aims to find plagiarism in the text through testing wholly of the available sources to determine the degree of similarity between a reference collection and a suspicious text [7, 8] .
2. Intrinsic PD: examines a suspicious document and attempts to identify sections of it that were not produced by the same author [9].

Plagiarism can occur within a single language (monolingual), such as English-English, or among two or more languages (multilingual), such as English-Chinese [10, 11].

Natural Language Processing (NLP) is a branch of computer science and artificial intelligence dealing with computer-human (natural) language interactions. It is used in several text analysis applications, such as plagiarism detection[9, 12].

The main contributions of this paper are:

1. Build a dynamic corpus for each suspicious document, utilizing the abilities of web search engines.

2. Combine Jaccard and TFIDF with cosine similarity to detect lexical plagiarism. Hybrid approaches to detect semantic plagiarism include Doc2vec and SBERT.
The focus of this paper is on external plagiarism in monolingual English texts. The paper is prearranged as follows: Related studies on the detection of plagiarism are presented in section 2. The adopted approaches and methodologies are described in Section 3, while the proposed system is presented in Section 4. The results from the experiments are explained in Section 5, and Section 6 will present the discussion. Finally, Section 7 discusses conclusions.

2. Literature Review

Plagiarism is threatening the growth and prosperity of academic communities, especially with the introduction of the Web, which is why efforts must be made to avoid it. Several plagiarism detection approaches are implemented. Some of those approaches are reviewed in Table 1.

Table 1: Reviewed studies

Ref.	Description	Cons
[13]	Implemented a Google API-based PD system named "Spotting and Neutralizing Internet Theft by Cheaters" (SNITCH), Each text is looked up on the internet. A concise report in the form of an annotated HTML page which includes statistics on the ratio of plagiarism, and the required time to accomplish the verification.	It supports only analysis of text documents and the concise report in the form of an annotated HTML page only.
[6]	They used the TF-IDF method to represent text numerically in order to detect plagiarism. The accuracy of the assignment was determined by counting the number of correctly assigned documents and dividing them by the total number of documents.	They have not given any results for the implementation of the system.
[14]	The web-based PD system is made up of two main modules: one global component based on heuristics; The searches are subsequently sent to Google via its API for search operation. A report of similarity for the suspicious document will be created, where the plagiarized sections will be highlighted with different colors to show where they came from. A report of similarity for the suspicious document will be created, where the plagiarized sections will be highlighted with different colors to show where they came from.	It uses only Google search engine, the maximum number of queries per day set by Google is 100 for free subscription account.
[15]	Described a web-based antiplagiarism technique at the academic level. The Google search API is used to get specific keywords or key phrases from a given Web content. The findings are shown in the form of a URL by the tool.	Only Google search engine is used. The acceptable text file formats are (.txt, .doc) only.
[16]	Used the concept of the k-Nearest Neighbor Algorithm (k-NN) machine learning (ML) technique to compare a set of text with some existing multiple files to determine the copied component.	The ratio of plagiarism for each source file is displayed only as one of two words (plagiarized, not plagiarized) and not as a detailed report.
[17]	Proposed a web tool for verifying multilingual texts (English and Arabic). The program searches the internet for duplicate material using three common engines for web searches: Yandex SERP, Bing, and Google. They used TFIDF and the cosine text similarity approach. For each suspicious sentence, the HTML report indicates if the sentence is plagiarized or not.	For each suspicious sentence, the HTML report indicates if the sentence is plagiarized or not. So, it only performs plagiarism detection at the sentence levels.

The previous reviewed papers did not consider the semantic approach for plagiarism detection and did not generate a colored PDF plagiarism detection report. Furthermore, they used only some sentences directly or saved them as text files, so to overcome the existing cons, the purpose of this paper is to design an online source retrieval-based system for plagiarism detection.

This system uses Application Programming Interfaces (APIs) of search engines Google, Yahoo and Bing to utilize the specific advantages of each one and can examine the suspicious document to detect plagiarism lexically and semantically. The suspicious document may be a pdf or docx file. The result of the search will be either web pages to scrape their text contents or pdf or docx file formats to download them. The citation for "quotes" removal was used in the article, which is used to eliminate sentences that appear inside quotation marks. When it comes to documents like research papers, journal papers, and articles, if the text is presented inside quote marks, it is not considered plagiarized. Furthermore, the cited sentences are ignored in detecting the plagiarism.

3. Methodology

The corpus, text processing techniques, and plagiarism detection algorithms employed in our tests are described in this section.

3.1. The corpus

The corpus is used in plagiarism detection projects to measure the similarity of suspicious documents and calculate the plagiarism percentage[18]. The corpus can be a pre-existing off-line dataset to which the experiments are applied, or it can be on-line sources on the World Wide Web [19, 20].

Since we do not have the huge corpus that the common plagiarism detection systems have, like the Turnitin and ithenticate platforms, this paper will propose to build a dynamic corpus by scraping the related HTML web pages, extracting the text by parsing those pages, and collecting text data by downloading free source documents such as pdf or docx files from the web.

3.2. Text Representation

A text representation or word embedding is a function that maps a word or a sentence to a small-dimensional vector, with the distance between vectors indicating how similar the words and sentences that correspond to the vectors are[21].

Many classical and intelligent models for PD are needed to represent or encode text as numeric vectors. Some of the text representation approaches are mentioned in the next paragraphs [22] . Recent advances in neural networks have made creating a distributed representation that accurately reflects word similarity from real-world data simple [23].

3.2.1. Term Frequency – Inverse Document Frequency (TFIDF)

TFIDF is a technique for text weighting that is frequently utilized in conjunction with cosine similarity to detect the similarity of two texts [24].

The TFIDF algorithm considers the frequency of various terms in all papers and is capable of distinguishing them. Term Frequency is abbreviated as TF, while Inverse Document Frequency is abbreviated as IDF[7, 25].

The following is the equation to calculate the weight of a term in a single document [25] :

$$W_{t,d} = TF_{t,d} * IDF_t \quad (1)$$

Where:
$W_{t,d}$: weight of term "t" in single document "d".
$TF_{t,d}$: The frequency with which the term t (Term) appears in the document d.
IDF_t = The frequency of inverse documents as calculated by equation 2 [25] :

$$IDF_t = \log\left(\frac{N}{n_t}\right) \quad (2)$$

Where:
N = Total of wholly documents
n_t = The total number of documents that contain the term "t". The IDF shows the difference in the term "t" in each text by reflecting the spread of the term throughout the document. The spread of the terms in a document is represented by TF.
TF-IDF is an excellent approach for calculating term weights because it can make exceptions for high-frequency terms that have little in common[7, 26].

3.2.2. Document to Vectors (Doc2Vec)

Word embedding is a sort of word representation as a numeric vector. It uses a low-dimensional vector to contain contextual information. Doc2vec is an unsupervised ML intelligent technique for generating vector representations of phrases, paragraphs, and documents. It does this in a very easy way: it considers a piece of text as a particular type of word [2]. In this technique, a text is converted into a vector that reflects the degree of significance of a specific word in the text.

Because the Doc2vec model retains the context of words encountered, the entire document may be plotted as a vector depending on its semantic meaning.

The Doc2Vec equivalent of the skip-gram model is termed Distributed Bag-of-Words (DBOW) (Figure 1-(a)), while the CBOW analogue is called Distributed Memory (DM) (Figure 1-(b)) [2]. The $w_{p;n}$ denotes the nth word in passage p in these diagrams, and p denotes a unique identification for a passage (or document). As a result, one can use the DM model to add the passage's identity to each context created from it.

Unlike the skip-gram model, which attempts to detect a context for a particular word, the DBOW model attempts to detect a context for a particular passage. A context in this example is a word sequence produced from a passage by selecting a text window at random.
In the distributed memory setup, a paragraph vector functions as an object that recalls when it was trained with which words [10].

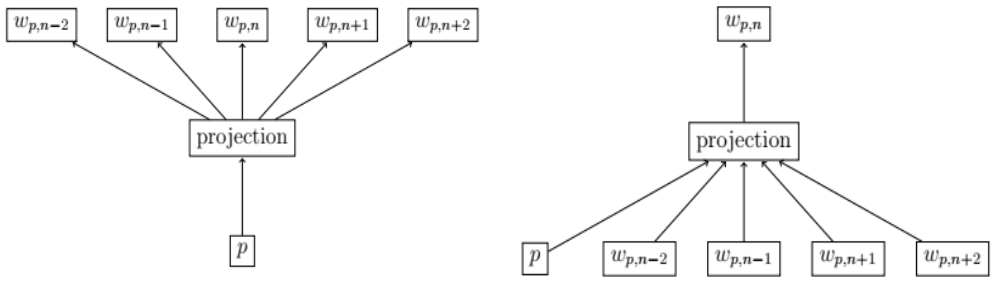


Figure 1: (**a**) Distributed Bag-of-Words (DBOW) type of doc2vec technique for word embedding (**b**) Distributed Memory (DM) type of doc2vec technique for word embedding [2]

3.2.3. Sentence Bidirectional Encoder Representations from Transformers (SBERT)

The Bidirectional Encoder Representations from Transformers (BERT) is a text representation paradigm made up of many transformer encoder blocks stacked on top of each other. Rather than extracting a word's semantic meaning, the entire sentence is considered, utilizing the Deep Learning (DL) paradigm. The BERT learning process is divided into two stages: pre-training and fine-tuning. The BERT model learned by predicting the masked word token in the pre-training procedure, which involved randomly masking word tokens in a phrase from a huge corpus. Fine-tuning is the process of relearning a previously trained BERT model with labelled data[27].

Sentence-BERT (SBERT) is a pre-trained BERT network that uses Siamese and Triplet network architectures to generate semantically relevant embeddings for each sentence, so that the generated embeddings can be compared using cosine-similarity [28].

SBERT's network structure (Figure 2) is determined by the training data supplied. Experiment with the structures and objective functions shown below. The cosine similarity between the two embeddings u and v is calculated.

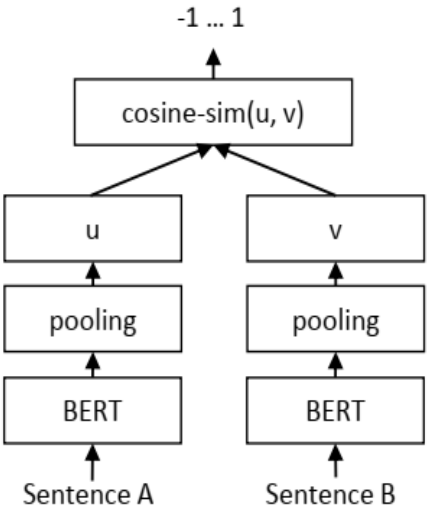


Figure 2: SBERT architecture at inference, for example, to compute similarity scores [28]

3.3. Approaches of text similarity

In general, the similarity method receives two texts and outputs the degree of similarity between them. The two things represented by numbers are the values produced by the similarity function range between [0,1] [29].

There are numerous measures for calculating similarity in the literature. Two of the most famous are Jaccard and cosine similarity[30].

Jaccard Similarity is a technique that uses a count-based co-occurrence measure and is used to determine text similarity. The number of elements in the intersection set divided by the number of elements in the union set can be used to calculate the Jaccard coefficient as follows [31]:

$$Jaccard\ (S,D) = \frac{(S\ \cap\ D)}{(S\ \cup\ D)} = \frac{|S\ \cap\ D|}{|S| + |D| - |S\ \cup\ D|} \quad (3)$$

Where |S| and |D| denotes to the word count for specious and source texts respectively, $|S\ \cap\ D|$ and $|S\ \cup\ D|$ are the count of words in intersection and count of words in union between suspicious and source texts, respectively.

The computation of similarities between two vectors by looking for cosines from the angle between them is known as cosine similarity, and it is commonly used in text mining to compare documents [32].

The following is the equation for calculating cosine similarity [9] :

$$Cos\ Similarity(x,y) = \frac{x.y}{||x||||y||} \quad (4)$$

Where:
y , x: the vector of dot product between x and y, calculated by :

$$x.y = \sum_{k=1}^{n} x_k\, y_k \quad (5)$$

||x|| : the length of the vector x, as determined by :

$$||x|| = \sum_{k=1}^{n} x_k^2 \quad (6)$$

||y|| : the length of the vector y, as determined by :

$$||y|| = \sum_{k=1}^{n} y_k^2 \quad (7)$$

The larger the values of the similarity function, the more similar the two items assessed are, as stated in [17] . If the reverse is true, then the lower the value of the similarity function, the more distinct the two items are thought to be.
Manhattan and Euclidian distances are two common dissimilarity measures. A dissimilar measure (d) can be used to calculate a similarity measure (s) by subtracting d from 1 [24].

The Cosine similarity metric is superior to others because even if two text documents are separated by significant distances, they are likely to be similar in terms of context [33]. Besides that, several authors got the best results when using cosine similarity.

4. The proposed system

This paper presents a source retrieval plagiarism detection system. It detects text plagiarism online by utilizing the APIs of three prominent search engines (Google, Yahoo, and Bing) for lexical and semantic text similarity concepts. It receives a text document with a plagiarism threshold as input and produces a report that informs the user if the document is unique or plagiarized lexically, semantically, or both lexically and semantically, as well as other information such as the source text and the whole percentage of plagiarism. The structure of the presented system is illustrated in Figure 3.

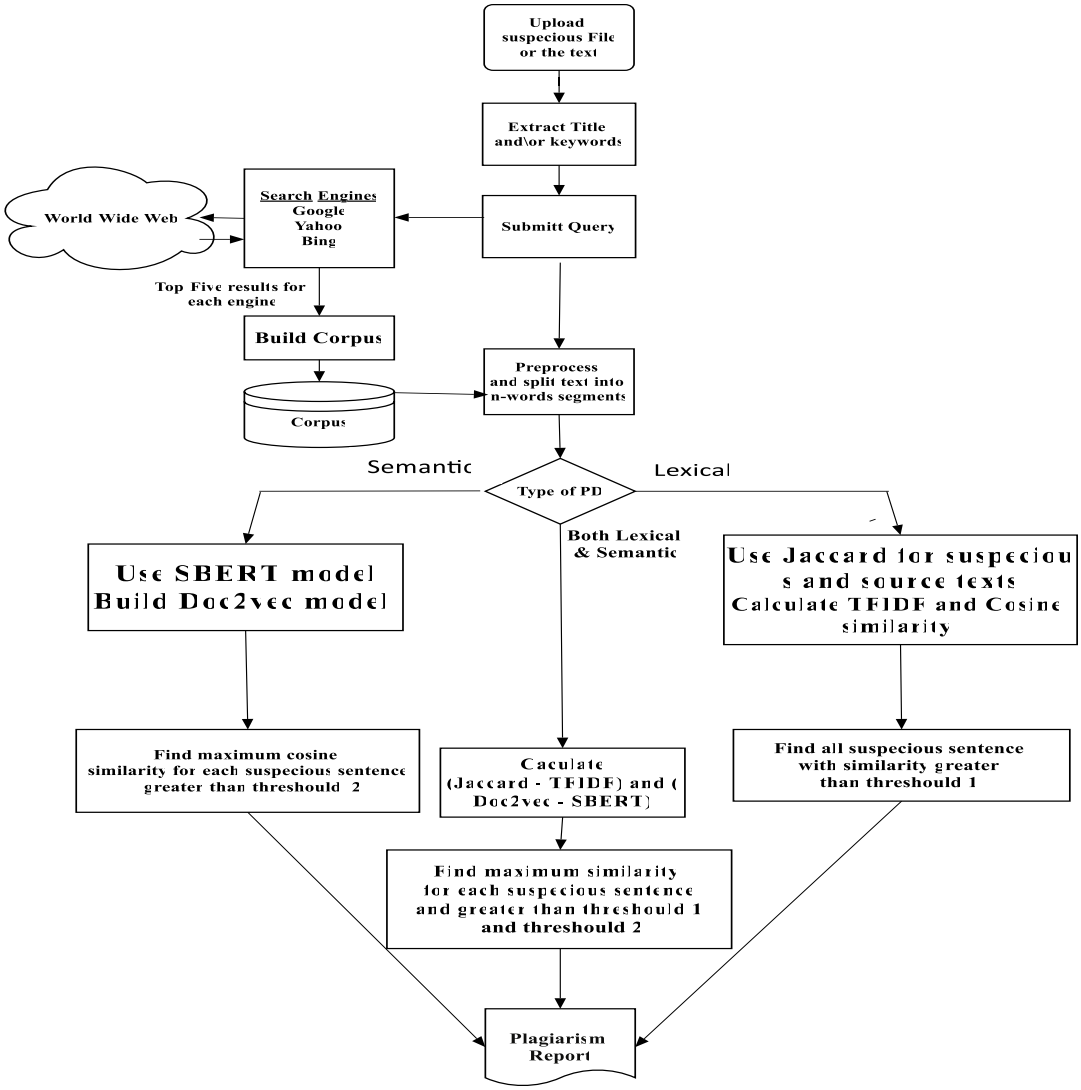


Figure 3: Structure of proposed system

The system proceeds in several stages, which will be explained in the next sections. There are three kinds of experiments: lexical, semantic, or both lexical and semantic plagiarism detection approches are implemented. In all three situations, the final similarity report for the suspicious text will be created, with the plagiarized sentences marked in several colors that identify the source, and the unique sentences will not be highlighted.

4.1. Lexical plagiarism detection using Jaccard Similarity -TFIDF with Cosine similarity

This can be done in two approaches after preprocessing and splitting them into n-words:

1- Jaccard Similarity : apply Jaccard similarity between suspicious and original texts to obtain lexical PD.

2- The texts are represented as TFIDF vectors. After that, the cosine similarity is applied to detect whether a sentence is lexically plagiarized or not .

The lexical plagiarism detection process includes several stages, as follows:

Stage 1 (**Input Text**): The user uploads a scientific file such as a pdf or docx formatted file. The user also inputs a plagiarism detection threshold and a number n to split texts into n-words per sentence.

Stage 2 (**Extract Title and Keywords**): Extract the title and keywords from the uploaded scientific paper file using the Regex and NLTK libraries of Python.
Stage 3 (**Submit the query**): This stage uses three widespread search engines: Google, Yahoo, and Bing APIs and attempts to scrape the top five results for each search engine if it's a webpage or download and store the obtained pdf or docx files using the bs4 and request libraries of Python according to the title and keywords extracted from the previous stage.

Stage 4 (**Build Corpus**): Convert the scraped webpages and downloaded documents to text files in order to build the corpus.

Stage 5 (**Pre-process using NLP**): The suspicious text and the source texts are pre-processed at this stage using NLP techniques that exist in the NLTK library of Python, including:
1. Tokenizing the text.
2. Removing stop words, single characters, punctuations, non-alphabetic symbols and numbers.
3. Convert all capital letters to small letters.
4. Lemmatization of verbs.
5. Split text into sentences with n words in each sentence according to the user's desire.

Considering the ability to remove the quoted and cited paragraphs from the suspicious text using the Regular Expressions (Regex) library in Python.

Stage 6 (**Text representation**):
Use Jaccard similarity: Apply the Jaccard similarity with each of the suspicious document sentences and source sentences. Suppose that J(S,D) is the Jaccard similarity between suspicious sentence S and source sentence D.
Calculate TFIDF text encoding for each of the source documents' sentences and suspicious document sentences. For each TFIDF vector of suspicious document sentences, apply the cosine similarity to each of the TFIDF vectors of source document sentences. Suppose that C(S,D) is the cosine similarity between suspicious sentence S and source sentence D.
Stage 7 (**Similarity measure**):
The suspicious sentence S is lexically plagiarized if the following Eq. 8 is satisfied:

$$LSim(S,D) = \frac{J(S,D) + C(S,D)}{2} \geq thresold\ 1 \tag{8}$$

Select the largest value that is greater than the threshold1 and mark it as lexical plagiarism.
Stage 8 (**The Result**) : Generate the plagiarism report as a web page and as a PDF file. Each plagiarized sentence is highlighted with a color that refers to the source document or web page from which this sentence is plagiarized. The report contains a percentage of plagiarism, a count of characters, a number of words in the suspicious text, and other information. The value of threshold1 must be inputted by the user to detect lexical plagiarism. The value must be chosen in the range [0,1] to carry out experiments.

4.2. Semantic Plagiarism detection using Doc2Vec - SBERT with cosine similarity

The text is preprocessed and splitted into n-words. Then the Doc2vec and SBERT models are used to represent text as vectors. After that, the cosine similarity is applied to detect whether a sentence is semantically plagiarized or not.

The stages of semantic plagiarism detection are similar to those of lexical plagiarism detection except that stages 6 and 7 will be replaced by the following :

Stage 6:
1. Use the Doc2Vec intelligent model: Build the Doc2Vec model for the text files of the corpus. Each text will be converted to a list of vectors using the Doc2Vec paradigm.
2. Use the SBERT deep learning model: Each of the suspicious and source sentences will be encoded using the SBERT model.

Stage 7 (Similarity measure):
1. Doc2Vec model: Find cosine similarity for each text vector of suspicious file sentences and find the largest similar value with each of the vectors of source file sentences. Suppose that CD(S,D) is the cosine similarity between suspicious sentence vector S and source sentence vector D.
2. SBERT model: For each suspicious sentence vector, apply the cosine similarity to the source sentence vectors. Suppose that CS(S,D) is the Cosine similarity between suspicious sentence vector S and source sentence vector D.

The suspicious sentence S is plagiarized semantically if the following Eq. 9 is satisfied:

$$SSim(S,D) = \frac{CS(S,D) + CD(S,D)}{2} \geq threshold\ 2 \quad (9)$$

The value of threshold2 must be inputted by the user to detect semantically plagiarized sentences. The value must be chosen in the range [0,1] to carry out experiments.

4.3. Lexical and Semantic plagiarism detection using (Jaccard - TFIDF) and (Doc2vec - SBERT) with cosine similarity

The text is preprocessed and splitted into n-words, then TFIDF, Doc2vec, and SBERT models are used to represent the text as two vectors, the first vector to check for lexical plagiarism and the second vector to detect semantic plagiarism.
In this case, stages 6 and 7 of lexical and semantic PD will be merged. So, each sentence of the suspicious document S is marked as lexical and semantically plagiarized LSSim if the following Eq. 10 is satisfied using equations 8 and 9:

$$LSSim(S,D) = \frac{LSim(S,D) + SSim(S,D)}{2} \geq \frac{threshold\ 1 + threshold\ 2}{2} \quad (10)$$

5. Results

The steps of the proposed system in this paper were applied to ten suspicious documents. No matter what the type of document is, pdf or docx, in any case, it will be converted to text to be handled according to the system. Meanwhile, if the suspicious file is a research or journal article, then the system will extract its title and keywords to use them as a query in the search engine. The result of the search will be either web pages to scrape their text contents or pdf or docx file formats to download them to build the related corpus of that file.
Table 2 illustrates information about the ten suspicious documents, including types and the count of words in each.

Table 2: Information of suspicious documents

DOC. NO.	TYPE	COUNT OF WORDS

1	Pdf	3165
2	Pdf	4328
3	Pdf	5203
4	Pdf	4782
5	Pdf	5139
6	Pdf	4943
7	Docx	3758
8	Docx	4829
9	Docx	3825
10	Docx	5104

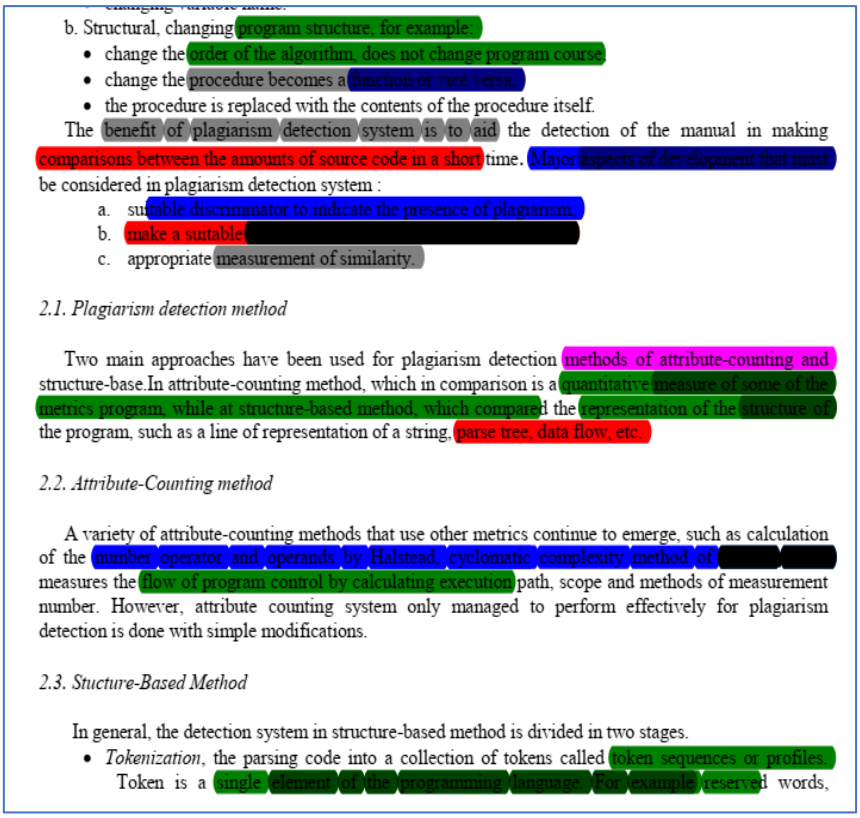
The results of lexical plagiarism are shown in Table 3, as percentages of plagiarism, with n-words equal to 5 and threshold1 = 0.2 as an experiment.

The result of the similarity measure between two texts may be 0, which means completely dissimilar (not plagiarized), or 1, which means identically similar (plagiarized). The other values between 0 and 1 reflect the limit of similarity/dissimilarity between the texts. Therefore, any value could be chosen as a threshold. If the value of the similarity measure is greater than or equal to the threshold, it is considered a similarity (plagiarized), otherwise it is considered a dissimilarity (not plagiarized). So, any value between 0 and 1 could be chosen as threshold1 and threshold2.

Table 3: Percentages of Lexical plagiarism

DOC. NO.	LEXICAL PLAGIARISM PERCENTAGES
1	0.22
2	0.17
3	0.26
4	0.32
5	0.39
6	0.59
7	0.45
8	0.21
9	0.27
10	0.39

Figure 4 depicts a snapshot of the lexical plagiarism report of one suspicious document. Each plagiarized sentence is highlighted with a color that refers to the source document or web page on which this sentence is plagiarized, while Figure 5 illustrates the total and partial plagiarism percentages for a suspicious document. It shows the source document titles along with the author and website address if the source is a webpage.



b. Structural, changing program structure, for example:

- change the order of the algorithm, does not change program course.
- change the procedure becomes a function or vice versa.
- the procedure is replaced with the contents of the procedure itself.

The benefit of plagiarism detection system is to aid the detection of the manual in making comparisons between the amounts of source code in a short time. Major aspects of development that must be considered in plagiarism detection system :

a. suitable discriminator to indicate the presence of plagiarism.
b. make a suitable
c. appropriate measurement of similarity.

2.1. Plagiarism detection method

Two main approaches have been used for plagiarism detection methods of attribute-counting and structure-base.In attribute-counting method, which in comparison is a quantitative measure of some of the metrics program, while at structure-based method, which compared the representation of the structure of the program, such as a line of representation of a string, parse tree, data flow, etc.

2.2. Attribute-Counting method

A variety of attribute-counting methods that use other metrics continue to emerge, such as calculation of the number operator and operands by Halstead, cyclomatic complexity method of measures the flow of program control by calculating execution path, scope and methods of measurement number. However, attribute counting system only managed to perform effectively for plagiarism detection is done with simple modifications.

2.3. Stucture-Based Method

In general, the detection system in structure-based method is divided in two stages.

- *Tokenization*, the parsing code into a collection of tokens called token sequences or profiles. Token is a single element of the programming language. For example reserved words,

Figure 4: A snapshot of Plagiarism report

Plagiarism Ratio Report

Suspecious File :

Total ratio of Plagiarism lexically is 26.91%

Original Sources

1 Overview of Plagiarism Checkers and Plagiarism Detection Tools: A Study Dr. Sudhir S. Patil — 6.89 %

2 https://www.semanticscholar.org/paper/Plagiarism-detection-in-learning-management-system-Sabonchi — 5.38 %

3 https://dl.acm.org/doi/pdf/10.5555/2667199.2667216 — 4.95 %

4 Process Model Improvement for Source Code Plagiarism Detection in Student Programming Assignments Dragutin KERMEK, Matija NOVAK — 3.98 %

5 A Survey on Plagiarism Detection Bhardwaj Akanksha, Arya Anukruti and Vyas Tarjni — 2.74 %

6 www.scribd.com — 1.18 %

7 www.dialog-21.ru — 0.70 %

8 A Survey on Plagiarism Detection Techniques and Understanding Textual Features Heena Sherwani, Mitesh Bargadiya — 0.38 %

Figure 5 : Total and partial plagiarism percentages

The results of semantic plagiarism are shown in Table 4 with n-words equal to 5 and threshold2 = 0.3 as an experiment, while Table 5 presents the results of semantic and lexical plagiarism with n-words equal to 5, threshold1 = 0.2 and threshold2 = 0.3 as an experiment.

Table 4 : Percentage of Semantic plagiarism.

DOC. NO.	SEMANTIC PLAGIARISM PERCENTAGES
1	0.23
2	0.19
3	0.14
4	0.27
5	0.31
6	0.21
7	0.26
8	0.18
9	0.13
10	0.18

Table 5: Percentages of Lexical and Semantic plagiarism

Doc. No.	*Lexical & Semantic plagiarism percentages*
1	0.18
2	0.12
3	0.8
4	0.13
5	0.10
6	0.16
7	0.14
8	0.7
9	0.11
10	0.19

Plagiarism has been detected in a sample of 10 documents with different percentages, where the highest percentage is 0.8% and the lowest percentage is 0.11%.

To evaluate the system, the research utilized some common metrics, such as accuracy, precision, recall, and F1-score.

The typical calculation of accuracy is obtained through formula 11 as follows [34]:

$$Accuracy = \frac{TP + TN}{(TP + TN + FP + FN)} \tag{11}$$

in which (True Negatives: TN) are clean texts correctly classified as plagiarized, (True Positives: TP) are plagiarized texts correctly classified as clean, (False Positives: FP) are clean texts incorrectly classified as plagiarized, and "clean" (False Negatives: FN) are plagiarized texts incorrectly classified as "clean"[34]. Table 6 shows the accuracy, precision, recall and F1-score obtained for the suspicious documents. The lexical plagiarism has higher accuracy since it is based on the TFIDF statistical text representation and Jaccard similarity, while the semantic plagiarism has lower accuracy since it is an intelligent approach for text representation.

Table 6: Overall evaluation of the Lexical, Semantic and Lexical & Semantic PD

DOC NO.	TEXT REPRESENT METHOD	ACCURACY	PRECISION	RECALL	F1-SCORE
1	Lexical	0.836	0.864	0.881	0.819
	Semantic	0.742	0.779	0.707	0.740
	Lex. & Sem.	0.625	0.664	0.593	0.626
2	Lexical	0.853	0.879	0.894	0.833
	Semantic	0.711	0.751	0.782	0.714
	Lex. & Sem.	0.594	0.636	0.669	0.600
3	Lexical	0.827	0.856	0.874	0.811
	Semantic	0.722	0.761	0.791	0.723
	Lex. & Sem.	0.683	0.626	0.560	0.590
4	Lexical	0.889	0.887	0.893	0.863
	Semantic	0.785	0.786	0.723	0.758
	Lex. & Sem.	0.643	0.672	0.653	0.641
5	Lexical	0.875	0.885	0.913	0.854
	Semantic	0.725	0.763	0.793	0.738
	Lex. & Sem.	0.623	0.646	0.675	0.649
6	Lexical	0.865	0.873	0.891	0.827
	Semantic	0.742	0.772	0.796	0.741
	Lex. & Sem.	0.692	0.648	0.571	0.623
7	Lexical	0.905	0.883	0.896	0.847
	Semantic	0.822	0.792	0.731	0.762
	Lex. & Sem.	0.696	0.684	0.643	0.648
8	Lexical	0.912	0.897	0.913	0.874
	Semantic	0.725	0.779	0.796	0.738
	Lex. & Sem.	0.643	0.653	0.681	0.642
9	Lexical	0.908	0.899	0.936	0.913
	Semantic	0.734	0.795	0.825	0.784
	Lex. & Sem.	0.697	0.656	0.683	0.619
10	Lexical	0.915	0.956	0.892	0.891
	Semantic	0.754	0.785	0.793	0.773
	Lex. & Sem.	0.697	0.642	0.661	0.634

6. Discussion

The system is focused on merging the results of several search engines in order to benefit from the specific advantages of each search engine in and eliminate some of the common blunders. Depending on the search methodology used by each search engine, each engine will return different results.

It is critical to guarantee that the project or design model produces reasonable outcomes and meets its objectives, which is why evaluation metrics are used to assess the project. Several constraints must be addressed in order to improve the accuracy and applicability of the proposed system. The most significant of these constraints are the accuracy of the three search engine

results (Google, Bing, and Yahoo), the hardware of the laptop being used, and the speed of the internet.

7. Conclusions and Recommendations

This paper has provided a plagiarism detection system, which is an online plagiarism detection tool that allows users to examine a text for duplicate material on the Internet. The system takes text as input. The title or keywords of the text are extracted to create a query, which is then sent to three prominent search engines (Bing, Google, and Yahoo) through the API of each search engine to utilize the web and collect possible source texts from it. The approach of text similarity is then used to determine whether or not the provided material was plagiarized from texts found on the internet. Finally, for the provided text, a similarity report will be created, in which the plagiarized content will be highlighted by utilizing different colors to identify the original texts.

The present outcomes are encouraging, demonstrating that integrating several search engines yields better results than using each search engine alone. The used approaches are shown to be efficient, fast, and easy; querying a sentence on a search engine instantly returns several sources linked to the query.

In the future, further experimental research is required to extend the presented system by evaluating the performance of other similarity metrics and utilizing popular off-the-shelf word embedding methods like Stanford GloVe, Facebook fastText, and other deep learning-based approaches. Besides that, it is possible to utilize an existing offline corpus to calculate the plagiarism percentage with or without calculating the plagiarism percentage on-line.

Acknowledgements

The authors would like to express their gratitude and thanks to the College of Computer Sciences and Mathematics at the University of Mosul for their assistance in the development of this research.

References

[1] D. Sorokina, J. Gehrke, S. Warner, and P. Ginsparg, "Plagiarism detection in arXiv," in *Sixth International Conference on Data Mining (ICDM'06)*, 2006: IEEE, pp. 1070-1075.

[2] H. El Mostafa and B. Faouzia "A New Online Plagiarism Detection System based on Deep Learning," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 11, no. 9, 2020.

[3] T. Foltýnek *et al.*, "Testing of support tools for plagiarism detection," *International Journal of Educational Technology in Higher Education*, vol. 17, no. 1, pp. 1-31, 2020.

[4] E. Gharavi, K. Bijari, K. Zahirnia, and H. Veisi, "A Deep Learning Approach to Persian Plagiarism Detection," *FIRE (Working Notes)*, vol. 34, pp. 154-159, 2016.

[5] K. Baba, T. Nakatoh, and T. Minami, "Plagiarism detection using document similarity based on distributed representation," *Procedia computer science*, vol. 111, pp. 382-387, 2017.

[6] V. G. R. Durga Bhavani Dasari "Detecting The Plagiarism For Text Documents On The World Wide Web," *International Journal of Social Relevence & Concern*, vol. 2, no. 10, 2014.

[7] I. Indriyanto and I. Sumitra, "Measuring the Level of Plagiarism of Thesis using Vector Space Model and Cosine Similarity Methods," in *IOP Conference Series: Materials Science and Engineering*, 2019, vol. 662, no. 2: IOP Publishing, p. 022111.

[8] S. Muhammad, "Mono- and cross-lingual paraphrased text reuse and extrinsic plagiarism detection," PhD, Computing and Communications Data Science Institute, Lancaster University, England, 2020.

[9] F. K. AL-Jibory, "Hybrid System for Plagiarism Detection on A Scientific Paper," *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, no. 13, pp. 5707-5719, 2021.

[10] M. Elamine, F. Bougares, S. Mechti, and L. H. Belguith, "Extrinsic plagiarism detection for French language with word embeddings," in *International Conference on Intelligent Systems Design and Applications*, 2019: Springer, pp. 217-224.
[11] M. N. Mansoor and M. S. Al-Tamimi, "Computer-based plagiarism detection techniques: A comparative study," *International Journal of Nonlinear Analysis and Applications*, vol. 13, no. 1, pp. 3599-3611, 2022.
[12] D. Sakamoto and K. Tsuda, "A detection method for plagiarism reports of students," *Procedia Computer Science*, vol. 159, pp. 1329-1338, 2019.
[13] S. Niezgoda and T. P. Way, "SNITCH: a software tool for detecting cut and paste plagiarism," *ACM SIGCSE Bulletin*, vol. 38, no. 1, pp. 51-55, 2012.
[14] I. H. Khan, M. A. Siddiqui, and K. Mansoor, "A framework for plagiarism detection in Arabic documents," in *Proceedings of the Conference on Computer Science & Information Technology*, 2015, pp. 1-9.
[15] J. Khatri and V. Mohan, "An Approach for Implementing Web-Based Tool for Plagiarism Detection," *International Journal of Engineering and Management Research (IJEMR)*, vol. 6, no. 3, pp. 57-60, 2016.
[16] M. Sahu, "Plagiarism detection using artificial intelligence technique in multiple files," *International Journal 0f Scientific and Technology Research*, vol. 5, no. 4, 2016.
[17] M. Alabbas, R. S. Khudeyer, M. Radif, and H. K. Hameed, "Online Multilingual Plagiarism Detection System Using Multi Search Engines," *Journal of Southwest Jiaotong University*, vol. 54, no. 6, 2019.
[18] M. Ilyas, N. Malik, A. Bilal, S. Razzaq, F. Maqbool, and Q. Abbas, "Plagiarism Detection Using Natural Language Processing Techniques," *Technical Journal*, vol. 26, no. 01, pp. 90-101, 2021.
[19] H. Asghari, O. Fatemi, S. Mohtaj, and H. Faili, "A crowdsourcing approach to construct mono-lingual plagiarism detection corpus," *International Journal on Digital Libraries*, vol. 22, no. 1, pp. 49-61, 2021.
[20] H. Polat and S. Oyucu, "Building a Speech and Text Corpus of Turkish: Large Corpus Collection with Initial Speech Recognition Results," *Symmetry*, vol. 12, no. 2, p. 290, 2020.
[21] C.-Y. Chang, S.-J. Lee, C.-H. Wu, C.-F. Liu, and C.-K. Liu, "Using word semantic concepts for plagiarism detection in text documents," *Information Retrieval Journal*, vol. 24, no. 4, pp. 298-321, 2021.
[22] J. D. Thom, "Combining tree kernels and text embeddings for plagiarism detection," Stellenbosch: Stellenbosch University, 2018.
[23] F. Khaled and M. S. H. Al-Tamimi, "Plagiarism Detection Methods and Tools: An Overview", *Iraqi Journal of Science*, vol. 62, no. 8, pp. 2771–2783, Aug. 2021.
[24] T. Mardiana, T. B. Adji, and I. Hidayah, "The Comparation of distance-based similarity measure to detection of plagiarism in Indonesian text," in *International Conference on Soft Computing, Intelligence Systems, and Information Technology*, 2015: Springer, pp. 155-164.
[25] S. Chawla, P. Aggarwal, and R. Kaur, "Comparative Analysis of Semantic Similarity Word Embedding Techniques for Paraphrase Detection," *EasyChair*, 2516-2314, 2021.
[26] M. Umadevi, "Document comparison based on tf-idf metric," *International Research Journal of Engineering and Technology (IRJET)*, vol. 7, no. 02, 2020.
[27] Y. Yoo, T.-S. Heo, Y. Park, and K. Kim, "A novel hybrid methodology of measuring sentence similarity," *Symmetry*, vol. 13, no. 8, p. 1442, 2021.
[28] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.
[29] S. Thaiprayoon and C. Haruechaiyasak, "WEB PLAGIARISM DETECTION BASED ON SEARCH RESULT SNIPPETS."
[30] H. Veisi, M. Golchinpour, M. Salehi, and E. Gharavi, "Multi-level text document similarity estimation and its application for plagiarism detection," *Iran Journal of Computer Science*, pp. 1-13, 2022.
[31] N. Pradhan, M. Gyanchandani, and R. Wadhvani, "A Review on Text Similarity Technique used in IR and its Application," *International Journal of Computer Applications*, vol. 120, no. 9, 2015.
[32] Š. Suchomel and M. Brandejs, "Source retrieval for plagiarism detection," *Journal of Advances in Information Technology Vol*, vol. 6, no. 1, 2015.

[33] A. B. Mausumi Goswami, B.S Purkayastha, "A Comparative Analysis of Similarity Measures to find Coherent Documents," *International Journal of Management, Technology And Engineering,* vol. 8, no. XI, pp. 786-797, 2018.

[34] A. Chitra and A. Rajkumar, "Plagiarism detection using machine learning-based paraphrase recognizer," *Journal of Intelligent Systems,* vol. 25, no. 3, pp. 351-359, 2016.