Multimedia Tools and Applications
https://doi.org/10.1007/s11042-023-15703-4

# An effective text plagiarism detection system based on feature selection and SVM techniques

Mohamed A. El-Rashidy[1] · Ramy G. Mohamed[1] · Nawal A. El-Fishawy[1] · Marwa A. Shouman[1]

Received: 3 December 2021 / Revised: 10 February 2022 / Accepted: 23 April 2023

## Abstract

Text plagiarism has greatly spread in the recent years, it becomes a common problem in several fields such as research manuscripts, textbooks, patents, academic circles, etc. There are many sentence similarity features were used to detect plagiarism, but each of them is not discriminative to differentiate the similarity cases. This causes the discovery of lexical, syntactic and semantic text plagiarism types to be a challenging problem. Therefore, a new plagiarism detection system is proposed to extract the most effective sentence similarity features and construct hyperplane equation of the selected features to distinguish the similarity cases with the highest accuracy. It consists of three phases; the first phase is used to preprocess the documents. The second phase is depended on two paths, the first path is based on traditional paragraph level comparison, and the second path is based on the computed hyperplane equation using Support Vector Machine (SVM) and Chi-square techniques. The third phase is used to extract the best plagiarized segment. The proposed system is evaluated on several benchmark datasets. The experimental results showed that the proposed system obtained a significant superiority in the performance compared to the systems with a higher ranking in the recent years. The proposed system achieved the best values 89.12% and 92.91% of the Plagdet scores, 89.34% and 92.95% of the F-measure scores on the complete test corpus of PAN 2013 and PAN 2014 datasets, respectively.

**Keywords** Text plagiarism · Natural language processing · Classification · Feature Selection

✉ Marwa A. Shouman
marwa.shouman@el-eng.menofia.edu.eg

Mohamed A. El-Rashidy
mohamed.elrashedy@el-eng.menofia.edu.eg

Ramy G. Mohamed
ramygamalawad23@el-eng.menofia.edu.eg

Nawal A. El-Fishawy
nawal.elfishawy@el-eng.menofia.edu.eg

[1] Department of Computer Science and Engineering, Faculty of Electronic Engineering, Menoufia University, Shebeen El-Kom, Egypt

## 1 Introduction

There is a huge amount of data available on the Internet, and it is also easily accessible, which has led to the emergence of text plagiarism. IEEE defines text plagiarism as "The reuse of someone else's prior ideas, processes, results, or words without explicitly acknowledging the original author and source" [8]. In general, plagiarism can occur in any form of data such as text, music, images, videos, and codes. Several studies of digital plagiarism have found that 79.5 % of the authors are implicated in digital plagiarism [22]. It is a serious problem for scientific publications and the academic community. Therefore, automated plagiarism detection systems are needed.

Plagiarism detection systems mainly deal with three types of changes used to convert the original text into the plagiarized text: lexical, syntactic, and semantic changes. The lexical changes are accomplished by adding and removing words from the text and using synonyms or concepts with similar meanings as a replacement of words. The syntactic changes are carried out by altering the syntax of sentences such as sentence restructuring, active-passive transformations, etc. Finally, semantic changes are a mixture between lexical strategy and syntactic strategy, which is the most difficult strategy for researchers nowadays. Therefore, semantic knowledge databases are developed in English language such as WordNet [34], Gene Ontology [18], and Transfer Standard [43]; in order to integrated with plagiarism detection systems for determining conceptual similarity without the need of human interaction. There are two types of the plagiarism detection systems: intrinsic and extrinsic [10]. In the intrinsic plagiarism detection system, the suspected documents are compared basing on the stylometric features, which were included within the documents. But in the extrinsic plagiarism detection system, the suspected document is compared with external source documents.

Recently, plagiarism detection systems have attempted to come up with an approach that can handle different types of the text plagiarism [23–25, 28, 51, 54, 58] by extracting lexical, syntactic and semantic characteristics. These approaches represent the extracted characteristics into vectors, and apply similarity measurements as cosine criterion, jaccard criterion, dice criterion, match criterion, etc. Afterward in the recent researches [2, 3, 13, 33, 52, 57, 64, 65, 67, 68], some researches proposed new equations for measuring the similarity of sentences, these equations depended on the integration of two criteria to calculate the resemblance of sentences and words, and the others relied on the integration of three or four sentence similarity features. Each similarity feature within the proposed equations in the previous researches is multiplied by a weighting coefficient. To find the appropriate value of the weighted coefficient in the previous researches, a lot of experiments were done to compare the results to find the best values of the criteria weights. The purposed systems that relied on two, three or four criteria instead of relying on one criterion have been developed to achieve a high level of accuracy for detecting the text plagiarism, and discover the different images of the lexical, syntactic, and semantic plagiarisms.

Natural Language Processing (NLP) is a field of computer science, linguistics and artificial intelligence dealing with computer-human interactions (natural). The processing of different human languages by computer systems is the aim of NLP. In many fields, NLP plays a significant role including computer-assisted language acquisition, search engine optimization, and biological data extraction [48]. In this paper, NLP approaches including feature selection and classification techniques are utilized to detect the different types of text plagiarism with the highest accuracy.

The previous research [2, 3, 23–25, 28, 39, 51, 54, 58] depended on 2, 3 or 4 sentence similarity features instead of depending on only one feature to enhance the text plagiarism detection. There is a challenge to depend on few numbers of sentence similarity features, because these features are not discriminative of the different types of text plagiarism cases. Therefore, the proposed system takes into consideration all the different plagiarism types by creating a supervised dataset of 34 sentence similarity features to train SVM classification algorithm. The proposed system is also interested to extract the most effective sentence similarity features that have the ability to differentiate the suspicious cases with the highest accuracy. Therefore, it is depended on filter feature selection approach using Chi-square algorithm to rank the 34 features and extract the most discriminative features of the different types of lexical, syntactic, and semantic text plagiarisms. The proposed system is also based on constructing the hyperplane equation of selected features using SVM classification algorithm, rather than conducting extensive experiments to find the best weighting coefficient values for incorporating the selected features.

The main contribution of this paper can be summarized as:

- Proposing a new plagiarism detection system that deals with all the different types of lexical, syntactic, and semantic plagiarisms.
- Negative Non-plagiarized" and positive "Plagiarized" cases are extracted from the documents of benchmark datasets that have different plagiarism types.
- Thirty-four values to the sentence similarity features of each extracted case are computed and recorded aggregating with the class label to build supervised training database.
- Chi-square algorithm is used to rank the thirty-four features of the created training database and extract the most discriminative features that achieve the highest detect accuracy.
- SVM classification algorithm is used to construct the hyperplane equation of selected features, which has the ability to add new dimensionality to distinguish the overlapping between the training cases of different classes.

The proposed system is implemented through three phases: preprocessing, seeding, and post-processing. In the preprocessing phase, preprocessing techniques such as sentence segmentation, paragraph composition, tokenization, lower casing, removing stop-words, punctuation removal and removed all tokens that did not start with a letter, part of speech tagging, and lemmatization are applied. Secondly, seeding process is utilized to extract the set of possible plagiarized cases, compute the lexical, syntactic, and semantic features, select the most effective features, create the training database, and construct the support vector machine model. It is based on two paths to detect the sentences similarity cases, the first path is based on traditional paragraph-level comparison, and the second path is based on the hyperplane equation of the constructed SVM classifier. The final phase is based on filter seeds, merging adjacent detected seeds, adaptive behavior, and filter segments techniques to extract the best-plagiarized segment between suspicious and source documents.

The performance of the proposed system was evaluated basing on three benchmark datasets from the PAN Workshop series: PAN 2012 [44], PAN 2013 [45], and PAN 2014 [47]. The performance is measured using five standard metrics of PAN series; Recall, Precision, F- measure, Granularity, and Plagdet. The proposed system achieved the highest F- measure and Plagdet scores comparing with the recent related systems on the complete test corpus of PAN 2013 and PAN 2014. The rest of the paper is structured as follows. Section 2 shed light on the previous related work. In Section 3, the details of the proposed system stages will be explained. In Section 4, the details of the experimental setup and the experimental results will be described. This research will be concluded in Section 5.

## 2 Related work

The recent plagiarism detection systems are explained in this section of the paper. The researchers work on different techniques such as n-gram based, semantic-syntactic, etc. most of the following researches consist of four levels to detect plagiarism: preprocessing, seeding, extension and filter. In this section, recent plagiarism detection systems are concluded, it developed to facilitate the comparison between plagiarism detection techniques. In addition, the limitations of each approach is discussed.

Kong et al. [23, 24, 28] presented a technique of plagiarism detection based on vector space model to capture semantic similarity. Mainly is divided into four stages: preprocessing, seeding, extension and filtering. The first stage prepared the documents by special characters elimination, stop words elimination, lower case conversion, and stemming. The second stage was interested in extracting plagiarized sentences and their corresponding source sentences. Therefore, the overlap similarity model is applied in passage level and sentence level. In the next stage, Bilateral Alternating Sorting between the detection pairs was applied for getting large plagiarized passages by merging adjacent pairs. In the final stage, passages whose words overlap under a modified Jaccard coefficient and less than a certain threshold, are eliminated. The main limitation is to find the appropriate thresholds and build an extension algorithm that can adapt with the different types of plagiarism. Rodríguez Torrejón and Martín Ramos [51] designed a model dependent on context n-grams and context skip n-grams. There are four steps in this model. Firstly, basic preprocessing techniques are performed in documents. Secondly, the suspected document is compared with the source document by using context 3-grams and skips context n-grams where n is between 1 and 3. In the third step, simultaneously nearby detections are joined if the gap is less than 4000 characters. Finally, small passages with a length of fewer than 190 characters are discarded. However, the model is not concerned with the semantic side in the analysis.

Shrestha and Solorio [58] proposed an approach that used n-grams having different characteristics, to deal with different plagiarism levels from copy-paste to high-level plagiarism. This approach compares documents upon stop words 8-grams, context 5-grams, and entity 5-grams to extract plagiarized cases. Afterward, adjacent catch plagiarized cases are extended, when the gap between these is less than 8 words. Finally, short passages are filtered out. The shortcoming in this approach is to find a way for mixing n-grams characteristics to improve the overall performance. Palkovskii and Belov [39] introduced a system for plagiarism detection based on context 5-grams for comparison between suspicious and the source documents. Then, the system used clustering by Euclidean distance for the extension of detected seeds. Finally, passes with a length less than 190 characters are deleted. The results of this system show that it failed to detect summary plagiarism types.

Küppers and Conrad [25] proposed an algorithm that based on the following steps: preprocessing text by Tokenization, Stop words removal, collapsing whitespace. The next processing step is to chunk documents with a semi-fixed window size of 250 characters. Then, it compared each suspicious chunk with all the source chunks by using Dice's coefficient. Afterward, the detected plagiarisms cases are joined together if the gap is less than 500 characters. However, the algorithm is failed on highly obfuscated plagiarism.

Sanchez-Perez et al [54] presented an adaptive algorithm consist of 4 phases: preprocessing, seeding, extension and filter. In the first phase, preprocessing natural language techniques such as sentence segmentation and tokenizing, special characters removal, converting all letters to lowercase, and stemming are applied. In the seeding phase, each sentence is represented in the VSM of tf-idf weights. Then, it applied Dice coefficient and cosine measure. In the extension phase, it used clustering and validation between fragments of the source document and corresponding fragments of the suspicious document. In the filter phase, overlapping cases are resolved and small cases with length 150 characters are removed. The main limitation of their method is neglecting the semantic aspects.

Altheneyan et al [3] proposed an automatic plagiarism detection system divided into four steps: paragraph level comparison, sentence-level comparison, SVM classifier, post-processing. In the first step, all documents are converted into paragraphs with length 500 characters, and a comparison between each suspicious paragraph is occurred with all source paragraphs according to unigram and bigram. In the second step, the retrieved pairs of suspicious paragraphs and their corresponding source paragraphs are split into sentences, and are compared using the common unigrams and Meteor score. The SVM classifier step is used to verify the decision using lexical, syntactic, and semantic features. In the post-processing step, the detected pairs with a gap less than 900 characters are merged, and short passages less than 150 characters are discard.

Lovepreet Ahuja et al [2] introduced a plagiarism detection method in which semantic and syntactical knowledge between documents was extracted to detect the plagiarized segment from the text. This approach was applied in three phases: preprocessing, detailed analysis, filtering. In the preprocessing phase, Text segmentation, stop words removal, and lemmatization processes are used to prepare both suspicious and source documents. In the detailed comparison phase, a unique joint vector was formed by using suspected sentences and source sentences. Then, syntactic and semantic scores are calculated upon assigned different weights to linguistic characteristics: inverse path length characteristic and depth estimation characteristic. Afterward, the overall similarity is computed by combine syntactical and semantic scores with various weights. In the filtering phase, the non-plagiarized sentences that do not meet the conditions are removed. The system unable to detect complicated instances of plagiarism and determine the method that provides the best weights. Table 1 shows a summary of the stages that used to detect plagiarism in related researches.

Chia-Yang et al [7] concerned in plagiarism detection for documents retrieval and text alignment, it is focused on embedding Word2Vec word. After that, the embedding is grouped into semantic concepts, which are represented at several granularity levels. Word-2vec is used to convert words into word vectors that contain semantic relations between the words. Spherical K-means is also used to cluster the words into semantic concepts. This method has major limitations, the terms used are single words, and the semantic meaning of the sentences remains unknown. Faisal Alvi et al [5] is proposed an approach to identify two important types of obfuscation: changes in sentence structure and synonym substitution. This is accomplished by proposing a three-step methodology: pre-processing, identification of word reordering, and identification of synonymous substitutions. In these steps, permutations of identical textual segments and paraphrase patterns of the reordered words have been used. In addition, the embedding word of ConceptNet Numberbatch and the Smith-Waterman algorithm are used to detect synonym substitution. The shortcoming of this approach is that it does not deal with the other types of obfuscation, such as deletion and addition.

Bilal Ghanem et al [12] introduced a hybrid arabic plagiarism detection system, it is called (HYPLAG) that deals with all different types of plagiarism such as

**Table 1** Summary of the stages that used to detect plagiarism in related researches

| Author | Preprocessing | Seeding | Post-processing | | Limitation |
|---|---|---|---|---|---|
| | | | extension | filter | |
| Kong et al. [23, 24, 28] | Special characters elimination, stop words elimination, lower Case conversion, stemming | Overlapping measure model | Bilateral alternating sorting | Passage similarity | -Appropriate threshold values<br>-The extension algorithm that can adapt with the different types of plagiarism |
| Rodríguez et al [51] | Stop words removal, lower Case conversion, stemming | -Context n-grams<br>-Context skip n-grams | Distance between seeds < 4000 characters | Small passages removal < 190 characters | Neglects the semantic side in the text analysis. |
| Shrestha et al [58] | Lower case conversion | -Context n-grams<br>-Stop word n-grams<br>-Named entity n-grams | -Distance between seeds < 8 words<br>-Extension with multiple features | -Overlapping removal<br>-Small passages removal < threshold | Finding a way for mixing n-grams characteristics. |
| Palkovskii and Belov [39] | Numbers removal, lower Case conversion, stemming | Context n-grams | Clusters Euclidean distance | Small passages removal < 190 characters | Fails to detect summary plagiarism types. |
| Küppers and Conrad [25] | Tokenization, stop words removal, whitespace removal. | Dice coefficient with 250 character chunks | Distance between seeds < 500 characters | - | Fails on highly obfuscated plagiarism. |
| Sanchez-Perez et al. [54] | Special characters removal, lower case conversion, stemming | -Dice coefficient and cosine measure with<br>-Vectors of tf-idf weights | Clustering and validation | -Overlapping removal<br>-Passage similarity<br>-Small passages removal <150 | Determining optimal parameter values. |
| Altheneyan et al. [3] | Stop words removal, punctuation removal, tokenization | -Context n-grams<br>-Meteor score<br>-Support vector machine | Distance between seeds < 900 characters | Small passages removal < 150 characters | Exploring Merging heuristics. |
| Lovepreet Ahuja et al. [2] | Special characters removal, whitespace removal, sentence segmentation, stop words removal, lemmatization | Combined semantic and syntactic scores | - | Overall similarity < 0.65 | Unable to detect complicated instances of plagiarism and determine the method that provides the best weights. |

**Table 1** (continued)

| Author | Preprocessing | Seeding | Post-processing | | Limitation |
|---|---|---|---|---|---|
| | | | extension | filter | |
| Asif Ekbal et al. [11] | sentence segmentation, tokenization, finding part of speech classes, lemmatization, character-offsets and named-entity (NE) classes | Cosine measure with vectors of tf-idf weights | Recursively attempts to merge matches if they satisfy the specific heuristics | Small passages removal < 100 characters | -Semantic aspect is not considered in the analysis.<br>-Fails to detect cases of the high obfuscation. |
| Faisal Alvi et al. [5] | Punctuation removal, case folding and sentence filtering | N-grams | - | - | Deals only with paraphrase types: synonymous substitution and word reordering. |
| Bilal Ghanem et al. [12] | Remove (diacritics, non-Arabic letters, all words that contain numbers and words that are composed of one letter), named entities, stemming, POS tagging | Graph-based technique and depth-first search algorithm | - | - | Selecting threshold values |
| Chia Yang et al. [7] | Lower case conversion, | Permutations of identical text segments | - | - | High rate of false positives sensitivity of each parameter. |

copy-and-paste, paraphrasing, and synonym substitution. The system is architecture-based on Arabic WordNet in order to extract all the synonyms. HYPLAG consists of three steps: sentence ranking, Td-Idf terms weighting, and feature based semantic similarity. In the first process, sentence ranking is based on the structure of the search engine. In the second process, the similarity measure is calculated based on the Td-Idf weights for retrieved sentences with cosine similarity. The feature based semantic similarity process is based on the calculation of Tversky sentences measurement. The main limitation is the selection of the threshold values. Asif Ekbal et al [11] proposed an approach for external plagiarism detection. This approach consists of four stages: pre-processing, subset selection, passage selection, and filtering of false detections. In the sub-selection stage, the document retrieval task is achieved by using a vector space model (VSM). This model is based on converting all documents into vectors, where each cell in the vector is assigned by term-frequency and inverse document frequency (TF-IDF). Then, the similarity measure between vectors is calculated by cosine similarity. Their approach has many limitations such as the semantic aspect is not considered in the analysis and the failure to detect cases of high obfuscation.

## 3 Proposed system

The proposed system aims to detect the text plagiarism with the highest possible accuracy. Therefore, it takes into consideration all the different types of lexical, syntactic and semantic similarity features. All previous researches of the text plagiarism depended on 1, 2, 3 or 4 sentence similarity features for detecting the text similarity [2, 3, 23–25, 28, 39, 51, 54, 58], they also carried out extensive experiments to find the best weighting coefficient values for incorporating their selected similarity features. On the other side, the proposed system is based on 34 features that reflect all the different types of the text similarity. Increasing the number of text similarity features, makes the proposed method more robust in differentiating the confusion similarities that have a difficulty to detect their plagiarism, and detecting the different variations of text plagiarism with more accuracy. The proposed system also takes into consideration the determination of most effective features that have the ability to discriminate the suspicious cases with the highest accuracy. Therefore, it is depended on filter feature selection approach using Chi-square algorithm to rank the 34 features and extract the most discriminative features for the different types of lexical, syntactic and semantic text plagiarisms. The proposed system is also depended on constructing the hyperplane equation of 34 features using SVM classification algorithm, rather than conducting extensive experiments to find the best weighting coefficient values for incorporating the 34 features.

The proposed system is constructed basing on three main phases: preprocessing, seeding and post-processing. The general workflow of the proposed system is described in Fig. 1. The first phase is used to preprocess the suspicious and source documents. The second phase "Seeding" is depended on two paths to detect the sentences similarity, the first path is based on traditional paragraph-level comparison, and the second path is based on the hyperplane equation of the constructed SVM classifier. The third phase "Post-processing" is used to extract the best-plagiarized segment between the suspicious and source documents.

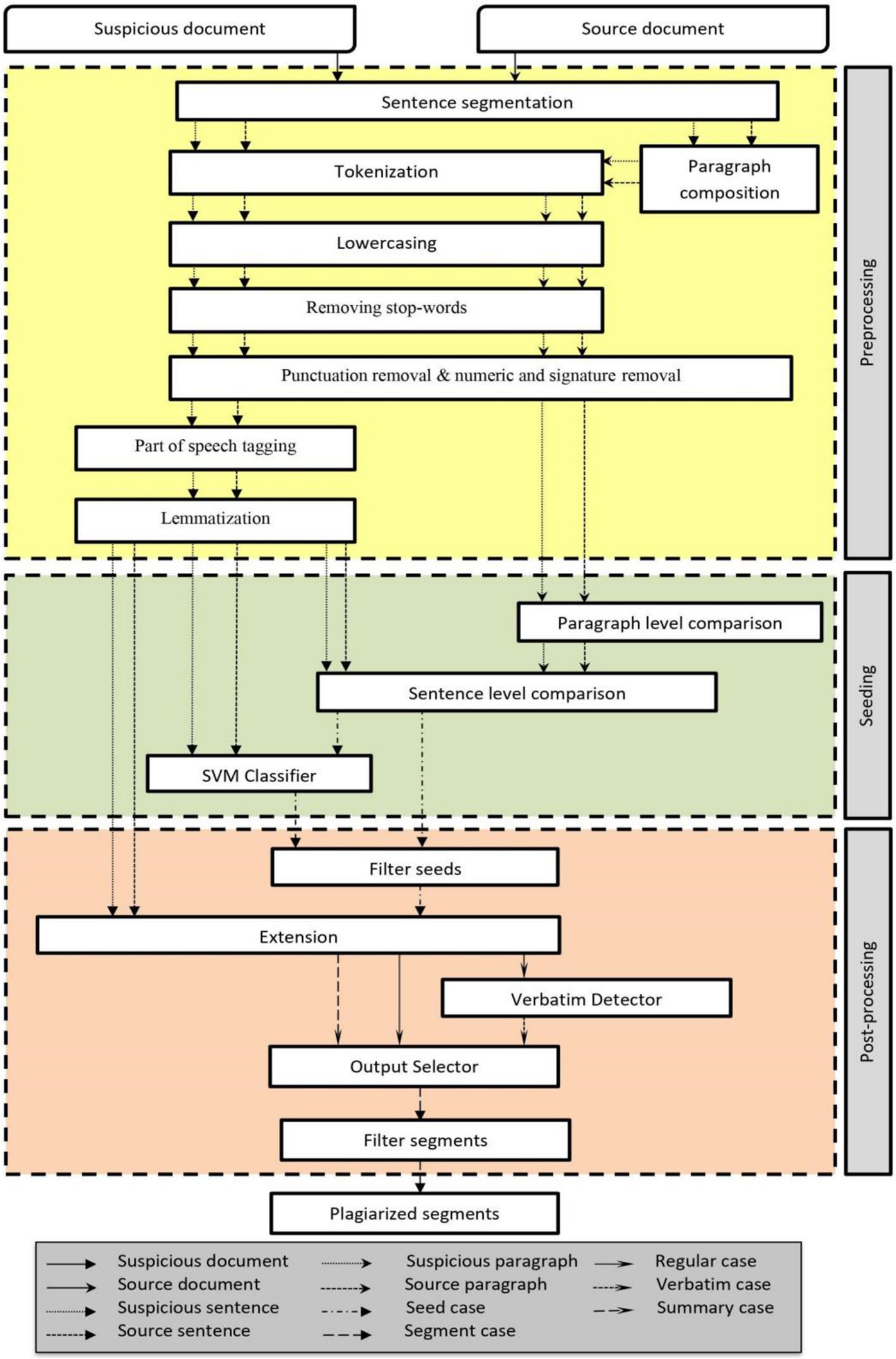


**Fig. 1** General workflow of the proposed system

### 3.1 Preprocessing

The inputs of proposed system are suspicious and source documents. The proposed system analyzes these inputs through three phases as shown in Fig. 1 to detect the text plagiarism. The first phase of the proposed system is used to preprocess the suspicious and source documents, it depends on several processes: sentence segmentation, paragraph composition, tokenization, lowercasing, removing all tokens that do not start with a letter, stop words removal, punctuation removal, part of speech tagging, remain valuable class (noun, verb, adjective, adverb) and lemmatization.

- Sentence segmentation: it splits the document into sentences by using sentence delimiters symbols such as ".", "?" and "!".
- Paragraph composition: each document is paraphrased into the form of paragraphs. In the English language, the average of paragraph length is 100 words [32] and the average of word length is 5.1 characters [6]. Therefore, 500 characters for each paragraph length is chosen, and the adjacent sentences are grouped until the required length of characters "500" is achieved. During the clustering process of the sentences to paragraphs, each sentence is grouped with all of its words without cutting to preserve the context of the sentences. Therefore, each extracted paragraph will contain number of characters around 500.
- Tokenization: it divides the text into smaller parts called tokens using unigrams and bigrams techniques.
- Lowercasing: it converts all the tokens into a lowercasing form.
- Removing stop-words: there are common terms that appearing in the text, they comprise around 40% to 50% of the words in plain text [62]. Therefore, these words will be removed from the text to reduce the computation time of the proposed system and improve its efficiency and accuracy. The proposed system removes the stop words basing on the NLTK stop word list, it contains around 160 stop-words including is, i, am, are, will, we, my, etc.
- Punctuation and all tokens that do not start with a letter are also removed.
- Part of speech tagging: it gives each word in the text its equivalent word-class basing on its definition and context. The word-classes contains noun, verb, adjective, adverb, conjunction, preposition, articles, pronouns, prepositions and determiners class labels. The proposed system eliminates all the classes except noun, verb, adjective and adverb classes, because these classes have a significant role in the semantics of a sentence.
- Lemmatization: it is an operation of converting words into a dictionary base form.

### 3.2 Seeding

Each case of the text plagiarism is fragments of the source and suspicious documents, which are consistent in the context and meaning. The goal of the proposed system is to detect the text plagiarism cases with the highest possible accuracy. Therefore, it doesn't only depend on the traditional techniques for the sentences comparison, but it is also based on the artificial intelligent approaches to achieve the desired goal. One of the recommended important phases of the proposed system is the seeding phase, it aims to extract a set of the possible plagiarized cases calling seeds. The seeding phase is based on two paths to detect the sentences similarity cases, the first path is based on traditional paragraph-level comparison using two levels; paragraph-level comparison and sentence-level comparison. The second path is based on the hyperplane equation of the constructed SVM classifier.

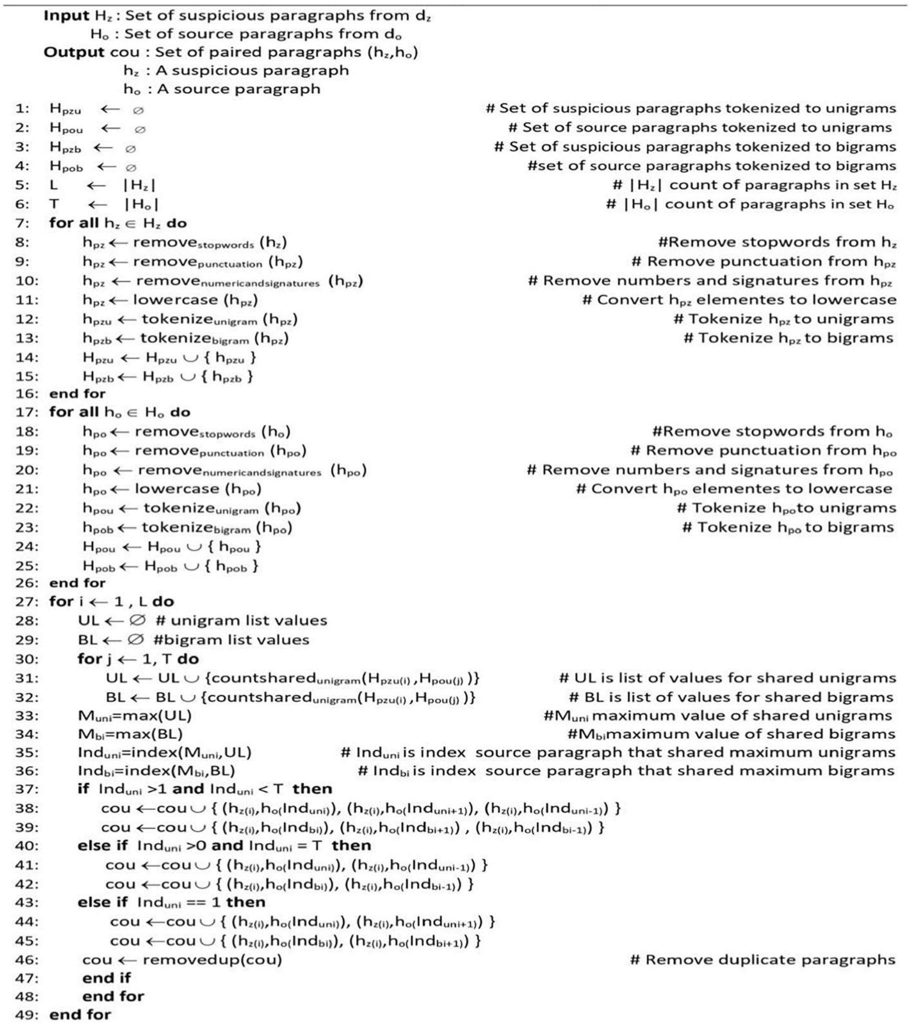
### 3.2.1 First path of the seeding phase

The first path of the seeding phase is based on traditional paragraph-level to detect the plagiarized cases, it consists of two levels; paragraph-level comparison and sentence-level comparison.

**Paragraph-level comparison:** this level aims to extract the most similar paragraphs between the suspicious and source documents. It compares each paragraph of the suspicious document with all the paragraphs of the source document based on n-gram

```
Algorithm 1: Paragraph-level comparison
Input $H_z$ : Set of suspicious paragraphs from $d_z$
      $H_o$ : Set of source paragraphs from $d_o$
Output cou : Set of paired paragraphs $(h_z,h_o)$
      $h_z$ : A suspicious paragraph
      $h_o$ : A source paragraph
$H_{pzu} \leftarrow \emptyset$                    # Set of suspicious paragraphs tokenized to unigrams
$H_{pou} \leftarrow \emptyset$                    # Set of source paragraphs tokenized to unigrams
$H_{pzb} \leftarrow \emptyset$                    # Set of suspicious paragraphs tokenized to bigrams
$H_{pob} \leftarrow \emptyset$                    #set of source paragraphs tokenized to bigrams
$L \leftarrow |H_z|$                              # $|H_z|$ count of paragraphs in set $H_z$
$T \leftarrow |H_o|$                              # $|H_o|$ count of paragraphs in set $H_o$
for all $h_z \in H_z$ do
    $h_{pz} \leftarrow remove_{stopwords}(h_z)$                        #Remove stopwords from $h_z$
    $h_{pz} \leftarrow remove_{punctuation}(h_{pz})$                  # Remove punctuation from $h_{pz}$
    $h_{pz} \leftarrow remove_{numericandsignatures}(h_{pz})$         # Remove numbers and signatures from $h_{pz}$
    $h_{pz} \leftarrow lowercase(h_{pz})$                              # Convert $h_{pz}$ elementes to lowercase
    $h_{pzu} \leftarrow tokenize_{unigram}(h_{pz})$                   # Tokenize $h_{pz}$ to unigrams
    $h_{pzb} \leftarrow tokenize_{bigram}(h_{pz})$                    # Tokenize $h_{pz}$ to bigrams
    $H_{pzu} \leftarrow H_{pzu} \cup \{h_{pzu}\}$
    $H_{pzb} \leftarrow H_{pzb} \cup \{h_{pzb}\}$
end for
for all $h_o \in H_o$ do
    $h_{po} \leftarrow remove_{stopwords}(h_o)$                        #Remove stopwords from $h_o$
    $h_{po} \leftarrow remove_{punctuation}(h_{po})$                  # Remove punctuation from $h_{po}$
    $h_{po} \leftarrow remove_{numericandsignatures}(h_{po})$         # Remove numbers and signatures from $h_{po}$
    $h_{po} \leftarrow lowercase(h_{po})$                              # Convert $h_{po}$ elementes to lowercase
    $h_{pou} \leftarrow tokenize_{unigram}(h_{po})$                   # Tokenize $h_{po}$ to unigrams
    $h_{pob} \leftarrow tokenize_{bigram}(h_{po})$                    # Tokenize $h_{po}$ to bigrams
    $H_{pou} \leftarrow H_{pou} \cup \{h_{pou}\}$
    $H_{pob} \leftarrow H_{pob} \cup \{h_{pob}\}$
end for
for $i \leftarrow 1, L$ do
    $UL \leftarrow \emptyset$ # unigram list values
    $BL \leftarrow \emptyset$ #bigram list values
    for $j \leftarrow 1, T$ do
        $UL \leftarrow UL \cup \{countshared_{unigram}(H_{pzu(i)}, H_{pou(j)})\}$     # UL is list of values for shared unigrams
        $BL \leftarrow BL \cup \{countshared_{unigram}(H_{pzu(i)}, H_{pou(j)})\}$     # BL is list of values for shared bigrams
    $M_{uni}=max(UL)$                                  #$M_{uni}$ maximum value of shared unigrams
    $M_{bi}=max(BL)$                                   #$M_{bi}$maximum value of shared bigrams
    $Ind_{uni}=index(M_{uni},UL)$        # $Ind_{uni}$ is index source paragraph that shared maximum unigrams
    $Ind_{bi}=index(M_{bi},BL)$          # $Ind_{bi}$ is index source paragraph that shared maximum bigrams
    if $Ind_{uni} > 1$ and $Ind_{uni} < T$ then
        $cou \leftarrow cou \cup \{ (h_{z(i)},h_{o(Ind_{uni})}), (h_{z(i)},h_{o(Ind_{uni+1})}), (h_{z(i)},h_{o(Ind_{uni-1})}) \}$
        $cou \leftarrow cou \cup \{ (h_{z(i)},h_{o(Ind_{bi})}), (h_{z(i)},h_{o(Ind_{bi+1})}), (h_{z(i)},h_{o(Ind_{bi-1})}) \}$
    else if $Ind_{uni} > 0$ and $Ind_{uni} = T$ then
        $cou \leftarrow cou \cup \{ (h_{z(i)},h_{o(Ind_{uni})}), (h_{z(i)},h_{o(Ind_{uni-1})}) \}$
        $cou \leftarrow cou \cup \{ (h_{z(i)},h_{o(Ind_{bi})}), (h_{z(i)},h_{o(Ind_{bi-1})}) \}$
    else if $Ind_{uni} == 1$ then
        $cou \leftarrow cou \cup \{ (h_{z(i)},h_{o(Ind_{uni})}), (h_{z(i)},h_{o(Ind_{uni+1})}) \}$
        $cou \leftarrow cou \cup \{ (h_{z(i)},h_{o(Ind_{bi})}), (h_{z(i)},h_{o(Ind_{bi+1})}) \}$
    $cou \leftarrow removedup(cou)$                    # Remove duplicate paragraphs
    end if
    end for
end for
```

**Fig. 2** Pseudo code of algorithm 1 for the paragraph-level comparison

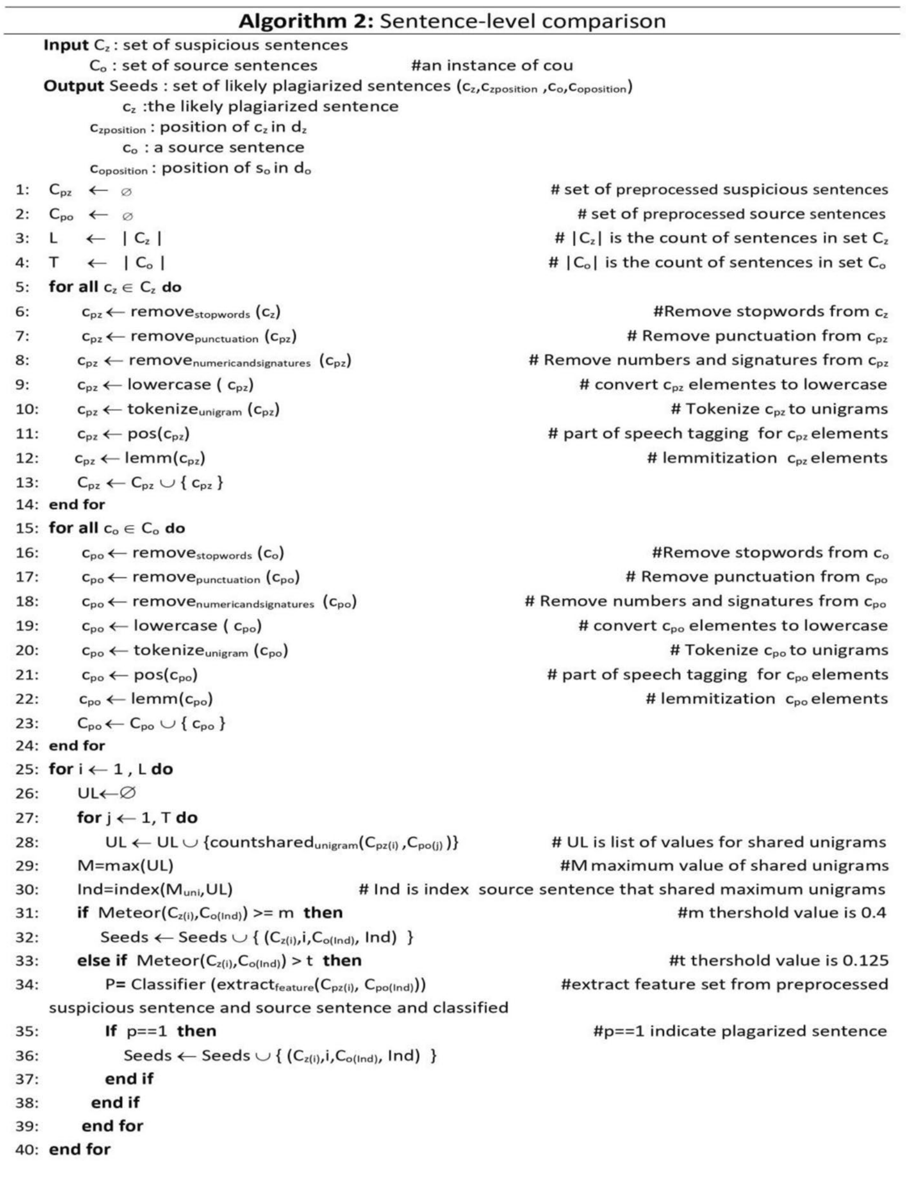


```
Input $H_z$ : Set of suspicious paragraphs from $d_z$
      $H_o$ : Set of source paragraphs from $d_o$
Output cou : Set of paired paragraphs $(h_z,h_o)$
      $h_z$ : A suspicious paragraph
      $h_o$ : A source paragraph
$H_{pzu} \leftarrow \varnothing$        # Set of suspicious paragraphs tokenized to unigrams
$H_{pou} \leftarrow \varnothing$        # Set of source paragraphs tokenized to unigrams
$H_{pzb} \leftarrow \varnothing$        # Set of suspicious paragraphs tokenized to bigrams
$H_{pob} \leftarrow \varnothing$        #set of source paragraphs tokenized to bigrams
$L \leftarrow |H_z|$        # $|H_z|$ count of paragraphs in set $H_z$
$T \leftarrow |H_o|$        # $|H_o|$ count of paragraphs in set $H_o$
for all $h_z \in H_z$ do
    $h_{pz} \leftarrow remove_{stopwords}(h_z)$        #Remove stopwords from $h_z$
    $h_{pz} \leftarrow remove_{punctuation}(h_{pz})$        # Remove punctuation from $h_{pz}$
    $h_{pz} \leftarrow remove_{numericandsignatures}(h_{pz})$        # Remove numbers and signatures from $h_{pz}$
    $h_{pz} \leftarrow lowercase(h_{pz})$        # Convert $h_{pz}$ elementes to lowercase
    $h_{pzu} \leftarrow tokenize_{unigram}(h_{pz})$        # Tokenize $h_{pz}$ to unigrams
    $h_{pzb} \leftarrow tokenize_{bigram}(h_{pz})$        # Tokenize $h_{pz}$ to bigrams
    $H_{pzu} \leftarrow H_{pzu} \cup \{h_{pzu}\}$
    $H_{pzb} \leftarrow H_{pzb} \cup \{h_{pzb}\}$
end for
for all $h_o \in H_o$ do
    $h_{po} \leftarrow remove_{stopwords}(h_o)$        #Remove stopwords from $h_o$
    $h_{po} \leftarrow remove_{punctuation}(h_{po})$        # Remove punctuation from $h_{po}$
    $h_{po} \leftarrow remove_{numericandsignatures}(h_{po})$        # Remove numbers and signatures from $h_{po}$
    $h_{po} \leftarrow lowercase(h_{po})$        # Convert $h_{po}$ elementes to lowercase
    $h_{pou} \leftarrow tokenize_{unigram}(h_{po})$        # Tokenize $h_{po}$to unigrams
    $h_{pob} \leftarrow tokenize_{bigram}(h_{po})$        # Tokenize $h_{po}$ to bigrams
    $H_{pou} \leftarrow H_{pou} \cup \{h_{pou}\}$
    $H_{pob} \leftarrow H_{pob} \cup \{h_{pob}\}$
end for
for $i \leftarrow 1, L$ do
    $UL \leftarrow \varnothing$ # unigram list values
    $BL \leftarrow \varnothing$ #bigram list values
    for $j \leftarrow 1, T$ do
        $UL \leftarrow UL \cup \{countshared_{unigram}(H_{pzu(i)}, H_{pou(j)})\}$        # UL is list of values for shared unigrams
        $BL \leftarrow BL \cup \{countshared_{unigram}(H_{pzu(i)}, H_{pou(j)})\}$        # BL is list of values for shared bigrams
    $M_{uni}=max(UL)$        #$M_{uni}$ maximum value of shared unigrams
    $M_{bi}=max(BL)$        #$M_{bi}$maximum value of shared bigrams
    $Ind_{uni}=index(M_{uni},UL)$        # $Ind_{uni}$ is index source paragraph that shared maximum unigrams
    $Ind_{bi}=index(M_{bi},BL)$        # $Ind_{bi}$ is index source paragraph that shared maximum bigrams
    if $Ind_{uni} > 1$ and $Ind_{uni} < T$ then
        $cou \leftarrow cou \cup \{(h_{z(i)},h_{o(Ind_{uni})}), (h_{z(i)},h_{o(Ind_{uni+1})}), (h_{z(i)},h_{o(Ind_{uni-1})})\}$
        $cou \leftarrow cou \cup \{(h_{z(i)},h_{o(Ind_{bi})}), (h_{z(i)},h_{o(Ind_{bi+1})}), (h_{z(i)},h_{o(Ind_{bi-1})})\}$
    else if $Ind_{uni} > 0$ and $Ind_{uni} = T$ then
        $cou \leftarrow cou \cup \{(h_{z(i)},h_{o(Ind_{uni})}), (h_{z(i)},h_{o(Ind_{uni-1})})\}$
        $cou \leftarrow cou \cup \{(h_{z(i)},h_{o(Ind_{bi})}), (h_{z(i)},h_{o(Ind_{bi-1})})\}$
    else if $Ind_{uni} == 1$ then
        $cou \leftarrow cou \cup \{(h_{z(i)},h_{o(Ind_{uni})}), (h_{z(i)},h_{o(Ind_{uni+1})})\}$
        $cou \leftarrow cou \cup \{(h_{z(i)},h_{o(Ind_{bi})}), (h_{z(i)},h_{o(Ind_{bi+1})})\}$
    $cou \leftarrow removedup(cou)$        # Remove duplicate paragraphs
    end if
    end for
end for
```

**Algorithm 1** Paragraph-level comparison

approach where n=1 and 2 [31]. The operational work of this step is described in algorithm 1 as shown in Fig. 2. It consists of three processes. Firstly, the common unigram between each suspicious paragraph $h_z$ and all source paragraphs are computed. Then, the $N$ source paragraphs that have the maximum value of common unigrams are selected, the previous and next paragraphs for the selected paragraph are also chosen. Additionally, the shared bigrams between each suspicious paragraph $h_z$ and all source paragraphs are calculated. Then, the $N$ source paragraphs that have the maximum value of shared bigrams are selected, the previous and next paragraphs for the selected paragraph are also chosen. Finally, in the case of selection the same source paragraph in unigram and bigram, unique source paragraph is selected.

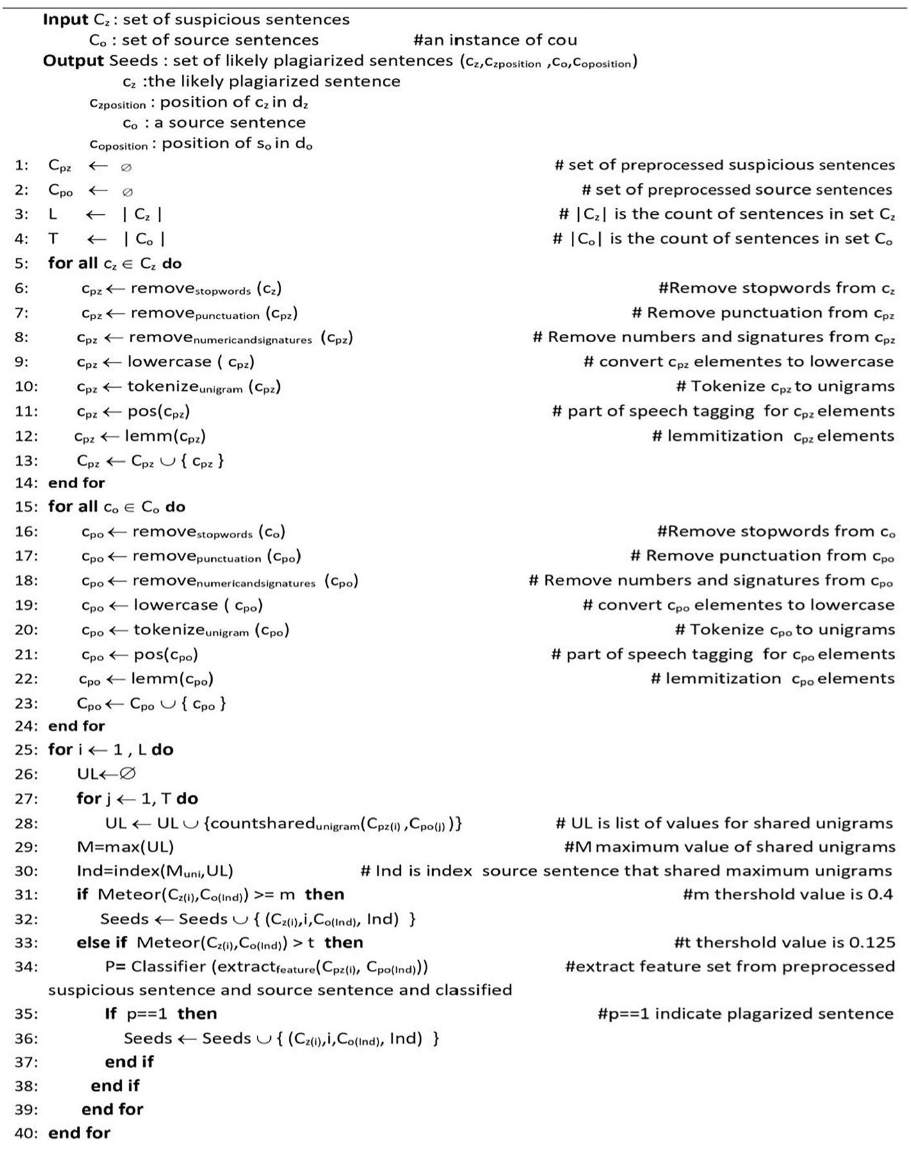
**Sentence-level comparison:** after extracting the suspicious paragraphs $h_z$ and their paired source paragraphs in the previous level, the preprocessed steps that explained in



```
Algorithm 2: Sentence-level comparison
Input $C_z$ : set of suspicious sentences
      $C_o$ : set of source sentences          #an instance of cou
Output Seeds : set of likely plagiarized sentences ($c_z, c_{zposition}, c_o, c_{oposition}$)
      $c_z$ :the likely plagiarized sentence
      $c_{zposition}$ : position of $c_z$ in $d_z$
      $c_o$ : a source sentence
      $c_{oposition}$ : position of $s_o$ in $d_o$
1:  $C_{pz} \leftarrow \varnothing$                                   # set of preprocessed suspicious sentences
2:  $C_{po} \leftarrow \varnothing$                                   # set of preprocessed source sentences
3:  $L \leftarrow |C_z|$                                  # $|C_z|$ is the count of sentences in set $C_z$
4:  $T \leftarrow |C_o|$                                  # $|C_o|$ is the count of sentences in set $C_o$
5:  for all $c_z \in C_z$ do
6:      $c_{pz} \leftarrow remove_{stopwords}(c_z)$                         #Remove stopwords from $c_z$
7:      $c_{pz} \leftarrow remove_{punctuation}(c_{pz})$                      # Remove punctuation from $c_{pz}$
8:      $c_{pz} \leftarrow remove_{numericandsignatures}(c_{pz})$    # Remove numbers and signatures from $c_{pz}$
9:      $c_{pz} \leftarrow lowercase(c_{pz})$                        # convert $c_{pz}$ elementes to lowercase
10:     $c_{pz} \leftarrow tokenize_{unigram}(c_{pz})$                        # Tokenize $c_{pz}$ to unigrams
11:     $c_{pz} \leftarrow pos(c_{pz})$                          # part of speech tagging  for $c_{pz}$ elements
12:     $c_{pz} \leftarrow lemm(c_{pz})$                                     # lemmitization  $c_{pz}$ elements
13:     $C_{pz} \leftarrow C_{pz} \cup \{c_{pz}\}$
14: end for
15: for all $c_o \in C_o$ do
16:     $c_{po} \leftarrow remove_{stopwords}(c_o)$                         #Remove stopwords from $c_o$
17:     $c_{po} \leftarrow remove_{punctuation}(c_{po})$                      # Remove punctuation from $c_{po}$
18:     $c_{po} \leftarrow remove_{numericandsignatures}(c_{po})$    # Remove numbers and signatures from $c_{po}$
19:     $c_{po} \leftarrow lowercase(c_{po})$                        # convert $c_{po}$ elementes to lowercase
20:     $c_{po} \leftarrow tokenize_{unigram}(c_{po})$                        # Tokenize $c_{po}$ to unigrams
21:     $c_{po} \leftarrow pos(c_{po})$                          # part of speech tagging  for $c_{po}$ elements
22:     $c_{po} \leftarrow lemm(c_{po})$                                     # lemmitization  $c_{po}$ elements
23:     $C_{po} \leftarrow C_{po} \cup \{c_{po}\}$
24: end for
25: for $i \leftarrow 1, L$ do
26:     $UL \leftarrow \varnothing$
27:     for $j \leftarrow 1, T$ do
28:         $UL \leftarrow UL \cup \{countshared_{unigram}(C_{pz(i)}, C_{po(j)})\}$   # UL is list of values for shared unigrams
29:     M=max(UL)                                          #M maximum value of shared unigrams
30:     Ind=index($M_{uni}$,UL)          # Ind is index  source sentence that shared maximum unigrams
31:     if Meteor($C_{z(i)}, C_{o(Ind)}$) >= m  then                      #m thershold value is 0.4
32:         Seeds $\leftarrow$ Seeds $\cup$ { ($C_{z(i)}$,i,$C_{o(Ind)}$, Ind) }
33:     else if  Meteor($C_{z(i)}, C_{o(Ind)}$) > t  then                  #t thershold value is 0.125
34:         P= Classifier (extract$_{feature}$($C_{pz(i)}$, $C_{po(Ind)}$))     #extract feature set from preprocessed
    suspicious sentence and source sentence and classified
35:         If  p==1  then                                 #p==1 indicate plagarized sentence
36:             Seeds $\leftarrow$ Seeds $\cup$ { ($C_{z(i)}$,i,$C_{o(Ind)}$, Ind) }
37:         end if
38:     end if
39:     end for
40: end for
```

**Fig. 3** Pseudo code of algorithm 2 for the sentence-level comparison

Section 3.1 are applied to these paragraphs. Subsequently, each suspicious sentence $C_z$ is compared with all the source sentences. Then, the source sentence will be extracted, if it has the maximum value of the common unigrams comparing with $C_z$. The operational work of this step is described in algorithm 2 as shown in Fig. 3. There are three different decisions for comparing the pair of sentences. These decisions are based on two threshold $m$ and $t$ comparing to the value of Meteor score between the pair of sentences. If a value exceeds or is equal to a threshold $m$, it is considered as "Plagiarized case". Else if a Meteor



```
Input $C_z$ : set of suspicious sentences
      $C_o$ : set of source sentences          #an instance of cou
Output Seeds : set of likely plagiarized sentences ($c_z, c_{zposition}, c_o, c_{oposition}$)
      $c_z$ :the likely plagiarized sentence
      $c_{zposition}$ : position of $c_z$ in $d_z$
      $c_o$ : a source sentence
      $c_{oposition}$ : position of $s_o$ in $d_o$
1:  $C_{pz} \leftarrow \varnothing$                            # set of preprocessed suspicious sentences
2:  $C_{po} \leftarrow \varnothing$                            # set of preprocessed source sentences
3:  $L \leftarrow |C_z|$                         # $|C_z|$ is the count of sentences in set $C_z$
4:  $T \leftarrow |C_o|$                         # $|C_o|$ is the count of sentences in set $C_o$
5:  for all $c_z \in C_z$ do
6:      $c_{pz} \leftarrow remove_{stopwords}(c_z)$                  #Remove stopwords from $c_z$
7:      $c_{pz} \leftarrow remove_{punctuation}(c_{pz})$              # Remove punctuation from $c_{pz}$
8:      $c_{pz} \leftarrow remove_{numericandsignatures}(c_{pz})$     # Remove numbers and signatures from $c_{pz}$
9:      $c_{pz} \leftarrow lowercase(c_{pz})$                      # convert $c_{pz}$ elementes to lowercase
10:     $c_{pz} \leftarrow tokenize_{unigram}(c_{pz})$                # Tokenize $c_{pz}$ to unigrams
11:     $c_{pz} \leftarrow pos(c_{pz})$                           # part of speech tagging for $c_{pz}$ elements
12:     $c_{pz} \leftarrow lemm(c_{pz})$                          # lemmitization $c_{pz}$ elements
13:     $C_{pz} \leftarrow C_{pz} \cup \{c_{pz}\}$
14: end for
15: for all $c_o \in C_o$ do
16:     $c_{po} \leftarrow remove_{stopwords}(c_o)$                  #Remove stopwords from $c_o$
17:     $c_{po} \leftarrow remove_{punctuation}(c_{po})$              # Remove punctuation from $c_{po}$
18:     $c_{po} \leftarrow remove_{numericandsignatures}(c_{po})$     # Remove numbers and signatures from $c_{po}$
19:     $c_{po} \leftarrow lowercase(c_{po})$                      # convert $c_{po}$ elementes to lowercase
20:     $c_{po} \leftarrow tokenize_{unigram}(c_{po})$                # Tokenize $c_{po}$ to unigrams
21:     $c_{po} \leftarrow pos(c_{po})$                           # part of speech tagging for $c_{po}$ elements
22:     $c_{po} \leftarrow lemm(c_{po})$                          # lemmitization $c_{po}$ elements
23:     $C_{po} \leftarrow C_{po} \cup \{c_{po}\}$
24: end for
25: for $i \leftarrow 1, L$ do
26:     $UL \leftarrow \varnothing$
27:     for $j \leftarrow 1, T$ do
28:         $UL \leftarrow UL \cup \{countshared_{unigram}(C_{pz(i)}, C_{po(j)})\}$   # UL is list of values for shared unigrams
29:     M=max(UL)                                   #M maximum value of shared unigrams
30:     Ind=index($M_{uni}$,UL)       # Ind is index source sentence that shared maximum unigrams
31:     if Meteor($C_{z(i)}, C_{o(Ind)}$) >= m then                  #m thershold value is 0.4
32:         Seeds $\leftarrow$ Seeds $\cup$ { ($C_{z(i)}$, i, $C_{o(Ind)}$, Ind) }
33:     else if Meteor($C_{z(i)}, C_{o(Ind)}$) > t then               #t thershold value is 0.125
34:         P= Classifier (extract$_{feature}$($C_{pz(i)}$, $C_{po(Ind)}$))     #extract feature set from preprocessed
    suspicious sentence and source sentence and classified
35:         If p==1 then                            #p==1 indicate plagarized sentence
36:             Seeds $\leftarrow$ Seeds $\cup$ { ($C_{z(i)}$, i, $C_{o(Ind)}$, Ind) }
37:         end if
38:     end if
39:     end for
40: end for
```

**Algorithm 2** Sentence-level comparison

score value is less than a threshold *t*, it is considered a "Non-plagiarized case". Otherwise, the pair of sentences will be analyzed using SVM classifier.

### 3.2.2 Second path of the seeding phase

Seeding phase of the proposed system depends on two paths to detect the sentences similarity. The first path is based on traditional paragraph-level comparison as described in the previous section, and the second path is based on SVM classifier as shown in Fig. 1. The second path of seeding phase is used, if the first path "Paragraph-level comparison" didn't able to discover the text similarity, it is based on constructing SVM classifier that has the ability to detect all the different types of lexical, syntactic, and semantic plagiarism cases. The construction process of SVM classifier is described in Fig. 4, it consists of four stages; negative and positive instances extraction, features computation, features selection, and classifier construction. In the first stage, negative "Non-plagiarized" and positive "Plagiarized" cases are extracted from the training documents to build a supervised training database. In the second stage, 34 values of the sentences similarity features are computed and recorded aggregating with the class label for each extracted case of the first stage, which can reflect all the different types of lexical, syntactic and semantic text similarities.

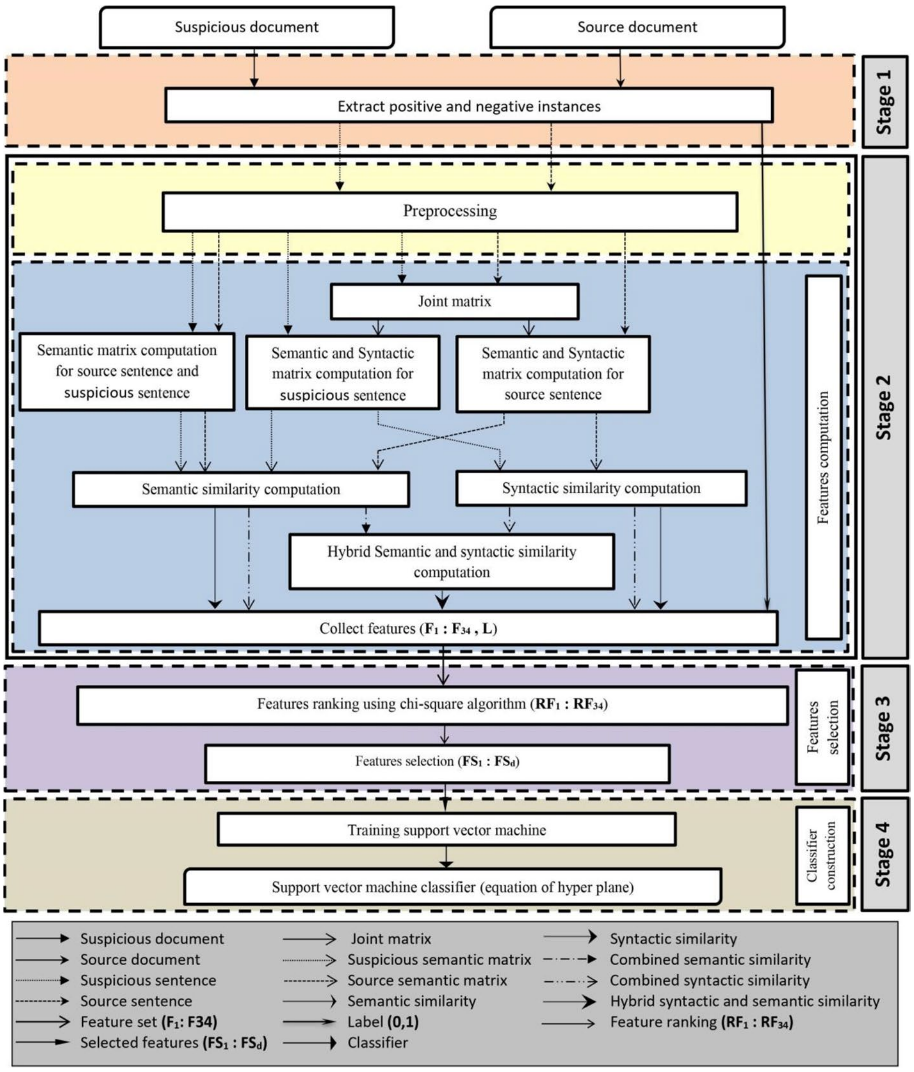
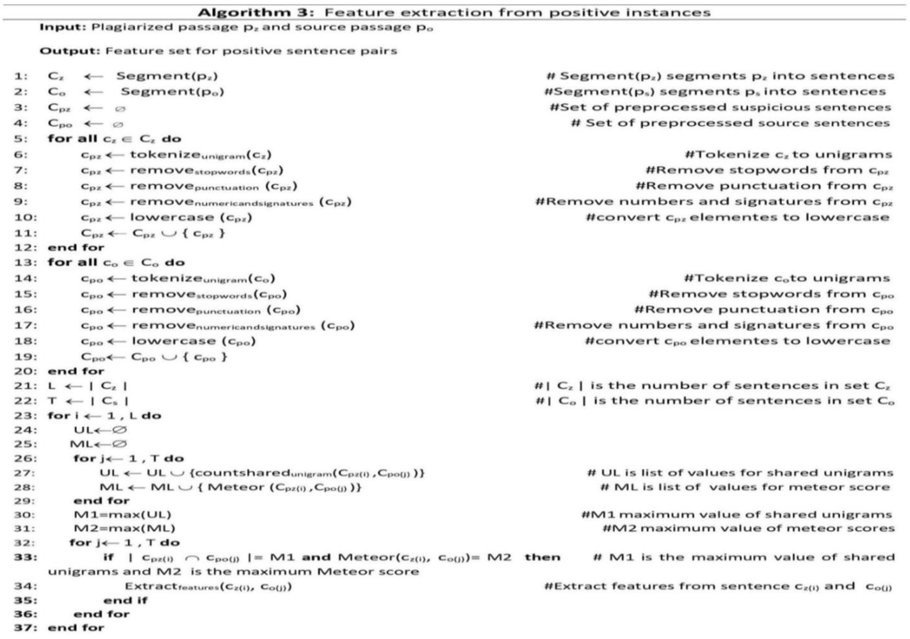

Suspicious document
Source document
Stage 1
Extract positive and negative instances
Preprocessing
Joint matrix
Semantic matrix computation for source sentence and suspicious sentence
Semantic and Syntactic matrix computation for suspicious sentence
Semantic and Syntactic matrix computation for source sentence
Features computation
Stage 2
Semantic similarity computation
Syntactic similarity computation
Hybrid Semantic and syntactic similarity computation
Collect features (F1 : F34 , L)
Features ranking using chi-square algorithm (RF1 : RF34)
Features selection (FS1 : FSd)
Features selection
Stage 3
Training support vector machine
Support vector machine classifier (equation of hyper plane)
Classifier construction
Stage 4
Suspicious document
Source document
Suspicious sentence
Source sentence
Feature set (F1: F34)
Selected features (FS1 : FSd)
Joint matrix
Suspicious semantic matrix
Source semantic matrix
Semantic similarity
Label (0,1)
Classifier
Syntactic similarity
Combined semantic similarity
Combined syntactic similarity
Hybrid syntactic and semantic similarity
Feature ranking (RF1 : RF34)

**Fig. 4** Construction process of SVM classifier

This phase of the proposed system also takes into consideration the determination of most effective features that have the ability to discriminate the suspicious cases and differentiate the variations of the text similarities with the highest accuracy. Therefore, in the third stage, it is depended on the filter feature selection approach using Chi-square algorithm to rank the 34 features, and extracted the most discriminative features for detecting the text plagiarism including the different types of lexical, syntactic and semantic text plagiarisms. The proposed system is also depended on constructing the hyperplane equation of 34 features using SVM classification algorithm in the fourth stage, rather than conducting extensive experiments to find the best weighting coefficient values for incorporating the 34 features.

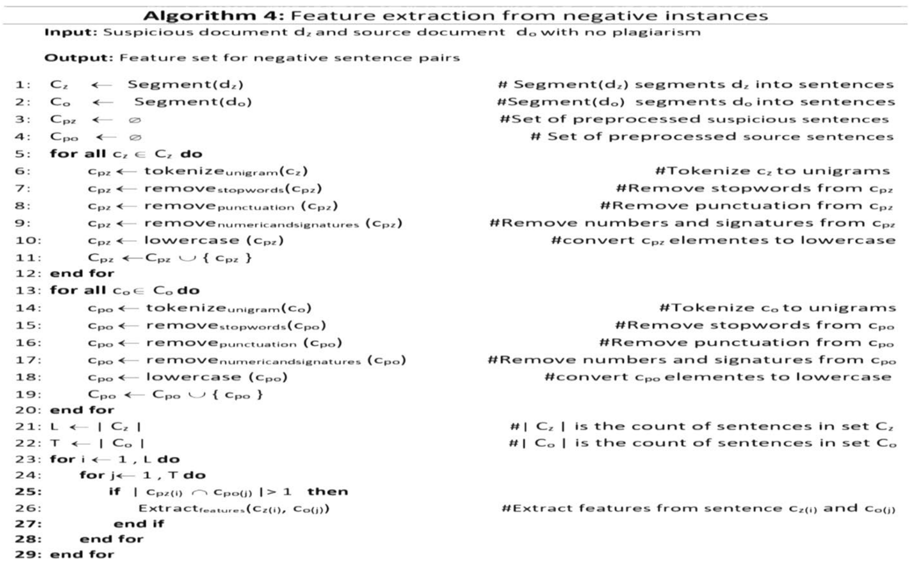
**Negative and positive instances extraction** In this stage, negative "Non-plagiarized" and positive "Plagiarized" cases are extracted from the training documents to build a supervised training database. The paragraphs of suspicious documents *dz* and source documents *do* are extracted from the training documents and converted into sentences. The extracted sentences are preprocessed using the preprocessing steps that described in Section 3.1. Then, each suspicious sentence will be compared with all the source sentences to compute meteor score and shared unigrams using algorithm 3 and 4 as shown in Figs. 5 and 6. The sentences are selected as a negative instance "Non-plagiarized case", if the shared unigrams of the non-plagiarized paragraphs are more than 1. The sentences are selected as a positive instance "Plagiarized case", if the pair of sentences have maximum meteor score and shared unigrams of the plagiarized paragraphs.



```
Algorithm 3: Feature extraction from positive instances
Input: Plagiarized passage $p_z$ and source passage $p_o$
Output: Feature set for positive sentence pairs

$C_z \leftarrow$ Segment($p_z$)                                   # Segment($p_z$) segments $p_z$ into sentences
$C_o \leftarrow$ Segment($p_o$)                                   #Segment($p_s$) segments $p_s$ into sentences
$C_{pz} \leftarrow \varnothing$                                     #Set of preprocessed suspicious sentences
$C_{po} \leftarrow \varnothing$                                     # Set of preprocessed source sentences
for all $c_z \in C_z$ do
    $c_{pz} \leftarrow$ tokenize$_{unigram}$($c_z$)                          #Tokenize $c_z$ to unigrams
    $c_{pz} \leftarrow$ remove$_{stopwords}$($c_{pz}$)                       #Remove stopwords from $c_{pz}$
    $c_{pz} \leftarrow$ remove$_{punctuation}$ ($c_{pz}$)                    #Remove punctuation from $c_{pz}$
    $c_{pz} \leftarrow$ remove$_{numericandsignatures}$ ($c_{pz}$)          #Remove numbers and signatures from $c_{pz}$
    $c_{pz} \leftarrow$ lowercase ($c_{pz}$)                             #convert $c_{pz}$ elementes to lowercase
    $C_{pz} \leftarrow C_{pz} \cup \{ c_{pz} \}$
end for
for all $c_o \in C_o$ do
    $c_{po} \leftarrow$ tokenize$_{unigram}$($c_o$)                          #Tokenize $c_o$to unigrams
    $c_{po} \leftarrow$ remove$_{stopwords}$($c_{po}$)                       #Remove stopwords from $c_{po}$
    $c_{po} \leftarrow$ remove$_{punctuation}$ ($c_{po}$)                    #Remove punctuation from $c_{po}$
    $c_{po} \leftarrow$ remove$_{numericandsignatures}$ ($c_{po}$)          #Remove numbers and signatures from $c_{po}$
    $c_{po} \leftarrow$ lowercase ($c_{po}$)                             #convert $c_{po}$ elementes to lowercase
    $C_{po} \leftarrow C_{po} \cup \{ c_{po} \}$
end for
$L \leftarrow | C_z |$                                          #| $C_z$ | is the number of sentences in set $C_z$
$T \leftarrow | C_s |$                                          #| $C_o$ | is the number of sentences in set $C_o$
for $i \leftarrow 1$ , L do
    $UL \leftarrow \varnothing$
    $ML \leftarrow \varnothing$
    for $j \leftarrow 1$ , T do
        $UL \leftarrow UL \cup$ {countshared$_{unigram}$($C_{pz(i)}$ ,$C_{po(j)}$ )}     # UL is list of values for shared unigrams
        $ML \leftarrow ML \cup$ { Meteor ($C_{pz(i)}$ ,$C_{po(j)}$ )}              # ML is list of  values for meteor score
    end for
    M1=max(UL)                                              #M1 maximum value of shared unigrams
    M2=max(ML)                                              #M2 maximum value of meteor scores
    for $j \leftarrow 1$ , T do
        if | $c_{pz(i)} \cap c_{po(j)}$ |= M1 and Meteor($c_{z(i)}$, $c_{o(j)}$)= M2  then     # M1 is the maximum value of shared
unigrams and M2  is the maximum Meteor score
            Extract$_{features}$($c_{z(i)}$, $c_{o(j)}$)                  #Extract features from sentence $c_{z(i)}$ and  $c_{o(j)}$
        end if
    end for
end for
```

**Fig. 5** Pseudo code of algorithm 3 to extract the features of the positive instance cases



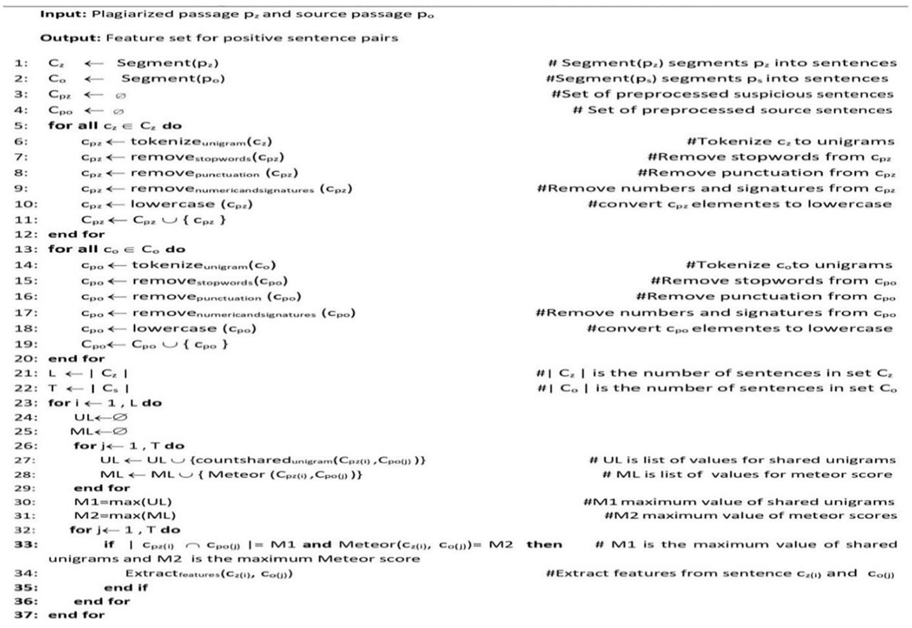
**Algorithm 4:** Feature extraction from negative instances

**Input:** Suspicious document $d_z$ and source document $d_o$ with no plagiarism

**Output:** Feature set for negative sentence pairs

```
1:  $C_z \leftarrow$ Segment($d_z$)                                   # Segment($d_z$) segments $d_z$ into sentences
2:  $C_o \leftarrow$ Segment($d_o$)                                   #Segment($d_o$) segments $d_o$ into sentences
3:  $C_{pz} \leftarrow \varnothing$                                    #Set of preprocessed suspicious sentences
4:  $C_{po} \leftarrow \varnothing$                                    # Set of preprocessed source sentences
5:  for all $c_z \in C_z$ do
6:      $c_{pz} \leftarrow$ tokenize$_{unigram}$($c_z$)                #Tokenize $c_z$ to unigrams
7:      $c_{pz} \leftarrow$ remove$_{stopwords}$($c_{pz}$)             #Remove stopwords from $c_{pz}$
8:      $c_{pz} \leftarrow$ remove$_{punctuation}$ ($c_{pz}$)          #Remove punctuation from $c_{pz}$
9:      $c_{pz} \leftarrow$ remove$_{numericandsignatures}$ ($c_{pz}$) #Remove numbers and signatures from $c_{pz}$
10:     $c_{pz} \leftarrow$ lowercase ($c_{pz}$)                       #convert $c_{pz}$ elementes to lowercase
11:     $C_{pz} \leftarrow C_{pz} \cup \{ c_{pz} \}$
12: end for
13: for all $c_o \in C_o$ do
14:     $c_{po} \leftarrow$ tokenize$_{unigram}$($c_o$)                #Tokenize $c_o$ to unigrams
15:     $c_{po} \leftarrow$ remove$_{stopwords}$($c_{po}$)             #Remove stopwords from $c_{po}$
16:     $c_{po} \leftarrow$ remove$_{punctuation}$ ($c_{po}$)          #Remove punctuation from $c_{po}$
17:     $c_{po} \leftarrow$ remove$_{numericandsignatures}$ ($c_{po}$) #Remove numbers and signatures from $c_{po}$
18:     $c_{po} \leftarrow$ lowercase ($c_{po}$)                       #convert $c_{po}$ elementes to lowercase
19:     $C_{po} \leftarrow C_{po} \cup \{ c_{po} \}$
20: end for
21: $L \leftarrow | C_z |$                                             #| $C_z$ | is the count of sentences in set $C_z$
22: $T \leftarrow | C_o |$                                             #| $C_o$ | is the count of sentences in set $C_o$
23: for $i \leftarrow 1$ , L do
24:     for $j \leftarrow 1$ , T do
25:         if $| c_{pz(i)} \cap c_{po(j)} | > 1$ then
26:             Extract$_{features}$($c_{z(i)}$, $c_{o(j)}$)           #Extract features from sentence $c_{z(i)}$ and $c_{o(j)}$
27:         end if
28:     end for
29: end for
```

**Fig. 6** Pseudo code of algorithm 4 to extract the features of the negative instance cases.



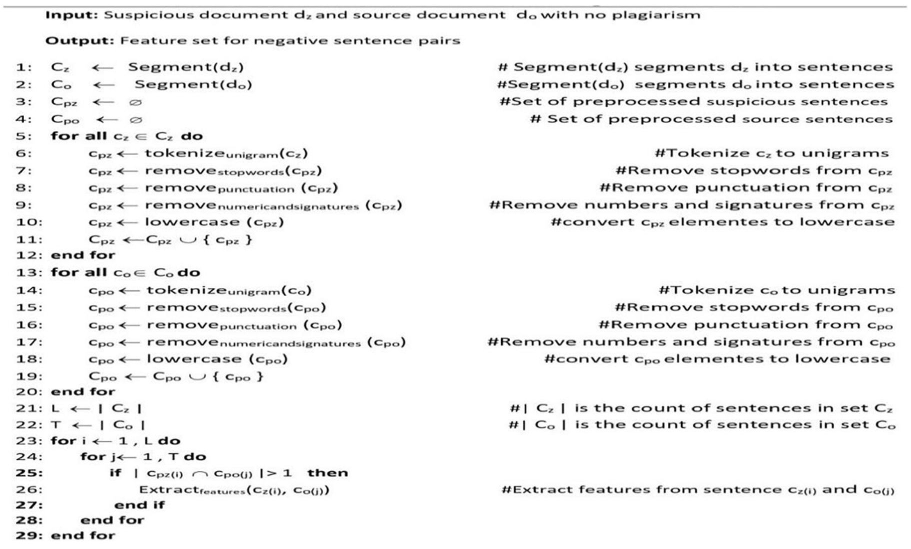
**Input:** Plagiarized passage $p_z$ and source passage $p_o$

**Output:** Feature set for positive sentence pairs

```
1:  $C_z \leftarrow$ Segment($p_z$)                                   # Segment($p_z$) segments $p_z$ into sentences
2:  $C_o \leftarrow$ Segment($p_o$)                                   #Segment($p_s$) segments $p_s$ into sentences
3:  $C_{pz} \leftarrow \varnothing$                                    #Set of preprocessed suspicious sentences
4:  $C_{po} \leftarrow \varnothing$                                    # Set of preprocessed source sentences
5:  for all $c_z \in C_z$ do
6:      $c_{pz} \leftarrow$ tokenize$_{unigram}$($c_z$)                #Tokenize $c_z$ to unigrams
7:      $c_{pz} \leftarrow$ remove$_{stopwords}$($c_{pz}$)             #Remove stopwords from $c_{pz}$
8:      $c_{pz} \leftarrow$ remove$_{punctuation}$ ($c_{pz}$)          #Remove punctuation from $c_{pz}$
9:      $c_{pz} \leftarrow$ remove$_{numericandsignatures}$ ($c_{pz}$) #Remove numbers and signatures from $c_{pz}$
10:     $c_{pz} \leftarrow$ lowercase ($c_{pz}$)                       #convert $c_{pz}$ elementes to lowercase
11:     $C_{pz} \leftarrow C_{pz} \cup \{ c_{pz} \}$
12: end for
13: for all $c_o \in C_o$ do
14:     $c_{po} \leftarrow$ tokenize$_{unigram}$($c_o$)                #Tokenize $c_o$to unigrams
15:     $c_{po} \leftarrow$ remove$_{stopwords}$($c_{po}$)             #Remove stopwords from $c_{po}$
16:     $c_{po} \leftarrow$ remove$_{punctuation}$ ($c_{po}$)          #Remove punctuation from $c_{po}$
17:     $c_{po} \leftarrow$ remove$_{numericandsignatures}$ ($c_{po}$) #Remove numbers and signatures from $c_{po}$
18:     $c_{po} \leftarrow$ lowercase ($c_{po}$)                       #convert $c_{po}$ elementes to lowercase
19:     $C_{po} \leftarrow C_{po} \cup \{ c_{po} \}$
20: end for
21: $L \leftarrow | C_z |$                                             #| $C_z$ | is the number of sentences in set $C_z$
22: $T \leftarrow | C_s |$                                             #| $C_o$ | is the number of sentences in set $C_o$
23: for $i \leftarrow 1$ , L do
24:     $UL \leftarrow \varnothing$
25:     $ML \leftarrow \varnothing$
26:     for $j \leftarrow 1$ , T do
27:         $UL \leftarrow UL \cup$ {countshared$_{unigram}$($C_{pz(i)}$ ,$C_{po(j)}$ )}   # UL is list of values for shared unigrams
28:         $ML \leftarrow ML \cup$ { Meteor ($C_{pz(i)}$ ,$C_{po(j)}$ )}                # ML is list of values for meteor score
29:     end for
30:     M1=max(UL)                                                     #M1 maximum value of shared unigrams
31:     M2=max(ML)                                                     #M2 maximum value of meteor scores
32:     for $j \leftarrow 1$ , T do
33:         if $| c_{pz(i)} \cap c_{po(j)} |$= M1 and Meteor($c_{z(i)}$, $c_{o(j)}$)= M2 then     # M1 is the maximum value of shared
    unigrams and M2 is the maximum Meteor score
34:             Extract$_{features}$($c_{z(i)}$, $c_{o(j)}$)           #Extract features from sentence $c_{z(i)}$ and $c_{o(j)}$
35:         end if
36:     end for
37: end for
```

**Algorithm 3** Feature extraction from positive instances

**Features computation** This stage aims to construct the training database of the SVM algorithm, it will be consisted by all the different similarity features that have the ability to detect the different types of the text plagiarism. Therefore, 34 values of the sentences



```
Input: Suspicious document $d_z$ and source document $d_o$ with no plagiarism
Output: Feature set for negative sentence pairs
$C_z \leftarrow$ Segment($d_z$)                              # Segment($d_z$) segments $d_z$ into sentences
$C_o \leftarrow$ Segment($d_o$)                              #Segment($d_o$) segments $d_o$ into sentences
$C_{pz} \leftarrow \varnothing$                                    #Set of preprocessed suspicious sentences
$C_{po} \leftarrow \varnothing$                                    # Set of preprocessed source sentences
for all $c_z \in C_z$ do
    $c_{pz} \leftarrow$ tokenize$_{unigram}$($c_z$)                  #Tokenize $c_z$ to unigrams
    $c_{pz} \leftarrow$ remove$_{stopwords}$($c_{pz}$)               #Remove stopwords from $c_{pz}$
    $c_{pz} \leftarrow$ remove$_{punctuation}$ ($c_{pz}$)            #Remove punctuation from $c_{pz}$
    $c_{pz} \leftarrow$ remove$_{numericandsignatures}$ ($c_{pz}$)   #Remove numbers and signatures from $c_{pz}$
    $c_{pz} \leftarrow$ lowercase ($c_{pz}$)                         #convert $c_{pz}$ elementes to lowercase
    $C_{pz} \leftarrow C_{pz} \cup \{ c_{pz} \}$
end for
for all $c_o \in C_o$ do
    $c_{po} \leftarrow$ tokenize$_{unigram}$($c_o$)                  #Tokenize $c_o$ to unigrams
    $c_{po} \leftarrow$ remove$_{stopwords}$($c_{po}$)               #Remove stopwords from $c_{po}$
    $c_{po} \leftarrow$ remove$_{punctuation}$ ($c_{po}$)            #Remove punctuation from $c_{po}$
    $c_{po} \leftarrow$ remove$_{numericandsignatures}$ ($c_{po}$)   #Remove numbers and signatures from $c_{po}$
    $c_{po} \leftarrow$ lowercase ($c_{po}$)                         #convert $c_{po}$ elementes to lowercase
    $C_{po} \leftarrow C_{po} \cup \{ c_{po} \}$
end for
$L \leftarrow | C_z |$                                         #| $C_z$ | is the count of sentences in set $C_z$
$T \leftarrow | C_o |$                                         #| $C_o$ | is the count of sentences in set $C_o$
for $i \leftarrow 1, L$ do
    for $j \leftarrow 1, T$ do
        if $| c_{pz(i)} \cap c_{po(j)} | > 1$ then
            Extract$_{features}$($c_{z(i)}$, $c_{o(j)}$)          #Extract features from sentence $c_{z(i)}$ and $c_{o(j)}$
        end if
    end for
end for
```

**Algorithm 4** Feature extraction from negative instances

similarity features are computed and recorded aggregating with the class label for each extracted case of the first stage. The computed sentence similarity features is based on similarity feature [17], hybrid syntactic and semantic similarity features [2], and 32 features have been calculated basing on four different criteria of the sentence similarity; cosine measure [70], dice measure [66], Jaccard measure [19] and Syntactic measure [29]. Each one of the sentence similarity is computed eight times basing on a different word similarity feature. The proposed system takes into consideration all the different features of the word similarities including path similarity [42], depth estimation similarity [53], combined word similarity [2], lch similarity [26], wup similarity [69], res similarity [49], lin similarity [30], and jcn similarity [21].

**Word similarity features** WordNet [34] lexical database was created to provide resemblance in the meaning between the words, it consists of the alternative synonyms and concepts for each word. These alternative concepts and synonyms are related to each other in relationships and structured in a tree called "Lexical tree".

- **Path similarity metric (PSM)** [42]**:** it is based on the shortest path length between two concepts in the lexical tree. The range of PSM values is between 0 to 1. If the two concepts are identical, "1" value is returned.

$$PSM\left(c_z, c_o\right) = \frac{1}{distance + 1} \tag{1}$$

where $c_z$ and $c_o$ are word concepts, and *distance* is shortest path between $\left(c_z, c_o\right)$.

- **Depth estimation similarity metric (DESM)** [53]**:** it is based on the depth and least common subsume (LCS) values.

$$DESM\left(c_z, c_o\right) = e^{-(depth\left(c_z\right)+depth\left(c_o\right)-2*depth(lcs\left(c_z,c_o\right)))} \tag{2}$$

Where *depth* $(c_x)$ is the maximum depth of the concept $c_x$ and *LCS* $(c_z, c_o)$ is the least common subsumer of the concept-pair.

- **Leacock-Chodorow similarity metric (LCHSM)** [26]**:** it uses the length of the shortest path between two concepts, and the maximum depth of the WordNet structure.

$$\text{LCHSM}\left(c_z, c_o\right) = -\log \frac{distance + 1}{2 * depth(c_z, c_o)} \tag{3}$$

- **Wu and Palmer similarity metric (WUPSM)** [69]**:** it is based on the depth of each concept and the depth of the nearest ancestor shared by both concepts, also known as the least common subsume (LCS).

$$WUPSM(c_z, c_o) = \frac{2 * depth(LCS(c_z, c_o))}{depth\left(c_z\right) + depth(c_o)} \tag{4}$$

- **Resnik similarity metric (RESSM)** [49]**:** it uses the Information Content (IC) of LCS. IC metric assesses a concept's specificity. It is based on the word frequency in a corpus, where each occurrence of the word affects the counts of all to its WordNet taxonomic ancestors.

$$RESSM\ \left(c_z, c_o\right) = IC(LCS(c_z, c_o)) \tag{5}$$

$$IC(c) = \log \frac{1}{P(c)} \tag{6}$$

$$P(c) = \frac{\sum_{t \in T(c)} appearances(t)}{V} \tag{7}$$

Where *P(c)* probability of the concept c in corpus,$T(c)$ is the set of terms in the corpus that can be inferred by *c* and *v* is the total number of corpus terms.

- **Lin similarity metric (LINSM)** [30]**:** it is based on Information Content (IC) metric.

$$LINSM\left(c_z, c_o\right) = \frac{2 * IC(LCS(c_z, c_o))}{IC\left(c_z\right) + IC\left(c_o\right)} \tag{8}$$

- **Jiang and Conrath similarity metric (JCNSM)** [21]**:** it is based on IC and LCS metrics.

$$JCNSM\left(c_z, c_o\right) = \frac{1}{IC\left(c_z\right) + IC\left(c_o\right) - 2 * IC(LCS(c_z, c_o))} \tag{9}$$

**Sentence similarity features:** these features were focused on two types of sentence similarity: semantic similarity [17, 19, 70] and syntactic similarity [29]. These features were computed based on four steps. The first step is to construct a joint matrix, given two preprocessed sentences $CP_z$ and $CP_o$, "joint matrix" was constructed using unique words of the sentences. Let $JM = \{W_1, W_2, \cdots Wd\}$ indicates a joint matrix, where d is the number of unique words in the joint matrix.

The second step constructs the semantic matrices by using semantic similarity between the words, it started by calculating the semantic similarity between the words using 7-WordNet similarity metrics [34]. The dimension of the semantic matrix equals the number of words in the joint matrix, which each cell corresponds to a word in the joint matrix. The semantic matrix's cells are weighted based on the estimated semantic similarity between the words in the joint matrix and the corresponding sentence. As an example, If the word *W* of the joint matrix exists in the sentence $CP_z$, the semantic matrix's weight of the word is assigned to 1. Otherwise, the weight of *W* in the semantic matrix is assigned to 0, if there is no similarity value between *W* and all of the words in the sentence $CP_z$.

The third step computes the semantic similarity measurements. The proposed system is based on three measures of semantic similarity: cosine measure Eq. (10), dice measure Eq. (11), and Jaccard measure Eq. (12), to measure the similarity degree between $CP_z$ and $CP_o$ using the semantic matrix.

$$\text{cosine } similarity\,(CM_z, CM_o) = \frac{\sum_{i=1}^{v} R_{zi} * R_{oi}}{\sqrt{\sum_{i=1}^{v} R_{zi}^2} * \sqrt{\sum_{i=1}^{v} R_{oi}^2}} \tag{10}$$

$$dice\ similarity\,(CM_z, CM_o) = \frac{2 * \sum_{i=1}^{v} R_{zi} * R_{oi}}{\sum_{i=1}^{v} R_{zi}^2 + \sum_{i=1}^{v} R_{oi}^2} \tag{11}$$

$$\text{Jaccard } similarity\left(CM_z, CM_o\right) = \frac{\sum_{i=1}^{v} R_{zi} * R_{oi}}{\sum_{i=1}^{v} R_{zi}^2 + \sum_{i=1}^{v} R_{oi}^2 - \sum_{i=1}^{v} R_{zi} * R_{oi}} \tag{12}$$

where $CM_z = (R_{11}, R_{12}, R_{13}, \ldots R_{1v})$ and $CM_o = (R_{21}, R_{22}, R_{23}, \ldots, R_{2v})$ are semantic matrices of the suspicious and source sentences, respectively, $R_{zi}$ is the suspicious semantic matrix's weight, $R_{oi}$ is the source semantic matrix's weight, and $v$ is the number of words.

The fourth step computes Word-order similarity measurements, it is utilized syntactic-matrix technique [29] to calculate the word-order similarity. The syntactic matrix is created from the joint matrix and corresponding sentence. For each *W* word in the joint matrix. The cell in the syntactic matrix is assigned to the index position of the corresponding word in the sentence $CP_z$, if the W appeared in the sentence $CP_z$. If the word W does not exist in sentence $CP_z$, a semantic similarity measure is computed between the W and each word in sentence $CP_z$. The cell's value is assigned to the index position of the word with the highest similarity measure in the sentence $CP_z$.

$$\text{Syntactic similarity (Jz, Jo)} = 1 - \frac{||J_z - J_o||}{||J_z + J_o||} \tag{13}$$

where $J_z = (J_{11}, J_{12}, J_{13}, \ldots, J_{1v})$ and $J_o = (J_{21}, J_{22}, J_{23}, \ldots, J_{2v})$ are syntactic matrices of the suspicious and source sentences, respectively. $\|Jz - Jo\|$ and $\|Jz + Jo\|$ are computed as follows:

$$||J_z - J_o|| = \sqrt{(J_{11} - J_{21})^2 + (J_{12} - J_{22})^2 + (J_{13} - J_{23})^2 + \cdots + (J_{1v} - J_{2v})^2} \tag{14}$$

$$||J_z + J_o|| = \sqrt{(J_{11} + J_{21})^2 + (J_{12} + J_{22})^2 + (J_{13} + J_{23})^2 + \cdots + (J_{1v} + J_{2v})^2} \tag{15}$$

**Hybrid syntactic and semantic features:** it is based on syntactic and semantic features, which used a cooperation coefficient $\alpha$ to weight the path and depth similarity metrics, it is estimation as shown in Eq. (16), where $\alpha$ between 0 and 1, $PSM\left(c_z, c_o\right)$ is the path similarity metric, and $DESM\left(c_z, c_o\right)$ is the depth estimation similarity metric.

$$combined\ word\ similarity\ (c_z, c_o) = \alpha * (PSM\left(c_z, c_o\right)) + (1 - \alpha) * (DESM\left(c_z, c_o\right)) \tag{16}$$

Then, syntactic similarity is calculated by using Eq. (13), and semantic similarities are calculated through Eqs. (10), (11), and (12). Afterwards, semantic similarity is computed using Eq. (17), where $0 \leq \beta \leq 1$ is used to determine the weights of semantic and syntactic.

$$\begin{aligned} &Hybrid\ syntactic\ and\ semantic\left(CM_z, CM_o\right) = \beta * (dice\ similarity\ \left(CM_z, CM_o\right) \\ &+ (1 - \beta) * syntactic\ similarity(J_z, J_o) \end{aligned} \tag{17}$$

**Fuzzy similarity feature:** the semantic similarity between the preprocessed suspicious sentence $\mathrm{Cp}_z$ and source sentence $\mathrm{Cp}_o$ is computed as shown in Eq. (18). Firstly, for each suspicious-source pair of words ($W_z$, $W_o$), synsets are selected using WordNet lexical database. A synset is a synonym ring of the words that are semantically similar [3]. Secondly, the synset lists of $W_z$ and $W_o$ "$W_{zsyn}$ *and* $W_{osyn}$" are created. Then, the semantic similarity between each pair of synsets ($w_{zsyn}$, $w_{osyn}$) is calculated using wup_similarity metric [69] Eq. (4) based on the fuzzy function $F_{z,\mathrm{o}}$ as shown in Eq. (20). This function adapts the semantic value using heuristic boundary conditions, and selects the maximum value of the synset pairs.

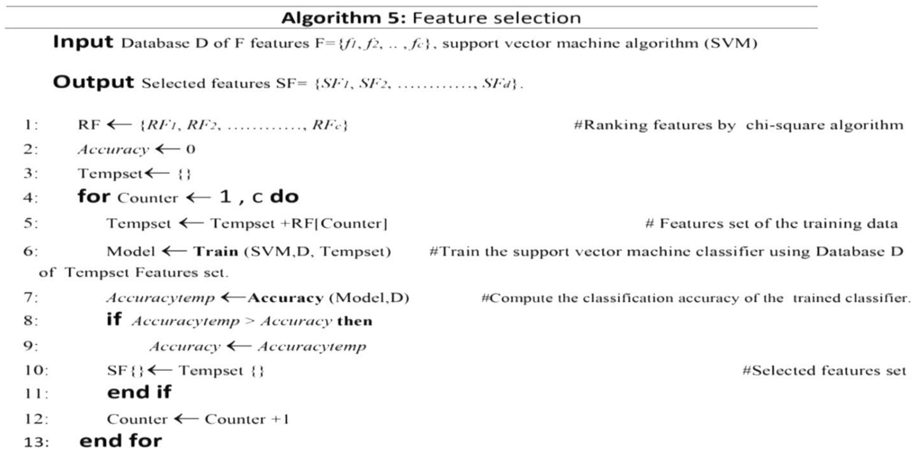
$$Sim(CPz, CPo) = (\mu_{1,o} + \mu_{2,o} + \cdots + \mu_{z,o} + \cdots + \mu_{n,o})/\mathrm{n} \tag{18}$$

$$\mu_{z,o} = 1 - \prod\nolimits_{wo \in Co} (1 - F_{z,o}) \tag{19}$$

$$Fz, o = \max_{wzsyn \in Wzsyn, wosyn \in Wosyn} Fz, \mathrm{o}(WUPSM(wzsyn, wosyn)) = \begin{cases} 1.0 ifp = 1.0 \\ 0.7 ifp \in [0.7, 1.0) \\ 0.5 ifp \in [0.5, 0.7) \\ 0.3 ifp \in [0.3, 0.5) \\ 0.2 ifp \in (0.0.0.3) \\ 0.0 ifp = 0.0 \end{cases} \tag{20}$$

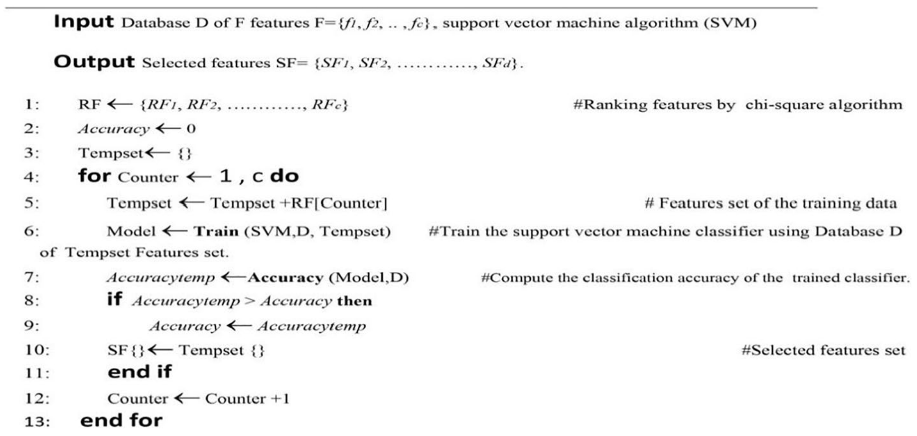
where $\mu_{z,o}$ is the word-to-sentence correlation factor for each word $W_z$ in $CP_z$ and $CP_o$, $n$ is the total number of words in $CP_z$, $Fz$,o is the fuzzy-semantic similarity between $W_z$ and $W_o$, and $p$ is the output of fuzzy-semantic similarity.

**Features selection** After computing the 34 features that described in the previous stage, the feature selection stage of this phase aims to extract the most discriminative features



```
Algorithm 5: Feature selection
Input Database D of F features F={f1, f2, .. , fc}, support vector machine algorithm (SVM)
Output Selected features SF= {SF1, SF2, ............, SFd}.
1:  RF ← {RF1, RF2, ............, RFc}                    #Ranking features by chi-square algorithm
2:  Accuracy ← 0
3:  Tempset← {}
4:  for Counter ← 1 , c do
5:      Tempset ← Tempset +RF[Counter]                    # Features set of the training data
6:      Model ← Train (SVM,D, Tempset)    #Train the support vector machine classifier using Database D
 of Tempset Features set.
7:      Accuracytemp ←Accuracy (Model,D)      #Compute the classification accuracy of the trained classifier.
8:      if Accuracytemp > Accuracy then
9:          Accuracy ← Accuracytemp
10:     SF{}← Tempset {}                                  #Selected features set
11:     end if
12:     Counter ← Counter +1
13: end for
```

**Fig. 7** Pseudo code of algorithm 5 for the feature selection stage.



```
Input Database D of F features F={f1, f2, .. , fc}, support vector machine algorithm (SVM)
Output Selected features SF= {SF1, SF2, ............, SFd}.
1:  RF ← {RF1, RF2, ............, RFc}                    #Ranking features by chi-square algorithm
2:  Accuracy ← 0
3:  Tempset← {}
4:  for Counter ← 1 , c do
5:      Tempset ← Tempset +RF[Counter]                    # Features set of the training data
6:      Model ← Train (SVM,D, Tempset)    #Train the support vector machine classifier using Database D
 of Tempset Features set.
7:      Accuracytemp ←Accuracy (Model,D)      #Compute the classification accuracy of the trained classifier.
8:      if Accuracytemp > Accuracy then
9:          Accuracy ← Accuracytemp
10:     SF{}← Tempset {}                                  #Selected features set
11:     end if
12:     Counter ← Counter +1
13: end for
```

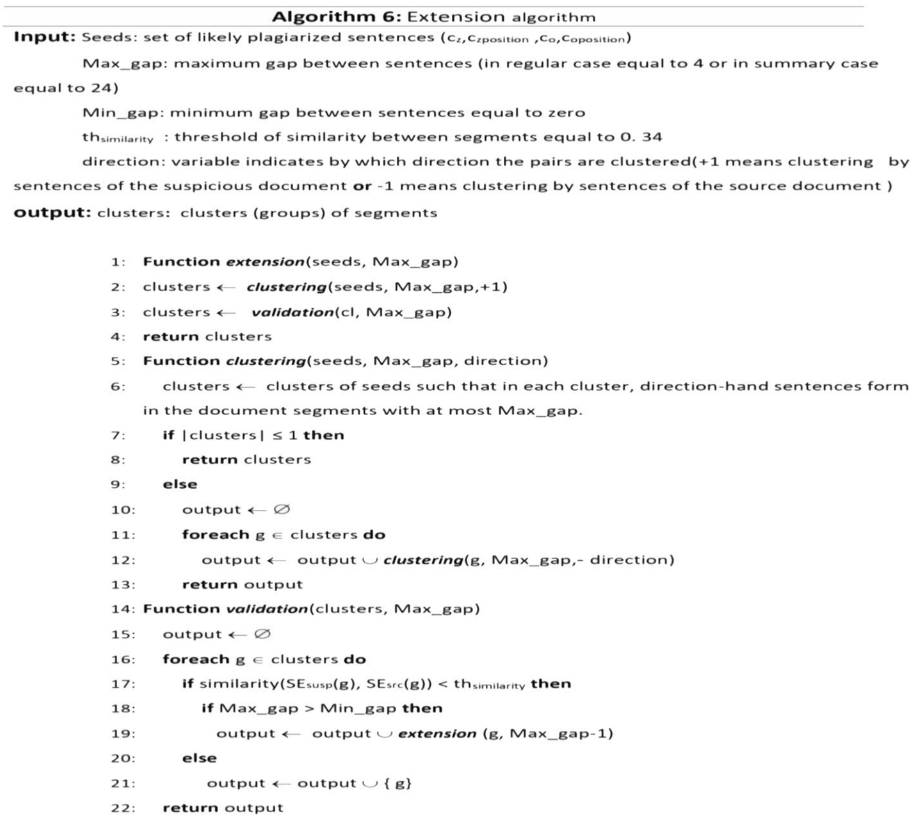
**Algorithm 5** Feature selection

for detecting the text plagiarism with the highest classification accuracy. Chi-square algorithm [41] is used to rank the features, which depends on the calculation of Chi-square value between each feature and the class labels. Then, the proposed system computes the classification accuracy for each subset of the ranked features. The operational work of this step is described in algorithm 5 as shown in Fig. 7. It starts with the first feature in the list of the ranked features as a first subset, it will continue generating new subsets of the ranked features to arrive to the last subset that have all the features, where each subset of the ranked features consists of the features of previous subset plus the next feature in the list of the ranked features. The subset of the ranked features that achieved the highest classification accuracy during the repetition process is selected to be the best subset of the features that have the ability to discover the different types of lexical, syntactic and semantic plagiarisms.

**Construction process of SVM classifier** After selecting the most effective subset of the computed features, the proposed system is based on SVM classification algorithm to fit the values of the selected features, and find the hyperplane equation of the selected features that have the ability to detect the plagiarism cases, rather than conducting extensive experiments to find the best weighting coefficient values for incorporating the selected subset of features. SVM classification algorithm is more effective in high dimensional structured datasets compared with the other classification algorithms, which has the ability to add new dimensionality to distinguish the overlapping between the training cases of different classes.

## 3.3 Post processing

Post processing phase of the proposed system aims to extract the best-plagiarized segment between the suspicious and source documents using filter seeds, merging adjacent detected seeds, adaptive behavior, and filter segments techniques. This phase helps to improve precision accuracy value by removing some “bad” plagiarism cases using filtering techniques. It also improves recall and granularity accuracy values using extension technique.



**Algorithm 6:** Extension algorithm

**Input:** Seeds: set of likely plagiarized sentences ($c_z, c_{zposition}, c_o, c_{oposition}$)

Max_gap: maximum gap between sentences (in regular case equal to 4 or in summary case equal to 24)

Min_gap: minimum gap between sentences equal to zero

$th_{similarity}$ : threshold of similarity between segments equal to 0. 34

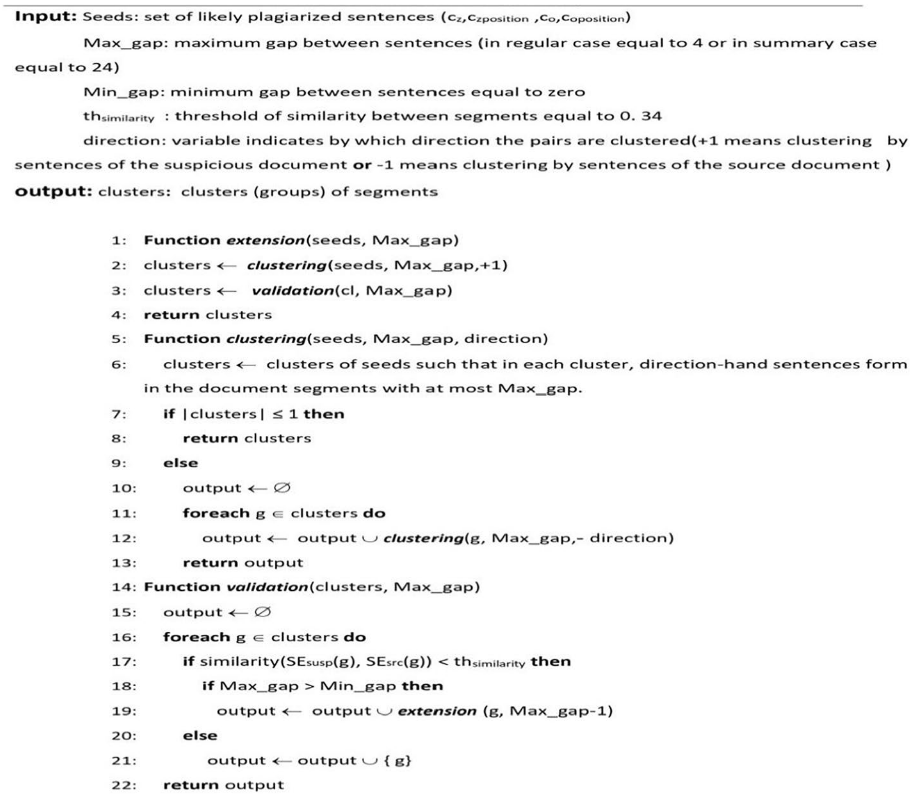
direction: variable indicates by which direction the pairs are clustered(+1 means clustering by sentences of the suspicious document **or** -1 means clustering by sentences of the source document )

**output:** clusters: clusters (groups) of segments

```
Function extension(seeds, Max_gap)
clusters ← clustering(seeds, Max_gap,+1)
clusters ← validation(cl, Max_gap)
return clusters
Function clustering(seeds, Max_gap, direction)
  clusters ← clusters of seeds such that in each cluster, direction-hand sentences form
  in the document segments with at most Max_gap.
  if |clusters| ≤ 1 then
    return clusters
  else
    output ← ∅
    foreach g ∈ clusters do
      output ← output ∪ clustering(g, Max_gap,- direction)
    return output
Function validation(clusters, Max_gap)
  output ← ∅
  foreach g ∈ clusters do
    if similarity(SE_susp(g), SE_src(g)) < th_similarity then
      if Max_gap > Min_gap then
        output ← output ∪ extension (g, Max_gap-1)
    else
        output ← output ∪ { g}
  return output
```

**Fig. 8** Pseudo code of algorithm 6 for extension technique

- Seeding phase of the proposed system may detect the plagiarized case with multiple source sentences. Therefore, filter seed technique is used to select the source sentence that having the highest Meteor score value.
- The proposed system is also based on filter segments, which removed small cases by rejecting the plagiarized the small segments; if either suspicions or source segments have a length less than 145 characters as a threshold of the suspicious segment, and 250 characters as a threshold for source segment, the case is discarded, else the case is saved.
- Extension technique is also used to merge adjacent detected seeds that are similar between the suspicious and source documents, to form larger text segments. The work flow of this technique is described in algorithm 6 as shown in Fig. 8. It depends on recursive algorithm used by Sanchez-Perez et al. [54]. The extension step is divided into two processes: clustering and validation. In the clustering process, seeds that are not separated by a gap will be grouped. But to avoid adding a noise in the clustering process, the extension technique uses a validating process, which bases on a similarity threshold (*$th_{similarity}$*) between the text segments.



```
Input: Seeds: set of likely plagiarized sentences ($c_z, c_{zposition}, c_o, c_{oposition}$)
        Max_gap: maximum gap between sentences (in regular case equal to 4 or in summary case equal to 24)
        Min_gap: minimum gap between sentences equal to zero
        $th_{similarity}$ : threshold of similarity between segments equal to 0. 34
        direction: variable indicates by which direction the pairs are clustered(+1 means clustering by sentences of the suspicious document or -1 means clustering by sentences of the source document )
output: clusters: clusters (groups) of segments

Function extension(seeds, Max_gap)
clusters ← clustering(seeds, Max_gap,+1)
clusters ← validation(cl, Max_gap)
return clusters
Function clustering(seeds, Max_gap, direction)
  clusters ← clusters of seeds such that in each cluster, direction-hand sentences form in the document segments with at most Max_gap.
  if |clusters| ≤ 1 then
    return clusters
  else
    output ← ∅
    foreach g ∈ clusters do
      output ← output ∪ clustering(g, Max_gap,- direction)
    return output
Function validation(clusters, Max_gap)
  output ← ∅
  foreach g ∈ clusters do
    if similarity($SE_{susp}$(g), $SE_{src}$(g)) < $th_{similarity}$ then
      if Max_gap > Min_gap then
        output ← output ∪ extension (g, Max_gap-1)
    else
      output ← output ∪ { g}
  return output
```

**Algorithm 6** Extension algorithm

Pan workshop series [44, 45, 47] is an international competition interested in the plagiarism detection, each of them consists of different plagiarism types. Pan 2013 consists of no-obfuscation, random obfuscation, translation obfuscation, and summary obfuscation types. Pan 2014 includes no-obfuscation and random obfuscation types. The parameters that achieved the best result in each plagiarism type are different from another type. Therefore, the proposed system has the ability to adapt its behavior depending on the adaptive extension technique, it is based on an extension algorithm using two different gap values of the sentences: maxgap_summary and maxgap. Maxgap_summary is the best observed gap value between the sentences of the summary obfuscation type, and maxgap is the best observed gap value between the sentences of other types.

## 4 Experimental evaluation

The proposed system was implemented by using python programming language on an Intel®Core™i5-4210U CPU @ 1.70 GHz-2.40 GHz and computer with a 4.00 GB RAM.

### 4.1 Datasets used for experiments

Experiments were conducted on three datasets from the PAN Workshop series: PAN 2012 [44], PAN 2013 [45] and PAN 2014 [47]. Each of them contains suspicious and source documents. Authors applied many obfuscation strategies on different length paragraphs of the source documents and incorporated into the suspicious documents. Number of document for each obfuscation strategy in the training and testing corpus of PAN 2012, PAN 2013 and PAN 2014 are described in Table 2. PAN 2012 used the books available at Project Gutenberg to extract its suspicious and source documents, it consists of training and testing sets. The training set contained 1804 suspicious and 4210 source documents, while the test set contained 3000 suspicious and 3500 source documents. PAN 2013 depended on ClueWeb 2009 corpus to extract its suspicious and source documents, it comprises of 3653 suspicious documents and 4774 source documents. PAN 2014 was also depended on ClueWeb 2009 corpus as PAN 2013, it has the same training corpus of PAN 2013 but introduced an additional test set.

### 4.2 Evaluation metrics

The proposed system performance is evaluated according to plagdet score, which was proposed by Potthast et al. [46] to rank the various proposed systems of the plagiarism detection depending on PAN competitions. There are other evaluation metrics also used to rank the various proposed systems including the overall score of precision, recall, F-measure, and granularity metrics.

$$plagdet\ (Q, W)(\%) = \left( \frac{F - measure}{\log_2(1 + gran(Q, W))} \right) \times 100 \tag{21}$$

where the $F$-measure is the harmonic mean of recall and precision scores, it is determined using the following equation:

$$F - measure\ (\%) = \left( \frac{2 \times rec(Q, W) \times prec(Q, W)}{rec(Q, W) + prec(Q, W)} \right) \times 100 \tag{22}$$

**Table 2** Comparison between Statistics of the PAN 2012, PAN 2013 and PAN 2014 datasets

| Obfuscation Approach | Approach explanation | Number of document pairs | | | | | |
|---|---|---|---|---|---|---|---|
| | | Test set | | | Training set | | |
| | | PAN 2012 | PAN 2013 | PAN 2014 | PAN 2012 | PAN 2013 | PAN 2014 |
| No plagiarism | Suspicious documents were not plagiarized | 500 | 1000 | 1600 | 1000 | 1000 | |
| No obfuscation | Source passages without any change (copy–paste) were added to the suspicious document | 500 | 1000 | 1600 | 1000 | 1000 | |
| Random obfuscation | artificial tool was used to obfuscated Source passages | - | 1000 | 1600 | - | 1000 | |
| Low obfuscation | Artificial tool with a low degree was used to obfuscated Source passages | 500 | - | | 1000 | - | |
| High obfuscation | Artificial tool with a high degree was used to obfuscated Source passages | 500 | - | | 1000 | - | |
| Paraphrasing | Humans paraphrased the Source passages were added to the suspicious document | 500 | - | | 1000 | - | |
| Translation | Source passages were translated to English where the originally written in Spanish and in German | 500 | - | | 1000 | - | |
| Cyclic translation | Different machine translators were used to translate source passages into several languages, then were returned to the original language | - | 1000 | - | - | 1000 | |
| Summarization | Humans summarized the Source passages | - | 1185 | - | - | 1185 | |

$$prec(Q, W)(\%) = \left( \frac{1}{W} \times \sum\nolimits_{w \in W} \frac{\left| \bigcup_{q \in Q} (q \cap w) \right|}{|w|} \right) \times 100 \tag{23}$$

$$rec(Q, W)(\%) = \left( \frac{1}{Q} \times \sum\nolimits_{q \in Q} \frac{\left| \bigcup_{w \in W} (q \cap w) \right|}{|q|} \right) \times 100 \tag{24}$$

where

$$q \cap w = \begin{cases} q \cap w \;\; if\ w\ detects\ q(number\ of\ overlapping\ characters) \\ \phi\ otherwise \end{cases}$$

where $w$ indicates a detected plagiarism case, $W$ indicates the set of all detected plagiarism cases provided that $w \in W$, $q$ indicates an actual plagiarism case, and $Q$ indicates the set of all plagiarism cases provided that $q \in Q$. Precision metric is the proportion of properly matched characters in the given documents to the total number of characters retrieved as shown in Eq. (23). Recall metric is the proportion of properly matched characters between the given documents to the number of actual plagiarized characters as shown in Eq. (24). Neither precision nor recall introduces interpretation, if the plagiarism detectors may indicate overlapping or multiple detections for a single plagiarism case. Therefore, in order to overcome this limitation, the granularity of a detector is also measured as follows:

$$gran(Q, W) = \frac{1}{|Q_W|} \times \sum\nolimits_{q \in Q_W} |W_Q| \tag{25}$$

where $Q_W \subseteq Q$ are the cases detected in W and $W_Q \subseteq W$ are all the detections of case q.

## 4.3 Results and discussion

The proposed system depends on two paths to detect the text plagiarism. The first path is based on traditional paragraph-level comparison, and the second path is based on SVM classifier. The second path is used, if the first path didn't able to discover the text similarity, it is based on constructing SVM classifier that has the ability to detect all the different types of lexical, syntactic, and semantic plagiarism cases. The proposed system depends on building a supervised training database to train SVM algorithm. Therefore, Negative "Non-plagiarized" and positive "Plagiarized" cases are extracted from PAN 2012 and PAN 2013 documents as explained in Section 3.2.3. Thirty-four values of the sentences similarity features are computed and recorded aggregating with the class label for each extracted case to build the supervised training database of SVM algorithm.

**Table 3** Comparison results of the first and second experiments for the random obfuscation on PAN 2013 test corpus

| No. of experiment | PlagDet (%) | F-measure (%) | Recall (%) | Precision (%) | Granularity |
|---|---|---|---|---|---|
| First | 83.98 | 84.09 | 75.89 | 94.28 | 1.00190 |
| Second | 88.33 | 88.44 | 86.72 | 90.22 | 1.00171 |

**Table 4** Comparison results of the first and second experiments on PAN 2013 of the complete test corpus

| No. of experiment | PlagDet (%) | F-measure (%) | Recall (%) | Precision (%) | Granularity |
|---|---|---|---|---|---|
| First | 86.74 | 86.85 | 79.32 | 95.97 | 1.00173 |
| Second | 89.12 | 89.34 | 86.56 | 92.32 | 1.00349 |

**Table 5** Comparison results of the first and second experiments on PAN 2014 of the complete test corpus

| No. of experiment | PlagDet (%) | F-measure (%) | Recall (%) | Precision (%) | Granularity |
|---|---|---|---|---|---|
| First | 90.62 | 90.66 | 84.51 | 97.76 | 1.00056 |
| Second | 92.91 | 92.95 | 90.14 | 95.94 | 1.00053 |

Two supervised training datasets are created. The first dataset is contained by the extracted positive and negative cases from the documents of PAN 2012. The second dataset is created by the extracted cases from the documents of PAN 2012 and PAN 2013. Two experiments are conducted to evaluate the created datasets. In the first experiment, the proposed system is constructed depending on the first dataset. The second experiment, SVM of the proposed system is trained on the second dataset. The purpose of the second experiment is to train the proposed system on more different cases of the text plagiarism and show their effectiveness. Tables 3, 4 and 5 show the performance results of the two experiments using test documents of random obfuscation PAN 2013 sub-corpora, complete PAN 2013 corpus, and complete PAN 2014. The results show that the proposed system that trained on the second dataset achieved the highest classification accuracy comparing with the constructed model that trained on the first dataset. This indicates that the second dataset creating by the extracted cases from the documents of PAN 2012 and PAN 2013 is more effective to train SVM, which make it more accurate to discover the different types of the text plagiarism.

The statistical analysis of the constructed training database of SVM algorithm is developed and shown in Table 6 to explain the importance of each created feature. The range of values for all sentence similarity features of the constructed training database is between 1 and 0. If the feature value is closer to 1, this indicates that the two sentences are similar, and if the feature value is closer to 0, this indicates that the two sentences are dissimilar. The discriminative sentence similarity feature that has the ability to differentiate the positive and negative cases with high accuracy, is the feature that contains high intra similarity and low inter similarity values of the class labels. Therefore, metrics of the mean, standard deviation and 95% confidence limits are calculated for each sentence similarity feature. Whenever the feature mean values of the positive and negative cases are closer, and 95% confidence limits values of positive cases are also closer to the mean value of negative cases, this indicates that relying on this feature alone will cause a confusion in the decision. As shown in Table 6, most of the sentence similarity features have closer positive and negative mean values, and 95% confidence limits values of the positive cases are far from the one value and closer to the positive and negative mean values. Therefore, the previous researches [2, 3, 26, 53, 67, 69] depended on 2, 3 or 4 sentence similarity features instead of depending on one feature to enhance the text plagiarism detection. There is also a challenge to depend on 2, 3 or 4 sentence similarity features, because these features are

**Table 6** Statistical analysis of the constructed dataset

| Sentence Similarity Feature | The Extracted Cases | | 95% Confidence Limits | |
|---|---|---|---|---|
| | Negative<br>N = 42983<br>Mean ± Standard Deviation | Positive<br>N = 42983<br>Mean ± Standard Deviation | Lower limit | Upper limit |
| Syntactic with word path similarity | 0.5114±0.1185 | 0.7407±0.1832 | 0.624784 | 0.627354 |
| Dice Semantic with word path similarity | 0.5324±0.0871 | 0.7673±0.1844 | 0.648576 | 0.651062 |
| Jaccard Semantic with word path similarity | 0.3676±0.0821 | 0.6574±0.2374 | 0.510969 | 0.514033 |
| Cosine Semantic with word path similarity | 0.5515±0.0847 | 0.7781±0.1746 | 0.663596 | 0.665975 |
| Syntactic with word depth estimation | 0.476±0.1212 | 0.7227±0.1956 | 0.59796 | 0.60069 |
| Dice Semantic with word depth estimation | 0.3427±0.1066 | 0.6908±0.2362 | 0.515034 | 0.518413 |
| Jaccard Semantic with word depth estimation | 0.2119±0.0808 | 0.5772±0.2796 | 0.392712 | 0.396391 |
| Cosine Semantic with word depth estimation | 0.3563±0.1066 | 0.6999±0.2294 | 0.52646 | 0.529776 |
| Syntactic with combined word similarity | 0.1185±0.1185 | 0.7409±0.1831 | 0.62525 | 0.627817 |
| Dice Semantic with combined word similarity | 0.442±0.0953 | 0.7306±0.2085 | 0.584856 | 0.587757 |
| Jaccard Semantic with combined word similarity | 0.2886±0.0808 | 0.6173±0.2581 | 0.451267 | 0.454638 |
| Cosine Semantic with combined word similarity | 0.4592±0.0939 | 0.7409±0.1998 | 0.598648 | 0.601458 |
| Syntactic with WUP word similarity | 0.5228±0.1183 | 0.7454±0.1804 | 0.632877 | 0.635402 |
| Dice Semantic with WUP word similarity | 0.8458±0.0641 | 0.9005±0.1089 | 0.872506 | 0.873755 |
| Jaccard Semantic with WUP word similarity | 0.7379±0.0927 | 0.8342±0.1563 | 0.785136 | 0.78697 |
| Cosine Semantic with WUP word similarity | 0.8555±0.06 | 0.9073±0.1 | 0.880838 | 0.881993 |
| Syntactic with LCH word similarity | 0.5094±0.1186 | 0.7391±0.1834 | 0.622928 | 0.625501 |
| Dice Semantic with LCH word similarity | 0.7297±0.042 | 0.7653±0.145 | 0.7468 | 0.748247 |
| Jaccard Semantic with LCH word similarity | 0.5762±0.0527 | 0.6438±0.2059 | 0.608993 | 0.611052 |
| Cosine Semantic with LCH word similarity | 0.7423±0.0483 | 0.7743±0.1454 | 0.757545 | 0.75901 |
| Syntactic with RES word similarity | 0.5108±0.1189 | 0.7348±0.1844 | 0.62152 | 0.624077 |
| Dice Semantic with RES word similarity | 0.3692±0.0714 | 0.4859±0.287 | 0.426109 | 0.429012 |
| Jaccard Semantic with RES word similarity | 0.2287±0.0538 | 0.3819±0.3251 | 0.303687 | 0.306967 |

Table 6 (continued)

| Sentence Similarity Feature | The Extracted Cases | | 95% Confidence Limits | |
|---|---|---|---|---|
| | Negative<br>N = 42983 | Positive<br>N = 42983 | Lower limit | Upper limit |
| | Mean ± Standard Deviation | Mean ± Standard Deviation | | |
| Cosine Semantic with RES word similarity | 0.3862±0.0872 | 0.5092±0.2925 | 0.446195 | 0.449196 |
| Syntactic with JCN word similarity | 0.5178±0.1202 | 0.7361±0.1832 | 0.625709 | 0.628244 |
| Dice Semantic with JCN word similarity | 0.2494±0.224 | 0.4086±0.4094 | 0.326756 | 0.331294 |
| Jaccard Semantic with JCN word similarity | 0.1616±0.1503 | 0.3562±0.3924 | 0.25682 | 0.261 |
| Cosine Semantic with JCN word similarity | 0.2611±0.2329 | 0.4161±0.4114 | 0.336309 | 0.340896 |
| Syntactic with LIN word similarity | 0.5278±0.1176 | 0.7404±0.1805 | 0.632853 | 0.635337 |
| Dice Semantic with LIN word similarity | 0.747±0.0913 | 0.8414±0.151 | 0.793299 | 0.795083 |
| Jaccard Semantic with LIN word similarity | 0.6042±0.1121 | 0.7521±0.202 | 0.676992 | 0.679389 |
| Cosine Semantic with LIN word similarity | 0.7615±0.0854 | 0.8502±0.1414 | 0.805012 | 0.806682 |
| Hybrid similarity | 0.8422±0.1048 | 0.9083±0.1312 | 0.874448 | 0.876095 |
| Fuzzy Semantic | 0.456±0.0863 | 0.7327±0.1959 | 0.592981 | 0.595723 |

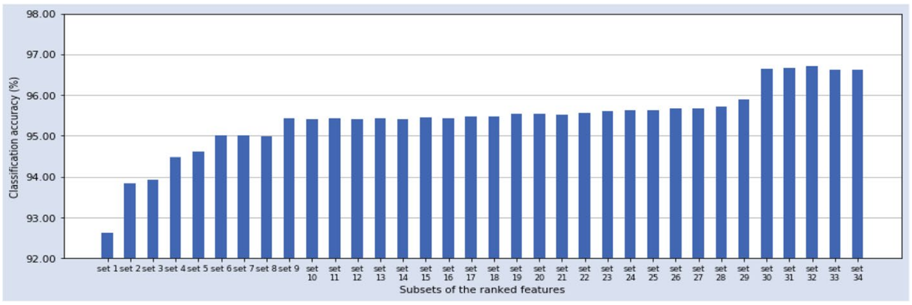


**Fig. 9** Classification accuracy of the different subsets of the ranked features on PAN 2013 test set by chi-square

not discriminative as shown in Table 6. Therefore, the proposed system takes into consideration all the different types of the sentence similarity features by creating the supervised dataset to train SVM classification algorithm, which has the ability to compute the hyperplane equation of the 34 features to distinguish the two classes, rather than conducting extensive experiments as the previous researches to find the best weighting coefficient values for incorporating features.

The proposed system also takes into consideration the determination of most effective features that have the ability to discriminate the suspicious cases and differentiate the variations of the text similarities with the highest accuracy. As shown in Table 6, there is a challenge to find the best subset of the features that discriminate the similarity cases. The proposed system is based on the filter feature selection approach using Chi-square algorithm to rank the 34 features, and generate 34 subset of features. The first subset is included with the first feature of the ranked features, and each subset of the ranked features consists of the features of previous subset plus the next feature in the list of the ranked features. The classification accuracy value was calculated for each generated subset, the subset of the ranked features that achieved the highest classification accuracy is extracted to be the most discriminative subset of the features.

Two experiments were developed to rank and select the most effective sentences similarity features. The first experiment was conducted on the documents of PAN 2013 test

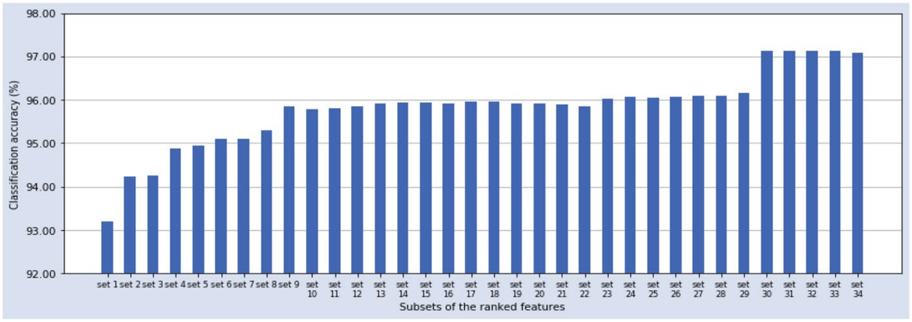


**Fig. 10** Classification accuracy of the different subsets of the ranked features on PAN 2014 test set by chi-square.

**Table 7** Comparison results of the proposed system and other relevant systems for random obfuscation on PAN 2013 test corpus

| Team | PlagDet (%) | F-measure (%) | Recall (%) | Precision (%) | Granularity |
|---|---|---|---|---|---|
| Sanchez-Perez et al. [54] | **88.42** | **88.48** | 86.07 | 91.02 | 1.00086 |
| Proposed system | 88.33 | 88.44 | **86.72** | 90.22 | 1.00171 |
| PlagLinSVM [3] | 88.27 | 88.27 | 84.72 | **92.13** | 1.00000 |
| PlagRbfSVM [3] | 87.32 | 87.32 | 85.17 | 89.58 | 1.00000 |
| Oberreuter and Eiselt [36] | 86.78 | 86.77 | 83.25 | 90.61 | 1.00000 |
| Shrestha et al. [59] | 86.56 | 86.95 | 83.16 | 91.10 | 1.00630 |
| Palkovskii and Belov [40] | 86.50 | 86.60 | 82.24 | 91.45 | 1.00176 |
| Vani and Gupta [68] | 83.65 | 83.71 | 79.92 | 87.88 | 1.00100 |
| Kong et al. 1 [23] | 83.24 | 83.24 | 77.90 | 89.37 | 1.00000 |
| Kong et al. 2 [28] | 82.30 | 82.30 | 78.08 | 87.00 | 1.00000 |
| Kong et al. 3 [24] | 82.28 | 82.28 | 78.68 | 86.22 | 1.00000 |
| Glinos [15] | 80.62 | 82.95 | 72.48 | 96.95 | 1.04037 |
| Gross and Modaresi [15] | 80.29 | 82.21 | 71.88 | 96.00 | 1.03336 |
| Palkovskii and Belov [38] | 79.69 | 79.69 | 75.13 | 84.84 | 1.00000 |
| Rodríguez Torrejón and Martín Ramos [50] | 75.38 | 75.38 | 62.99 | 93.84 | 1.00000 |
| Suchomel et al. [60] | 75.28 | 75.28 | 68.89 | 82.97 | 1.00000 |
| Oberreuter et al. [37] | 74.96 | 74.95 | 65.32 | 87.92 | 1.00000 |
| Rodríguez Torrejón and Martín Ramos [51] | 74.71 | 74.71 | 63.37 | 91.00 | 1.00000 |
| Gharavi et al. [13] | 73.99 | 73.99 | 72.90 | 75.10 | 1.00000 |
| Daud et al. [9] | 71.86 | 71.86 | 64.12 | 81.76 | 1.00000 |
| Rodríguez Torrejón and Martín Ramos [63] | 70.15 | 70.15 | 60.28 | 83.88 | 1.00000 |
| Shrestha and Solorio [58] | 66.71 | 80.57 | 71.46 | 92.34 | 1.30962 |
| Saremi and Yaghmaee [56] | 65.67 | 78.71 | 68.88 | 91.81 | 1.29511 |
| Suchomel et al. [61] | 65.21 | 65.22 | 51.95 | 87.58 | 1.00000 |
| Küppers and Conrad [25] | 51.60 | 52.29 | 36.87 | 89.89 | 1.01847 |
| Alvi et al. [4] | 50.25 | 52.81 | 36.60 | 94.79 | 1.07203 |
| Palkovskii and Belov [39] | 49.96 | 52.36 | 36.42 | 93.14 | 1.06785 |
| Abnar et al. [1] | 49.06 | 49.59 | 35.36 | 82.99 | 1.01509 |
| Sánchez-Vega et al. [55] | 45.60 | 46.32 | 43.50 | 49.52 | 1.02200 |
| Nourian [35] | 35.08 | 37.92 | 23.61 | 96.27 | 1.11558 |
| Jayapal and Goswami [20] | 18.15 | 30.38 | 18.18 | 92.31 | 2.19096 |

set, and the second experiment was developed on the documents of PAN 2014 test set to verify the results. As shown in Figs. 9 and 10, the subset No. 32 that includes all the ranked features except the last two features, achieved the highest classification accuracy in the two experiments. Therefore, it was selected to be the best subset of the sentences similarity features.

Three experiments were also conducted to evaluate the proposed system comparing with the recent approaches. These experiments were implemented using the documents of

**Table 8** Comparison results of the proposed system and other relevant systems on the complete PAN 2013 test corpus

| Team | PlagDet (%) | F-measure (%) | Recall (%) | Precision (%) | Granularity |
|---|---|---|---|---|---|
| Proposed system | **89.12** | **89.34** | 86.56 | 92.32 | 1.00349 |
| Sanchez-Perez et al. [54] | 87.82 | 88.03 | **87.90** | 88.17 | 1.00344 |
| Oberreuter and Eiselt [36] | 86.93 | 87.17 | 85.78 | 88.60 | 1.00369 |
| Palkovskii and Belov [40] | 86.81 | 87.17 | 82.64 | 92.23 | 1.00580 |
| PlagLinSVM [3] | 86.45 | 86.83 | 85.01 | 88.72 | 1.00609 |
| Glinos [15] | 85.93 | 86.97 | 79.33 | **96.25** | 1.01695 |
| PlagRbfSVM [3] | 84.95 | 85.43 | 86.00 | 84.87 | 1.00791 |
| Rodríguez Torrejón and Martín Ramos [63] | 82.95 | 83.12 | 76.90 | 90.43 | 1.00278 |
| Shrestha et al. [59] | 84.40 | 84.83 | 83.78 | 85.91 | 1.00701 |
| Gross and Modaresi [16] | 82.64 | 84.13 | 76.62 | 93.27 | 1.02514 |
| Rodríguez Torrejón and Martín Ramos [51] | 82.22 | 82.30 | 76.19 | 89.48 | 1.00141 |
| Kong et al. 1 [23] | 82.16 | 82.35 | 80.75 | 84.01 | 1.00309 |
| Kong et al. 2 [28] | 81.90 | 82.09 | 81.34 | 82.86 | 1.00336 |
| Gharavi et al. [13] | 79.90 | 79.90 | 76.71 | 83.36 | 1.00000 |
| Suchomel et al. [60] | 74.48 | 74.49 | 76.59 | 72.51 | 1.00028 |
| Saremi and Yaghmaee [56] | 69.91 | 81.55 | 77.12 | 86.51 | 1.24450 |
| Shrestha and Solorio [58] | 69.55 | 80.06 | 73.81 | 87.46 | 1.22084 |
| Abnar et al. [1] | 67.22 | 68.30 | 61.16 | 77.33 | 1.02245 |
| Alvi et al. [4] | 65.95 | 69.28 | 55.07 | 93.38 | 1.07111 |
| Palkovskii and Belov [39] | 61.52 | 64.70 | 53.56 | 81.70 | 1.07295 |
| Nourian [35] | 57.72 | 59.50 | 43.38 | 94.71 | 1.04343 |
| Gillam [27] | 40.06 | 40.06 | 25.89 | 88.49 | 1.00000 |
| Gillam and Notley [14] | 28.30 | 28.30 | 16.84 | 88.63 | 1.00000 |
| Jayapal and Goswami [20] | 27.08 | 53.25 | 38.19 | 87.90 | 2.90698 |

PAN 2013 and PAN 2014. Tables 7, 8 and 9 show the performance results of the proposed system. The results show that the proposed system achieved the best recall score and the second score for f-measure and plagdet values in the random obfuscation sub-corpora of PAN 2013. The performance results also show that the proposed system outperformed all the other state-of-the-art systems on the all documents of PAN 2013 and PAN 2014 corpus.

From the results shown in Tables 8 and 9, it can be seen that the most of the previous systems achieved a varied rank into the different datasets, this change depends on the structure of the dataset and the types of its plagiarism. On the other hand, the proposed system maintained its rank and superiority in the performance with the different datasets. So based on these results, the proposed system achieves the efficiency and robustness to detect the different forms of the text plagiarism. These also indicate the ability of the support vector machine algorithm to find the hyperplane equation of the selected 32 features to detect the different types of text similarities. Additionally, the utilization of the recursive extension algorithm led to improve the recall and granularity scores without the false impact of precision. The adaptive behavior of the proposed system is also affected on improving Plagdet

**Table 9** Comparison results of the proposed system and other relevant systems on the complete PAN 2014 test corpus

| Team | PlagDet (%) | F-measure (%) | Recall (%) | Precision (%) | Granularity |
|---|---|---|---|---|---|
| Proposed system | **92.91** | **92.95** | 90.14 | 95.94 | 1.00053 |
| Palkovskii and Belov [40] | 90.78 | 90.80 | 88.92 | 92.76 | 1.00027 |
| PlagLinSVM [3] | 90.01 | 90.15 | 90.55 | 89.75 | 1.00210 |
| Oberreuter and Eiselt [36] | 89.27 | 89.30 | 91.54 | 87.17 | 1.00051 |
| Sanchez-Perez et al. [54] | 89.20 | 89.21 | **91.98** | 86.61 | 1.00026 |
| Glinos [15] | 88.77 | 89.89 | 84.51 | **96.01** | 1.01761 |
| PlagRbfSVM [3] | 88.27 | 88.40 | 91.49 | 85.52 | 1.00209 |
| Shrestha et al. [59] | 86.81 | 87.05 | 89.84 | 84.42 | 1.00381 |
| Gross and Modaresi [16] | 85.50 | 86.84 | 81.82 | 92.52 | 1.02187 |
| Rodríguez Torrejón and Martín Ramos [63] | 84.87 | 84.87 | 80.27 | 90.03 | 1.00000 |
| Kong et al. [24] | 83.51 | 83.52 | 84.16 | 82.88 | 1.00000 |
| Alvi et al. [4] | 73.42 | 77.03 | 67.28 | 90.08 | 1.06943 |
| Abnar et al. [1] | 66.38 | 66.59 | 84.78 | 54.83 | 1.00455 |
| Gillam and Notley [14] | 44.08 | 44.07 | 29.66 | 85.74 | 1.00000 |

score in the summary sub-corpus without negative influence of the recall value in the no-obfuscation sub-corpus.

## 5 Conclusions and future work

In this paper, a new system is proposed to detect all the different types of text plagiarism including the lexical, syntactic, and semantic cases. It is based on two paths to detect the sentences similarity cases, the first path is based on traditional paragraph-level comparison, and the second path is based on the hyperplane equation of the constructed SVM classifier. Training database of SVM algorithm is created including 34 sentence similarity features. Statistical analysis of the constructed training database is developed, which indicated that the 34 sentence similarity features have closer positive and negative mean values, and values of 95% confidence limits of the positive cases are closer to the positive and negative mean values, which indicted that each feature is not discriminative to distinguish the similarity cases. Two experiments were developed to rank the sentences similarity features and select the most effective subset of these features. The first experiment was conducted on the documents of PAN 2013 test set, and the second experiment was developed on the documents of PAN 2014 test set to verify the results. The results showed that all the ranked features except the last two features, achieved the highest classification accuracy in the two experiments. Therefore, the ranked 32 features were selected to be the best subset of the sentences similarity features. SVM classification algorithm was used to fit the training dataset values, which has the ability to compute the hyperplane equation of the selected 32 features and add new dimensionality to distinguish the overlapping between the training cases of different classes.

Three experiments were also conducted to evaluate the proposed system comparing with the recent approaches. The results showed that the proposed system achieved the best scores 88.42%, 89.12% and 92.91% of Plagdet metric and 88.48%, 89.34% and 92.95% of F-measure metric on the test documents of random obfuscation in PAN 2013 dataset, and all documents of the different plagiarism types in PAN 2013 and PAN 2013 datasets, respectively. These results indicated that the proposed system outperformed all the other state-of-the-art systems.

In the future work, we plan to create a training database that containing a variation in the level of lexical, syntactic and semantic similarity cases. The constructed SVM classifier can be more accurate and robust, if it will be trained on cases with confusion similarities that have a difficulty to detect their plagiarism, rather than the extracted cases from PAN 2012 and PAN 2013 databases using meteor score and shared unigrams. We also plan to use Meta-heuristic algorithms to detect the best values of the threshold m and t parameters that used in the path of sentence level comparison of the seeding phase. The proposed system cannot be able to detect the text plagiarism between the documents that have different languages, so we plan to modify it by adding an extra phase to translate the documents into English language to detect the text plagiarism between the documents that written by different languages. Moreover, we also plan to apply audio recognize phase to convert it into sentences. After extracting the sentences from the audio, the proposed system can be applied using the recognized sentences to detect the audio plagiarism in videos.

**Funding** Open access funding provided by The Science, Technology & Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB).

## Declarations

**Conflict of interest** The authors have no affiliation with any organization with a direct or indirect financial interest in the subject matter discussed in the manuscript.

## References

1. Abnar S, Dehghani M, Zamani H, Shakery A (2014) Expanded N-Grams for Semantic Text Alignment. In: CLEF (working notes) 1180:928-938. Available: http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-AbnarEt2014.pdf
2. Ahuja V, Gupta R. Kumar (2020) A New Hybrid Technique for Detection of Plagiarism from Text Documents. Arab J Scie Eng 45(12):9939–9952. https://doi.org/10.1007/s13369-020-04565-9
3. Altheneyan AS, El BachirMenai M (2020) Automatic plagiarism detection in obfuscated text. Pattern Anal Applic 23(4):1627–1650. https://doi.org/10.1007/s10044-020-00882-9
4. Alvi F, Stevenson M, Clough P (2014) Hashing and Merging Heuristics for Text Reuse Detection. In: CLEF (working notes) 1180:939-946. Available: http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-AlviEt2014.pdf
5. Alvi Faisal, Stevenson Mark, Clough Paul (2021) Paraphrase type identification for plagiarism detection using contexts and word embeddings. Int J Educ Technol Higher Educ 18(1):1–25. https://doi.org/10.1186/s41239-021-00277-8

6. Bochkarev VV, Shevlyakova AV, Solovyev VD (2015) The average word length dynamics as an indicator of cultural changes in society. Soc Evol Hist 14(2):153–175
7. Chang Chia-Yang et al (2021) Using word semantic concepts for plagiarism detection in text documents. Inform Retrieval J 24(4):298–321. https://doi.org/10.1007/s10791-021-09394-4
8. Craig Causer (2011) The Way Ahead. IEEE Potentials 30(4):3–3. https://doi.org/10.1109/MPOT.2011.942130
9. Daud A, Khan JA, Nasir JA, Abbasi RA, Aljohani NR, Alowibdi JS (2018) Latent dirichlet allocation and POS tags based method for external plagiarism detection: LDA and POS tags based plagiarism detection. Int J Semantic Web Inform Syst 14(3):53–69. https://doi.org/10.4018/IJSWIS.2018070103
10. Eissen SMZ, Stein B (2006) Intrinsic plagiarism detection. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 3936:565-569.https://doi.org/10.1007/11735106_66
11. Ekbal A, Saha S, Choudhary G (2012) Plagiarism detection in text using vector space model. In: 2012 12th international conference on hybrid intelligent systems (HIS) IEEE, pp. 366-371.https://doi.org/10.1109/HIS.2012.6421362
12. Ghanem Bilal et al (2018) HYPLAG Hybrid Arabic text plagiarism detection system. International conference on applications of natural language to information systems. Springer, Cham, pp 315–323. https://doi.org/10.1007/978-3-319-91947-8_33
13. Gharavi E, Veisi H, Rosso P (2020) Scalable and language-independent embedding-based approach for plagiarism detection considering obfuscation type: no training phase. Neural Comput Applic 32(14):10593–10607. https://doi.org/10.1007/s00521-019-04594-y
14. Gillam L, Notley S (2014) Evaluating robustness for 'IPCRESS': Surrey's text alignment for plagiarism detection. In: CLEF (working notes) 1180:951-957. Available: http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-GillamEt2014.pdf
15. Glinos DG (2014) A hybrid architecture for plagiarism detection. In: CLEF (working notes) 1180:958-965. Available: http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-Glinos2014.pdf
16. Gross P, Modaresi P (2014) Plagiarism alignment detection by merging context seeds. In: CLEF (working notes) 1182:966-972. Available: https://pan.webis.de/downloads/publications/papers/gross_2014.pdf
17. Gupta D, Vani K, Singh CK (2014) Using Natural Language Processing techniques and fuzzy-semantic similarity for automatic external plagiarism detection. In: 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp 2694-2699. https://doi.org/10.1109/ICACCI.2014.6968314
18. Harris MA et al (2004) The Gene Ontology (GO) database and informatics resource. Nucl Acids Res 32:258–261. https://doi.org/10.1093/nar/gkh036
19. Jaccard P (1912) THE distribution of the flora in the alpine zone. New Phytologist 11(2):37–50. https://doi.org/10.1111/j.1469-8137.1912.tb05611.x
20. Jayapal A, Goswami B (2013) Vector Space Model and Overlap Metric for Author Identification. In: CLEF (working notes). Available: http://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-JayapalEt2013.pdf
21. JJiang JJ, Conrath DW (1997) Semantic similarity based on corpus statistics and lexical taxonomy. In: Proceedings of the 10th Research on Computational Linguistics International Conference, pp 19-33
22. Kauffman Yashu, Young Michael F (2015) Digital plagiarism: An experimental study of the effect of instructional goals and copy-and-paste affordance. Comput Educ 83:44–56. https://doi.org/10.1016/j.compedu.2014.12.016
23. Kong L, Qi H, Wang S, Du C, Wang S and Han Y (2012) Approaches for candidate document retrieval and detailed comparison of plagiarism detection. CLEF (working notes). Available: http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-LeileiEt2012.pdf
24. Kong L, Han Y, Han Z, Yu H, Wang Q, Zhang T, Qi H (2014) Source Retrieval Based on Learning to Rank and Text Alignment Based on Plagiarism Type Recognition for Plagiarism Detection. CLEF (working notes) 1180: 973-976. Available: http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-KongEt2014.pdf
25. Küppers R, Conrad S (2012) A set-based approach to plagiarism detection. CLEF (working notes). Available: http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-KuppersEt2012.pdf
26. Leacock C, Chodorow M (1998) Combining Local Context and WordNet Similarity for Word Sense Identification. In: WordNet: An electronic lexical database 49(2):265-283. https://doi.org/10.7551/mitpress/7287.003.0018
27. Lee G (2013) Guess again and see if they line up: Surrey's runs at plagiarism detection. In: CLEF (working notes) 1179. Available: http://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-Gillam2013.pdf

28. Leilei K, Haoliang Q, Cuixia D, Mingxing W, Han Z (2013) Approaches for source retrieval and text alignment of plagiarism detection. conference and labs of the evaluation forum and workshop (CLEF'13). Available: http://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-LeileiEt2013.pdf
29. Li Y, McLean D, Bandar ZA, O'Shea JD, Crockett K (2006) Sentence similarity based on semantic nets and corpus statistics. IEEE Trans Knowl Data Eng 18(8):1138–1150. https://doi.org/10.1109/TKDE.2006.130
30. Lin, D (1998) An information-theoretic definition of similarity. In: Icml 98(1998):296-304
31. Lyon C, Malcolm J, Dickerson B (2001) Detecting short passages of similar text in large document collections. In: Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing (EMNLP 2001), pp. 118-125
32. Larock Margaret, Jacob Tressler, and Claude Lewis (1980) Mastering effective English. Copp Clark Pitman, Mississauga
33. Mariani J, Francopoulo G, Paroubek P (2018) Reuse and plagiarism in Speech and Natural Language Processing publications. Int J Digital Libr 19:2–3. https://doi.org/10.1007/s00799-017-0211-0
34. Miller George A (1995) WordNet: a lexical database for English. Commun ACM 38:39–41. https://doi.org/10.1145/219717.219748
35. Nourian A (2013) Submission to the 5th international competition on plagiarism detection. Available: http://www.uni-weimar.de/medien/webis /events/pan-13
36. Oberreuter G, Eiselt A (2014) Submission to the 6th international competition on plagiarism detection. Available: https://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/
37. Oberreuter G, Carrillo-Cisneros D, Scherson I, Velásquez J (2012) Submission to the 4th international competition on plagiarism detection. Available: http://www.uni-weima r.de/medie n/webis / event s/pan- 12
38. Palkovskii Y, Belov A (2012) Applying specific clusterization and fingerprint density distribution with genetic algorithm overall tuning in external plagiarism detection. In: CLEF (working notes). Available: http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-PalkovskiiEt2012.pdf
39. Palkovskii Y, Belov A (2013) Using hybrid similarity methods for plagiarism detection. CLEF (working notes). Available: http://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-PalkovskiiEt2013.pdf
40. Palkovskii Y, Belov A (2014) Developing high-resolution universal multi-type n-gram plagiarism detector. In: Conference and Labs of the Evaluation Forum and Workshop (CLEF'14) 1180:984-989. Available: http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-PalkovskiiEt2014.pdf
41 Pearson K (1900) X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. London, Edinburgh, Dublin Philos Mag J Sci 50(302):157–175. https://doi.org/10.1080/14786440009463897
42. Pedersen T, Patwardhan S, Michelizzi J (2004) WordNet: Similarity - Measuring the relatedness of concepts. AAAI 4:25–29
43. Phyllis A (2016) Spatial Data Transfer Standard (SDTS). In: Encyclopedia of GIS, pp 1-11. https://doi.org/10.1007/978-3-319-23519-6_1259-2
44. Potthast M, Gollub T, Hagen M, Graßegger J, Kiesel J, Michel M, Oberländer A, Tippmann M, Barrón-Cedeño A, Gupta P, Rosso P, Stein B (2012) Overview of the 4th international competition on plagiarism detection
45. Potthast M, Gollub T, Hagen M, Tippmann M, Kiesel J, Rosso P, Stamatatos E, Stein B (2013) Overview of the 5th international competition on plagiarism detection. In: Forner P, Navigli R, Tufs D (eds) Working notes papers of the CLEF 2013 evaluation labs, pp. 301–33
46. Potthast M, Stein B, Barrón-Cedeño A, Rosso P (2010) An evaluation framework for plagiarism detection. In Coling 2010: Posters, pp 997-1005
47. Potthast M, Hagen M, Beyer A, Busse M, Tippmann M, Rosso P, Stein B (2014) Overview of the 6th international competition on plagiarism detection. In: Cappellato L, Ferro N, Halvey M, Kraaij W (eds) Working notes papers of the CLEF 2014 evaluation labs, CLEF and CEUR-WS.org, CEUR workshop proceedings, pp 845–876
48. Reshamwala A, Mishra D, Pawar P (2013) Review on natural language processing. IRACST Eng Sci Technol: An Int J (ESTIJ) 3(1):113–116
49. Resnik P (1995) Using information content to evaluate semantic similarity in a taxonomy. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, Montreal 1:448-453
50. Rodríguez Torrejón D, Martín Ramos J (2014) CoReMo 2.3 Plagiarism Detector Text Alignment Module. In: CLEF (working notes) 1180:997-1003. Available: http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-RodriguezTorrejonEt2014.pdf

51. Rodríguez Torrejón DA, Ramos JMM (2013) Text Alignment Module in CoReMo 2.1 Plagiarism Detector. CLEF (working notes). Available: http://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-RodriguezTorrejonEt2013.pdf
52. Roostaee M, Fakhrahmad SM, Sadreddini MH (2020) Cross-language text alignment: A proposed two-level matching scheme for plagiarism detection. Expert Syst Applic 160:113718. https://doi.org/10.1016/j.eswa.2020.113718
53. Sahi M, Gupta V (2017) A Novel Technique for Detecting Plagiarism in Documents Exploiting Information Sources. Cogn Comput 9(6):852–867. https://doi.org/10.1007/s12559-017-9502-4
54. Sanchez-Perez M, Sidorov G, Gelbukh A (2014) A winning approach to text alignment for text reuse detection at PAN 2014– notebook for PAN at CLEF. In: Cappellato L, Ferro N, Halvey M, Kraaij W (eds) CLEF 2014 evaluation labs and workshop-working notes papers, 15–18 September, CEUR-WS.org, Shefeld, 1180:1004–1011. http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-SanchezPerezEt2014.pdf
55. Sánchez-Vega F, Montes-y-Gómez M, Pineda LV(2012) Optimized fuzzy text alignment for plagiarism detection In: CLEF (working notes). Available: http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-SanchezVegaEt2012.pdf
56. Saremi M, Yaghmaee F (2013) Submission to the 5th international competition on plagiarism detection. Available: http://www.uni-weima r.de/medie n/webis /event s/pan-13
57. Shahmohammadi H, Dezfoulian MH, Mansoorizadeh M (2021) Paraphrase detection using LSTM networks and handcrafted features. Multimedia Tools Applic 80(4):6479–6492. https://doi.org/10.1007/s11042-020-09996-y
58. Shrestha P, Solorio T (2013) Using a variety of n-grams for the detection of different kinds of plagiarism. CLEF (working notes). Available: http://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-ShresthaEt2013.pdf
59. Shrestha P, Maharjan S, Solorio T (2014) Machine translation evaluation metric for text alignment: Notebook for PAN at CLEF 2014. In: CEUR Workshop Proceedings 1180:1012-1016. Available: https://pan.webis.de/downloads/publications/papers/shrestha_2014.pdf
60. Suchomel Š, Kasprzak J, Brandejs M (2013) Diverse queries and feature type selection for plagiarism discovery. In: CLEF (working notes). Available: http://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-SuchomelEt2013.pdf
61. Suchomel Š, Kasprzak J, Brandejs M (2012) Three way search engine queries with multi-feature document comparison for plagiarism detection. In: CLEF (working notes). Available: http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-SuchomelEt2012.pdf
62. Tomasic A, Garcia-Molina H (1993) Query processing and inverted indices in shared-nothing text document information retrieval systems. VLDB J 2(3):243–275. https://doi.org/10.1007/BF01228671
63. Torrejón DA, Ramos JMM (2012) Detailed Comparison Module In CoReMo 1.9 Plagiarism Detector. In: CLEF (working notes). Available: http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-RodriguezTorrejonEt2012.pdf
64. Ullah F, Wang J, Farhan M, Jabbar S, Wu Z, Khalid S (2020) Plagiarism detection in students' programming assignments based on semantics: multimedia e-learning based smart assessment methodology. Multimedia Tools Applic 79(13):8581–8598. https://doi.org/10.1007/s11042-018-5827-6
65. Vani K, Gupta D (2015) Investigating the impact of combined similarity metrics and POS tagging in extrinsic text plagiarism detection system. In: 2015 international conference on advances in computing, communications and informatics (ICACCI), pp 1578-1584. https://doi.org/10.1109/ICACCI.2015.7275838.
66. Vani K, Gupta D (2014) Using K-means cluster based techniques in external plagiarism detection. In: 2014 international conference on contemporary computing and informatics (IC3I), pp 1268-1273. https://doi.org/10.1109/IC3I.2014.7019659
67. Vani K, Gupta D (2017) Detection of idea plagiarism using syntax–Semantic concept extractions with genetic algorithm. Expert Syst Applic 73:11–26. https://doi.org/10.1016/j.eswa.2016.12.022
68. Vani K, Gupta D (2018) Unmasking text plagiarism using syntactic-semantic based natural language processing techniques: Comparisons, analysis and challenges. Inform Process Manag 54(3):408–432. https://doi.org/10.1016/j.ipm.2018.01.008
69. Wu Z, Palmer M (1994) Verbs semantics and lexical selection. In: Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics, pp 33-38. https://doi.org/10.3115/981732.981751
70. Zobel J, Moffat A (1998) Exploring the similarity space. In: Acm Sigir Forum, New York 32(1):18-34. https://doi.org/10.1145/281250.281256