Neural Computing and Applications (2022) 34:18837–18858
https://doi.org/10.1007/s00521-022-07486-w

ORIGINAL ARTICLE

Reliable plagiarism detection system based on deep learning approaches

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

Received: 28 February 2022 / Accepted: 26 May 2022 / Published online: 24 June 2022

Abstract
The phenomenon of scientific burglary has seen a significant increase recently due to the technological development in software. Therefore, many types of research have been developed to address this phenomenon. However, detecting lexical, syntactic, and semantic text plagiarism remains to be a challenge. Thus, in this study, we have computed and recorded all the features that reflect different types of text similarities in a new database. The created database is proposed for intelligent learning to solve text plagiarism detection problems. Using the created database, a reliable plagiarism detection system is also proposed, which depends on intelligent deep learning. Different approaches to deep learning, such as convolution and recurrent neural network architectures, were considered during the construction of this system. A comparative study was implemented to evaluate the proposed intelligent system on the two benchmark datasets: PAN 2013 and PAN 2014 of the PAN Workshop series. The experimental results showed that the proposed system based on long short-term memory (LSTM) achieved the first rank compared to up-to-date ranking systems.

Keywords Text plagiarism · Natural language processing · Deep learning · Convolution neural network · Recurrent neural network · LSTM

1 Introduction

Despite the importance of Internet use, the massive availability, accessibility, and rapid growth of information on the Internet have brought about benefits and drawbacks. Recently, information is exploited on the Internet and unethically attributed to other authors as a source of development and knowledge. This act is considered a crime and a direct threat to the research community and education. This act is called plagiarism, which, according to the Writing Program Board of Directors (WPA), is defined as "In an instructional setting, plagiarism occurs when a writer deliberately uses someone else's language, ideas, or another original (not common-knowledge) material without acknowledging its source" [1].

Text plagiarism can be classified to direct plagiarism or various degrees of obfuscation. Copy–paste is a direct plagiarism type that involves copying text from an original document and pasting it into a suspicious document. The other degrees of obfuscation vary according to the text variations types, such as changing the structure of the text, replacing words with their synonyms, and using both aspects together. These variations can be made automatically by using paraphrasing software tools or manually.

To overcome the phenomenon of text plagiarism, researchers have developed several approaches that rely on text linguistic analysis, which depends on its semantic features to calculate the similarity. Some approaches are established on n-grams features [2, 3], while the others are established on converting the texts to vectors using Vector Space Model (VSM) representation [4, 5], Wordnet based-knowledge [6, 7], several pre-trained methods such as

✉ 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

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

[1] Department of Computer Science and Engineering, Faculty of Electronic Engineering, Menoufia University, Menouf, Egypt

Word2Vec [8], InferSent [9], Glove [10], etc. These vectors are used as an input for cosine, Dice, Jaccard, Fuzzy, Match, or Euclidean resemblance measurements. The main weakness of these approaches is that many experiments are required to find the best threshold values for classifying cases. In addition, these approaches fail to detect the cases with the highest degrees of obfuscation. Therefore, to solve these problems, an intelligent deep learning system has been developed depending on statistics [11] and deep learning approaches [12].

Machine learning algorithms are now used in scientific computing, as well as in data and text mining [13–17]. Recent research found that it is important to speed up and improve the performance of the models by involving neural network models in the system. The key reasons for the superiority of deep learning over traditional machine learning are determining decision boundaries and feature engineering. Therefore, our proposed system depends on deep learning architectures: densely connected convolutional network and long short-term memory to determine the most convenient model that can fit the training data.

The key contributions of this research are summed up as follows:

- Creating a new text similarity feature (TSF) database to learn the deep learning models for detecting text plagiarism. This database was created by considering all the similarity features that reflect different types of lexical, syntactic, and semantic text aspects. Each row of this database was contained by a similarity case, represented in a vector of 42 values of the words and sentences similarity criteria. In sum, this database can be used for lexical, syntactic, and semantic text similarity research purposes.
- Preprocessing the created database to be appropriate with different deep learning models: Each one-dimensional (1D) vector of the database was converted to a three-dimensional (3D) image for the convolution neural network models and a signal for the recurrent neural network models.
- Proposing two classifiers for detecting different types of text plagiarism. The first classifier is based on densely connected convolutional network (DenseNet), while the second is based on long short-term memory (LSTM).
- Conducting a comparative study to find the most convenient model to fit the training data. This study was developed between the proposed classifiers based on different deep learning models and the traditional machine learning models.

The proposed system comprises three steps: preprocessing, detailed analysis, and post-processing. Firstly, according to basic natural language preprocessing, the suspicious and original documents are prepared into sentence and passage levels. Secondly, detailed analysis is responsible for extracting plagiarized cases by applying techniques like common n-grams, meteor scores, and intelligent deep learning classification. Finally, post-processing is applied to find the best largest plagiarized segment by solving the overlapping issue, merging adjacent cases, and removing small cases.

A new database of the lexical, syntactic, and semantic text similarity is created for the deep learning approaches, having 42 features for each similarity case. The constructed features' values are computed based on the similarity metrics of words and sentences from two benchmark datasets, that is, PAN 2013 [18] and PAN 2014 [19]. The constructed database trains the proposed system, and it is evaluated using the recall, precision, F-measure, granularity, and Plagdet measures. The performance of the proposed system based on LSTM has the first rank in PAN 2013 and PAN 2014 compared to the state-of-the-art systems.

The rest of the paper is organized as follows. Section 2 describes recent works on plagiarism detection. Section 3 explains in detail the proposed system. Section 4 presents the experimental setup and the obtained results. Finally, Sect. 5 presents the conclusions and some future work directions.

2 Related work

The phenomenon of plagiarism mainly using the Internet has attracted many researchers to develop efficient systems for its detection. However, the main challenge is the diversity of complexity degrees of plagiarism, whether at the lexical, syntactic, or semantic levels. In this section, we explain recent plagiarism detection techniques. Many researchers have used different statistical techniques and methodologies to detect plagiarism. Sánchez Vega et al. [3] introduced an automatic paraphrase plagiarism identification approach based on character-level features. Their key contribution was exploring text by content and style features according to n-grams to detect similarity fragments. The main drawback of this method is that it neglects the semantic aspect.

Sanchez-Perez et al. [4] proposed a system for detecting text alignment between the suspicious and source documents. Their core contributions are the usage of tf-isf (term frequency-inverse sentence frequency) method between the sentences to extract the plagiarized cases; a recursive algorithm was also proposed to combine adjacent cases to configure the largest plagiarized fragments and solving the overlap issues. Additionally, adaptive behavior was developed to adjust the parameters depending on the type of plagiarism. This system lacks the capability to integrate the linguistic aspect into the analysis. In addition, it had not a method to determine the type of plagiarism that being dealt with it.

Meysam et al. [5] suggested a two-level matching strategy to properly align similarity fragments from the source and suspicious documents. It is taken into consideration both syntactic and semantic components. The first level applied a vector space model depending on a local weighting approach and a multilingual word embedding based on dictionary to select the smallest collection of highly similar candidate fragment pairs. Then, at the second level, graph of word representations is used for pairwise comparisons. The primary limitation of this strategy is the recognition of short length cases.

Vishal et al. [7] introduced an approach based on extracting syntactic and semantic knowledge from text documents. This knowledge was calculated based on linguistic aspects: path similarity and depth estimation with different weights. In addition, the syntactic and the Dice measures were utilized as matrices similarity for determining syntactic and semantic knowledge between sentences. This approach did not use machine learning techniques and fails to detect complex cases such as manually, translated, and summarized. Gharavi et al. [11] developed a scalable and language-independent system for plagiarism detection based on text embedding with syntactic and semantic information for comparison. This comparison was made at the sentence level for extracting pairs of suspicious and source sentences with the highest similarity. This system proposed three methods for parameter tuning. The first method, known as offline threshold tuning, involves several tests on a training dataset with varying parameter values. The second method was called offline threshold tuning with obfuscation. The obfuscation type was treated as a special type in their analysis. The third method was called online threshold tuning; the parameters were changed from one type to another according to standard deviation or median absolute deviation for all Jaccard similarity values between each detected pair. The result shows that the obfuscation type still needs to be investigated further.

Altheneyan et al. [20] presented an automatic plagiarism detection system based on a support vector machine (SVM) classifier with lexical, syntactic, and semantic features. According to the kernel used, there were two prototypes: a linear kernel (PlagLinSVM) and a radius kernel (PlagRbfSVM). The implementation mainly occurred in two stages: paragraph and sentence. The paragraph stage identified the pairs of similar paragraphs between the suspicious and source documents using comparison depending on common unigram and bigram. Afterward, the sentencing stage detects the pairs of similar sentences based on common unigrams and meteor scores with the pre-defined condition. However, if the condition is not satisfied, the SVM classifier then determines whether the sentences are similar or not. Lastly, the adjacent cases were extended to form large plagiarism passages between the suspicious and the source documents. The drawback of this system is that it did not investigate several merging heuristics and deep learning techniques.

Nguyen et al. [12] designed a t-step plagiarism detection approach using an LSTM network model and feature extraction approach: a passage phase and a word phase. The passage phase extracted plagiarism passages according to the maximum passage similarity, intersection, and important features. The word phase identified the exact plagiarism strings relying on word and average word similarities and sentence-based similarity features. This system has some shortcomings in finding optimal parameters, sentential redundancy, word missing, and redundancy. Table 1 summarizes recent systems used for plagiarism detection.

3 Proposed system

In this manuscript, we present the proposed system to identify text plagiarism by extracting different linguistic characteristics of texts using the WordNet lexical database [21]. The target of the proposed system is to find text plagiarism with the best possible accuracy. These characteristics discover various aspects of plagiarism, such as changing the structure of the text, replacing words with their synonyms, and using both aspects together. The proposed system is mainly divided into three steps: preprocessing, detailed analysis, and post-processing. Figure 1 depicts these steps. The proposed system depends on creating a database to learn the deep learning models for detecting text plagiarism. It was constructed by considering all the similarity features that reflect different types of lexical, syntactic, and semantic text aspects. Each row of this database was contained by a similarity case, represented in a vector of 42 values of the words and sentences similarity criteria.

3.1 Preprocessing

Document text is one of the most unstructured forms of data; it contains noise in various forms such as special characters, punctuation, and text in different case. Machine learning approaches do not have also the ability to understand text data. Therefore, a preprocessing step is applied to clean the text and convert it into a more analyzable form so that the machine learning algorithms can better classify the text.

Table 1 An overview of the recent systems used for plagiarism detection

Authors	Techniques & methodologies	Datasets	Limitations
Sánchez Vega et al. [3]	Explore both content and style features according to n-grams	P4PIN	Neglects the semantic aspect
Sánchez-Perez et al. [4]	TF-ISF (Term Frequency-Inverse Sentence Frequency) with Dice and cosine measures	PAN 2013 PAN 2014	Lacks in integrating the linguistic aspect Not specifying the best parameters
Meysam et al. [5]	Vector Space Model with cosine metric Graph-of-words representations with match graph	PAN 2011 PAN 2012	Fails in recognition of short-length cases
Vishal et al. [7]	Wordnet based knowledge Path similarity Depth estimation Syntactic and the Dice measures	PAN 2011	Lack of deep learning techniques Fails to detect complex cases
Gharavi et al. [11]	Text embedding with syntactic and semantic information Statistics with Jaccard similarity	PAN 2013 PAN 2014	Fails to detect the high degrees of obfuscation cases
Altheneyan et al. [20]	Unigrams and bigrams Lexical, syntactic, and semantic features Support vector machine	PAN 2012 PAN 2013 PAN 2014	Deficiency in investigating merging heuristics Lack of deep learning techniques
Nguyen et al. [12]	Multi-layer LSTM network model Feature extraction Passage-phase Word-phase	PAN 2014	Not determining optimal parameters Sentential redundancy Word missing or redundancy

In this step, the input documents are prepared in two levels: sentence and passage levels. Sentence level is based on seven natural language preprocessing techniques: sentence segmentation, tokenization, lowercasing, removing stop words, removing special characters, removing irrelevant parts of speech tags, and lemmatization. The first technique, which is sentence segmentation, has been determined to be responsible for splitting documents into meaningful sentences. Secondly, tokenization converts each sentence to small units called unigrams. The third technique, lowercasing, reformats each token into lowercase form. The fourth technique removes stop words such as my, is, are, and the. The fifth technique removes special characters and numeric such as $, @, &,!. The sixth technique is used to prune irrelevant parts of speech tags (conjunction, preposition, articles, pronouns, prepositions, and determiners). Lastly, lemmatization obtains the basic form of the word through its context, where morphological analysis of the words is performed according to parts of speech.

Passage level was based on converting documents into meaningful passages using sentence segmentation technique and then merging adjacent sentences until the length of passage extended to at most 500 characters. After forming the passages, each passage was tokenized into unigram and bigram units; then, lowercasing and removing stop words, special characters, and numbers were implemented.

3.2 Detailed analysis

This step is used to discover plagiarized cases between the documents; it depended on three phases: passage, sentence, and intelligent classification. The first phase is applied to get the pairs of suspicious and source passages with the highest probability of plagiarism, which reduces the search area for plagiarized cases from whole documents. The second phase is based on n-grams and meteor score techniques to detect the plagiarized cases. The third phase is

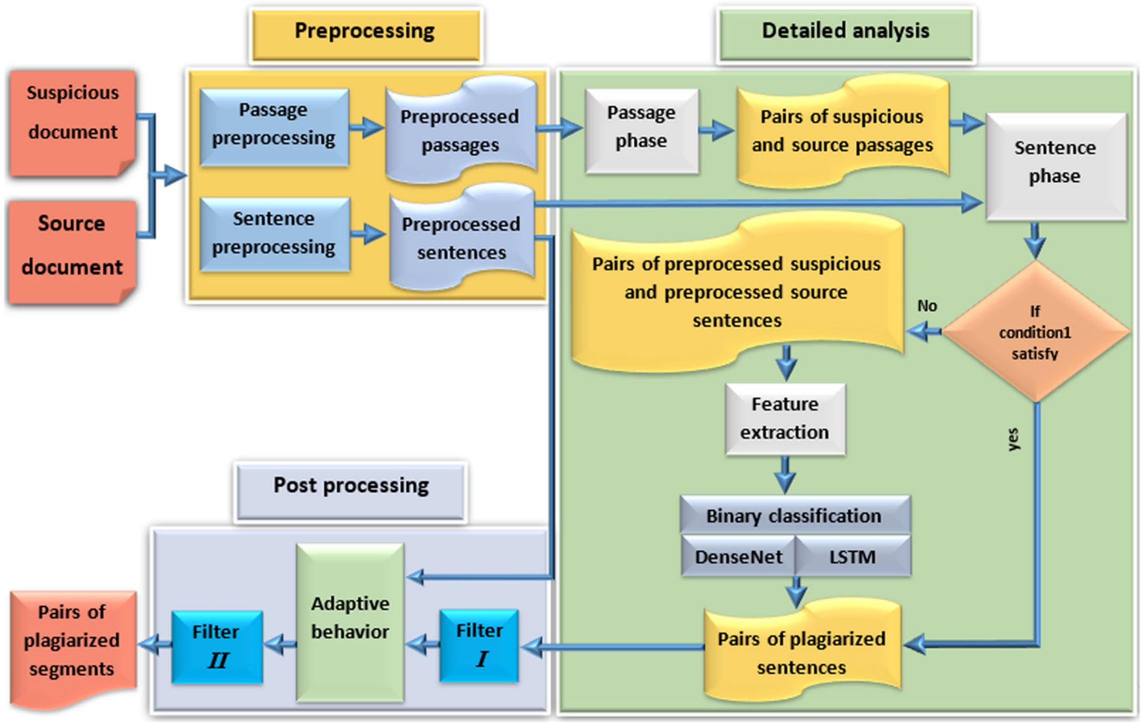

Fig. 1 Structure of the proposed system

developed to discover the complex plagiarized cases with a high level of obfuscation using deep learning approach.

Passage-phase In this phase, the goal is to get the pairs of suspicious and source passages with the highest probability of plagiarism. This goal is achieved by examining each suspected and source passage according to maximum common unigrams and bigrams [22]. Then, if plagiarism is applicable, it retrieves the source passage's prior and subsequent passages. The duplicate pairs are eliminated once the passage pairs have been extracted.

Sentence phase After reducing the search area for plagiarized cases from whole documents and limiting it to the passage level, the pairs of passages are analyzed for the plagiarized sentences. This comparison is based on maximum common unigrams and the meteor score [20] between the sentences. Two conditions are applied with two thresholds (th_{upper}, th_{low}). The conditioned output is compared with the meteor score to determine the type of case and if it requires a deeper analysis. For the first condition, if the meteor score value is more than or equal to th_{upper}, it is considered a plagiarized case. However, for the second condition, if the meteor score value is less than th_{low}, the case is discarded. Finally, if neither the first nor second conditions are fulfilled, the case is classified using the constructed deep algorithm.

Intelligent classification phase Complex cases with a high level of obfuscation can pass successfully undetected. The role of the intelligent classification phase is to train the deep learning model with the lexical, syntactic, and semantic features for each case. The deep learning models need a supervised database for the training process and constructing the deep classifier. This phase is divided into two steps: TSF database creation and binary deep classifier construction.

3.2.1 TSF Database creation

All the benchmark datasets used in the text similarity research were constructed with a set of sentences without the similarity features of the words and sentences. Therefore, in this paper, the TSF database was constructed to contain the features that have the ability to discriminate the suspicious cases and differentiate the variations in the text similarities. It was created by considering all the similar features that reflect lexical, syntactic, and semantic text similarity types. Each row of this database was consisted of a plagiarized case, represented in a vector of 42 values of the words and sentences similarity criteria. This database is beneficial for training the intelligent classification models to detect the text similarities, which have lexical, syntactic, and semantic text similarity features.

Positive and negative case extraction To create the supervised database, it must contain the supervised cases: positive and negative cases. Therefore, the pairs of the plagiarized passages were first extracted and formatted according to the sentence level preprocessing as mentioned in Sect. 3.1. Then, each suspicious sentence was scanned with all the source sentences upon the meteor score and common unigrams. The pair with the maximum meteor score and common unigrams was considered a positive case. While extracting the negative cases, the pairs of documents that excluded plagiarism were pre-processed using the sentence level mentioned in Sect. 3.1. After that, each suspicious sentence was checked with all source sentences based on common unigrams. If equal to 1, it is considered a negative case.

Features computation The characteristics that can classify the cases were computed in this step. These characteristics were mainly explored using their alternative meanings and the order of words between the sentences for each case. Each row of the TSF database was contained by a similarity case, represented in a vector of 42 values of the sentences' similarity criteria. The similarity features of the training database (TSF database) were based on fuzzy and gross criteria of sentence similarity. Additionally, they contained five different criteria of the similarity sentences. Each of which was computed for eight different criteria [6, 7, 23–27] for the word similarity. The computed word and sentence similarity criteria are explained as follows:

Path similarity criteria [6]**:** It returns a value indicating the degree of similarity between two-word senses.

$$Crit_{PA}(m_l, m_r) = \frac{1}{d+1} \tag{1}$$

where m_l and m_r are word senses and d is the shortest path distance.

Depth estimation similarity criteria [6]**:** It uses the difference between the summation of depths and least common subsumer.

$$Crit_{DE}(m_l, m_r) = e^{-(H(m_l)+H(m_r)-2*H(L(m_l,m_r)))} \tag{2}$$

$H(m)$ indicates the maximum depth of word sense, while m and $L(m_l, m_r)$ are the least common subsumer.

Hybrid path and depth estimation similarity criteria [7]**:** It depends on integrating both path and depth estimation similarity criteria by a specific weight δ.

$$Crit_H(m_l, m_r) = \delta * Crit_{PA}(m_l, m_r) + (1-\delta) * Crit_{DE}(m_l, m_r) \tag{3}$$

where δ is equal to 0.5.

Leacock–Chodorow similarity criteria [23]**:** It is based on the shortest path that connects the senses and the maximum depth of the taxonomy in which the senses occur.

$$Crit_{Lch}(m_l, m_r) = -\log\frac{d+1}{2 * H(m_l, m_r)} \tag{4}$$

where the maximum depth of noun taxonomy is 13 and verbs is 19.

Depth similarity criteria [24] It deals with the depth only for word senses and their least common subsumer.

$$Crit_D(m_l, m_r) = \frac{2 * H(L(m_l, m_r))}{H(m_l) + H(m_r)} \tag{5}$$

Resnik, Lin, and Jiang–Conrath similarity criteria [25–27] These criteria are based on the information content (INF), which indicates the quantity of information conveyed by a specific word in a particular corpus. So, its value varies from one corpus to another according to the size of the corpus and the number of occurrences and meanings. The proposed system used Brown Corpus to calculate information content. The Brown Corpus was the first million-word electronic corpus of English, created in 1961 at Brown University. This corpus contains text from 500 sources, and the sources have been categorized by

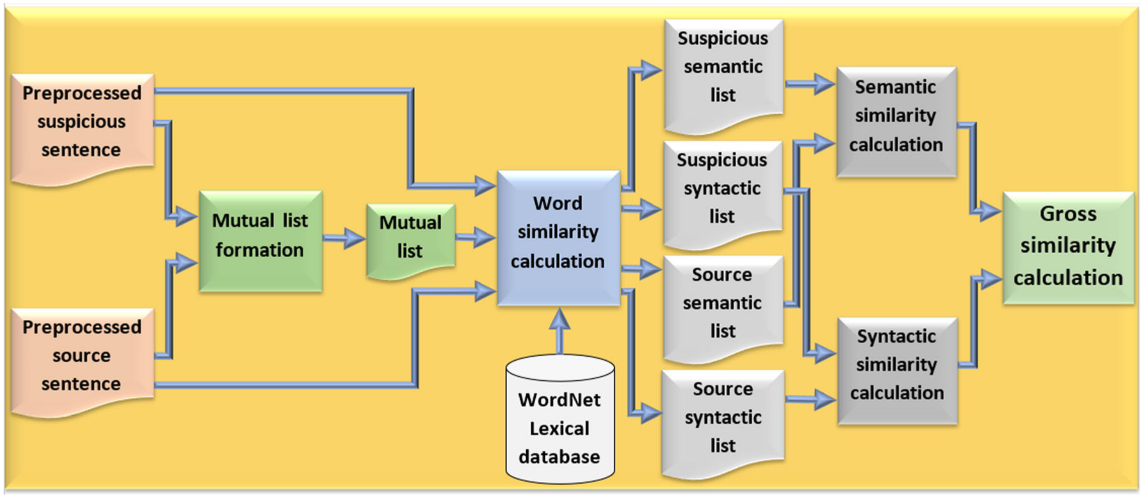
Preprocessed suspicious sentence
Preprocessed source sentence
Mutual list formation
Mutual list
Word similarity calculation
WordNet Lexical database
Suspicious semantic list
Suspicious syntactic list
Source semantic list
Source syntactic list
Semantic similarity calculation
Syntactic similarity calculation
Gross similarity calculation

Fig. 2 Workflow for semantic, syntactic, and gross characteristics calculation

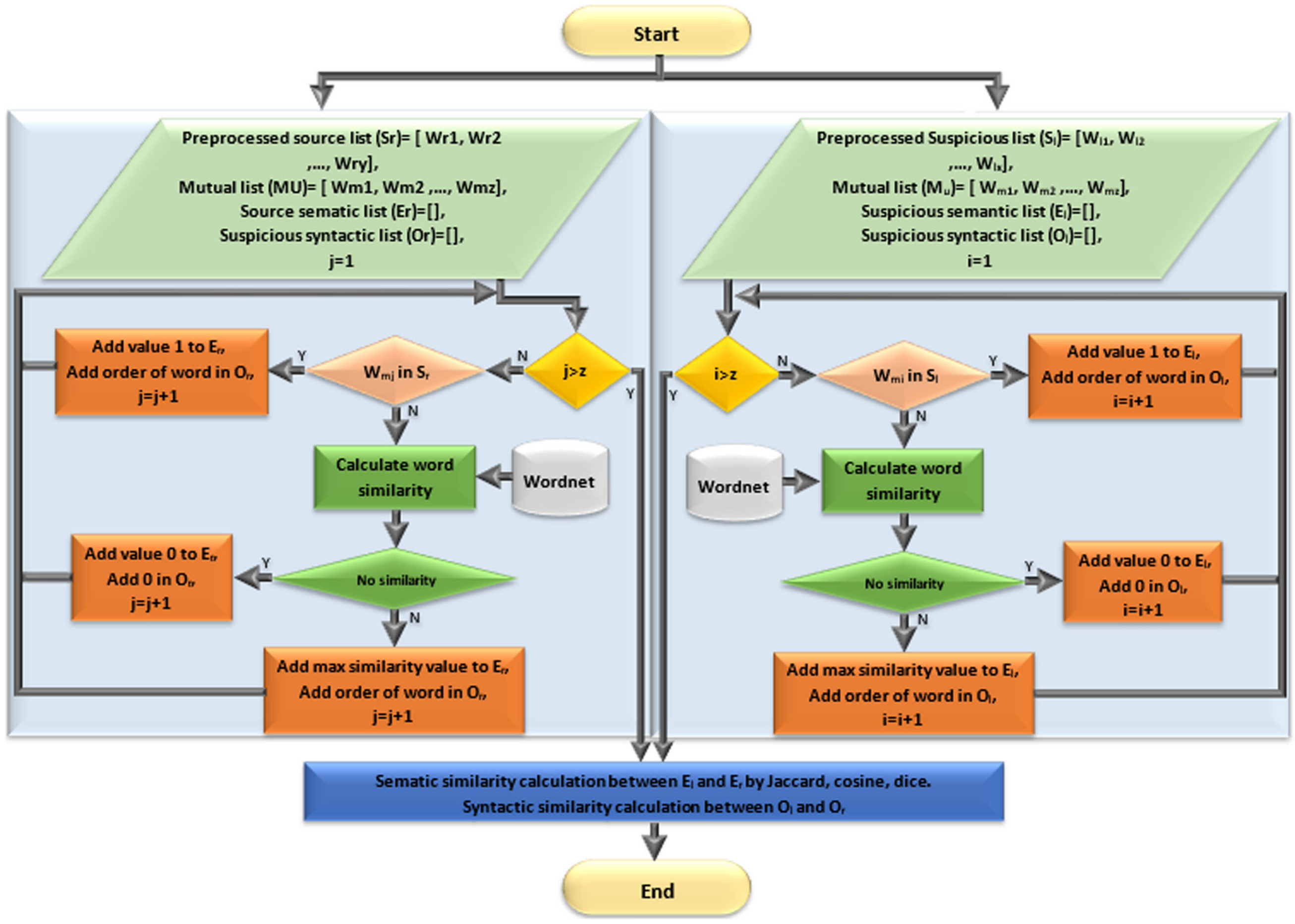
Start
Preprocessed source list (Sr)= [Wr1, Wr2 ,..., Wry], Mutual list (MU)= [Wm1, Wm2 ,..., Wmz], Source sematic list (Er)=[], Suspicious syntactic list (Or)=[], j=1
Preprocessed Suspicious list (Sl)= [Wl1, Wl2 ,..., Wlx], Mutual list (Mu)= [Wm1, Wm2 ,..., Wmz], Suspicious semantic list (El)=[], Suspicious syntactic list (Ol)=[], i=1
j>z
Wmj in Sr
Add value 1 to Er, Add order of word in Or, j=j+1
Calculate word similarity
Wordnet
No similarity
Add value 0 to Er, Add 0 in Or, j=j+1
Add max similarity value to Er, Add order of word in Or, j=j+1
i>z
Wmi in Sl
Add value 1 to El, Add order of word in Ol, i=i+1
Add value 0 to El, Add 0 in Ol, i=i+1
Add max similarity value to El, Add order of word in Ol, i=i+1
Y
N
Sematic similarity calculation between El and Er by Jaccard, cosine, dice.
Syntactic similarity calculation between Ol and Or
End

Fig. 3 Flowchart for calculating semantic and syntactic similarity

genres, such as news and editorial. Additionally, it supports access as a list of words or sentences.

$$Crit_{Res}(m_l, m_r) = INF(L(m_l, m_r)) \tag{6}$$

$$Crit_{Lin}(m_l, m_r) = \frac{2 * INF(L(m_l, m_r))}{INF(m_l) + INF(m_r)} \tag{7}$$

$$Crit_{Jcn}(m_l, m_r) = \frac{1}{INF(m_l) + INF(m_r) - 2 * INF(L(m_l, m_r))} \tag{8}$$

$$INF(m) = \log \frac{1}{P(m)} \tag{9}$$

$$P(m) = \frac{\sum_{w \epsilon W(m)} appearances(w)}{C} \tag{10}$$

where *INF(m)* is information content for word sense *m*, *P(m)* is the probability of word sense *m* in Brown Corpus, $W(m)$ is the set of words in the corpus whose senses are subsumed by *m*, and *C* is the total number of corpus words. The characteristics can be divided into two main types: fuzzy and semantic and syntactic and gross characteristics.

3.2.2 Fuzzy characteristic

This characteristic is a special type of semantic similarity. This characteristic can discover summary cases because it mainly depends on the number and meanings of the words of the suspected sentence when compared with the source sentence without a mediator, as shown in Eq. (11), where each word in the pre-processed suspicious sentence is compared with all the words in pre-processed source sentence by eight criteria [6, 7, 23–27]. According to WordNet, there is a list of synonyms for each word in the comparison process between two words. Therefore, each suspicious synonym was compared with all source synonyms, as shown in Eq. (13) and Eq. (14).

$$Sim\ \ (E_l,\ E_r) = (\rho_{1,r} + \rho_{2,r} + \ldots. + \rho_{l,r} + \ldots. + \rho_{n,r})/k \tag{11}$$

$$\rho_{l,r} = 1 - \ \Pi W_r \in E_r(1 - F_{lr}) \tag{12}$$

$$Flr = \max_{ml \in Wl, mr \in Wr} Flr(Crit_{ws}(m_l, m_r)) \tag{13}$$

$$OFlr = \max_{ml\in Wl,mr\in Wr} Flr(Crit_D(m_l, m_r)) = \begin{cases} 1.0 & if\ o = 1.0 \\ 0.7 & if\ o \in [0.7, 1.0) \\ 0.5 & if\ o \in [0.5, 0.7) \\ 0.3 & if\ o \in [0.3, 0.5) \\ 0.2 & if\ o \in (0.0.0.3) \\ 0.0 & if\ o = 0.0 \end{cases} \tag{14}$$

where E_l *and* E_r rep resent the suspicious sentence and source sentence, respectively, $\rho_{l,r}$ is the correlation coefficient from word to sentence for each word W_l in E_l and E_r, k is the total number of words in E_l, and F_{lr} ($Crit_{ws}$ (m_l, m_r)) is the fuzzy-semantic similarity between W_l and W_r. m_l, m_r are the synonyms/meaning; *ws* is the word similarity criterion. OF_{lr} is the optimized semantic function; it optimizes the semantic output value of the WordNet using heuristic boundary conditions. *o* is the output of optimized fuzzy semantic similarity.

Semantic, syntactic, and gross characteristics

These characteristics depend on their calculation on creating semantic and syntactic lists for both suspicious and source sentences, as shown in Figs. 2 and 3. A mutual list is required for estimating these lists, which was constructed by adding the distinct words for both suspected and source sentences. This led to the dimensions of both the semantic and syntactic lists. The following steps describe in detail how to calculate these lists using eight criteria [6, 7, 23–27] and the desired characteristics:

Step 1: The proposed system examines whether the word from the mutual list is shown in the suspected sentence.

Step 2: If this condition is satisfied, the corresponding dimension in the semantic list is recorded to the value1, and the corresponding dimension in the syntactic list is recorded by the value of the word order of the suspected sentence.

Step 3: Otherwise, the similarity between the word against all words in the suspicious sentence is computed, and the maximum degree of similarity between the pairs of words is recorded in the semantic list, and their order is stored as a value in the syntactic list.

Step 4: If there is no degree of similarity, the value is assigned to 0 in the semantic list corresponding to that word dimension.

Step 5: The above steps are repeated for all words in the mutual list.

Step 6: In like manner, the semantic and syntactic lists for the source sentence are created.

Step 7: Afterward, the semantic similarity is calculated using Jaccard similarity, cosine similarity, and dice similarity as follows:

$$\text{Jaccard}(E_l, E_r) = \frac{\sum_{i=1}^{v} T_{li} * T_{ri}}{\sum_{i=1}^{v} T_{li}^2 + \sum_{i=1}^{v} T_{ri}^2 - \sum_{i=1}^{v} T_{li} * T_{ri}} \tag{15}$$

$$\text{cosine}(E_l, E_r) = \frac{\sum_{i=1}^{v} T_{li} * T_{ri}}{\sqrt{\sum_{i=1}^{v} T_{li}^2} * \sqrt{\sum_{i=1}^{v} T_{ri}^2}} \tag{16}$$

$$dice(E_l, E_r) = \frac{2 * \sum_{i=1}^{v} T_{zi} * T_{ri}}{\sum_{i=1}^{v} T_{li}^2 + \sum_{i=1}^{v} T_{ri}^2} \tag{17}$$

where E_l and E_r are semantic lists derived from suspicious and source sentences, respectively; T_{xi} is the value of the i^{th} dimension in the semantic list; and v is the number of words.

Step 8: The syntactic similarity is calculated as follows:

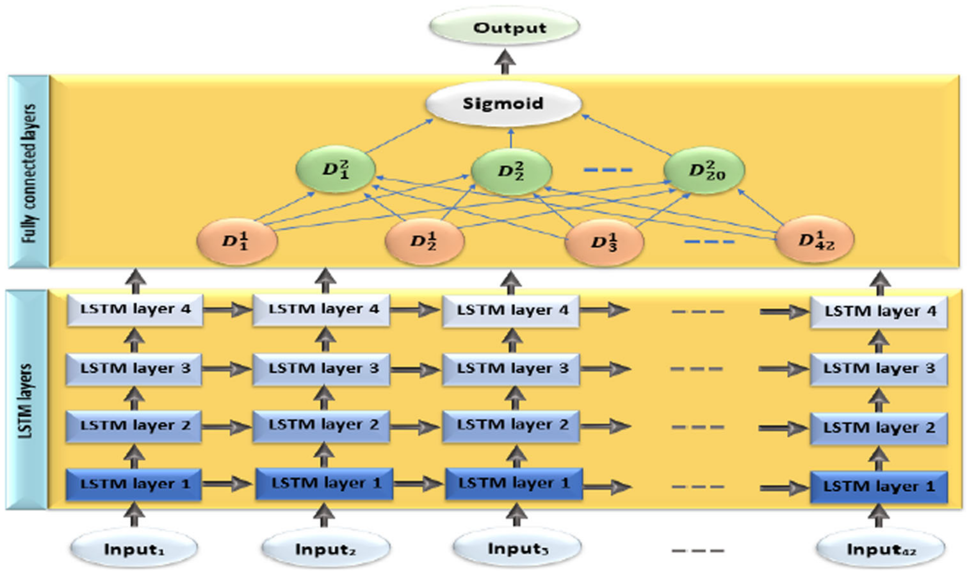

Fig. 4 Structure of the LSTM model

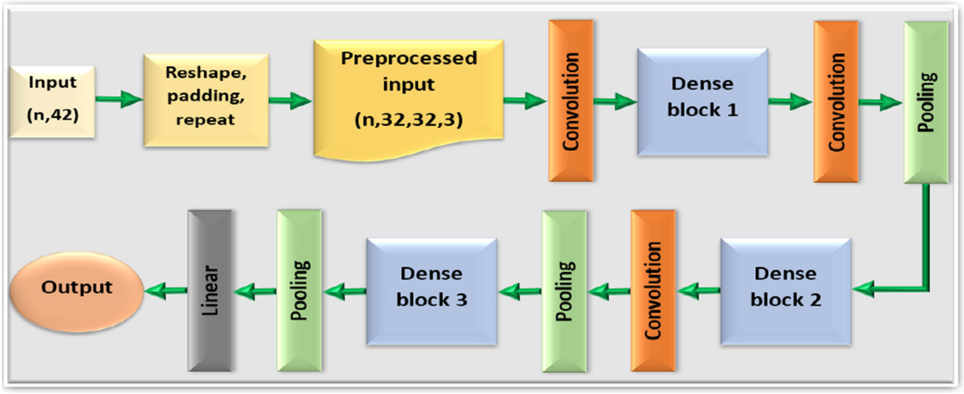

Fig. 5 Structure of DenseNet model

$$\text{Syntactic}(O_l, O_r) = 1 - \frac{||O_l - O_r||}{||O_l + O_r||} \tag{18}$$

$$||O_l - O_r|| = \sqrt{(O_{11} - O_{21})^2 + (O_{12} - O_{22})^2 + \ldots + (O_{1v} - O_{2v})^2} \tag{19}$$

$$||O_l + O_r|| = \sqrt{(O_{11} + O_{21})^2 + (O_{12} + O_{22})^2 + \ldots + (O_{1v} + O_{2v})^2} \tag{20}$$

where O_l and O_r are syntactic lists derived from suspicious and source sentences, respectively; O_{xv} is the value of the i^{th} dimension in the syntactic list; and v is the number of words.

Step 9: The gross similarity between dice semantic and syntactic similarity is calculated based on hybrid word similarity as follows:

$$\text{Gross}(E_l, E_r) = \gamma * (dice(E_l, E_r) + (1 - \gamma) * \text{syntactic}(O_l, O_r) \tag{21}$$

Here, $\gamma = 0.8$ represents the significance coefficient for semantic and syntactic similarity, where dice semantic similarity has the most significant effect on gross similarity.

3.3 Intelligent classification construction

This phase is used to detect the complex plagiarized cases with a high level of obfuscation, if the second phase "Sentence" didn't able to discover the text similarity. The proposed system depends on deep learning architectures: Densely connected convolutional network (DenseNet) [28] and LSTM [29] to determine the most convenient model that can fit the training data. The input data for each technique were normalized using the Z-score method to change the value of data in the dataset to a common scale without distorting the differences in the ranges of the values [30].

3.3.1 LSTM

LSTM is defined as an artificial recurrent neural network (RNN) architecture used in the field of deep learning. It is suitable to classify and make predictions. Here it was used as a classifier that requires the input in the three-dimensional form: seq_len, time_steps, and batch_size. Therefore, the input was reshaped to the desired form and fed into the classifier, where seq_len equals the number of characteristics (seq_len = 42), time_steps equal 1, and batch_size equals 1 upon stochastic mode. According to the sigmoid activation function, the output was in range (0,1), so the cases with values greater than or equal to 0.5 were considered a positive case. Figure 4 shows the architecture of the LSTM model. Figure 4 shows the architecture of the LSTM model, it consists of four LSTM, two fully connected, and output layers. Each cell of the first, second, third, and fourth LSTM layers was constructed of 256, 128, 64, and 32 units, respectively. The fully connected layers were contained by 42 and 20 units in the first and second layers, respectively.

3.3.2 DenseNet

DenseNet is a type of convolutional neural network (CNN) that uses dense connections between layers through dense blocks, where all layers are directly connected with each other, as shown in Fig. 5. It mainly deals with at least $n \times 32 \times 32 \times 3$ input shape (number of samples $\times$ width $\times$ height $\times$ channels). Therefore, each sample with 42 characteristics must satisfy the desired input. So, each case will be reshaped to 6×7 (2D), padding with 26 columns and 25 rows with zero value to be 32×32 forms (2D). Then, it will be repeated thrice to give $32 \times 32 \times 3$ (3D). Figure 6 describes this method. The prediction range 0–1 was used to classify the cases. If the output is greater than or equal to 0.5, it is considered a positive case.

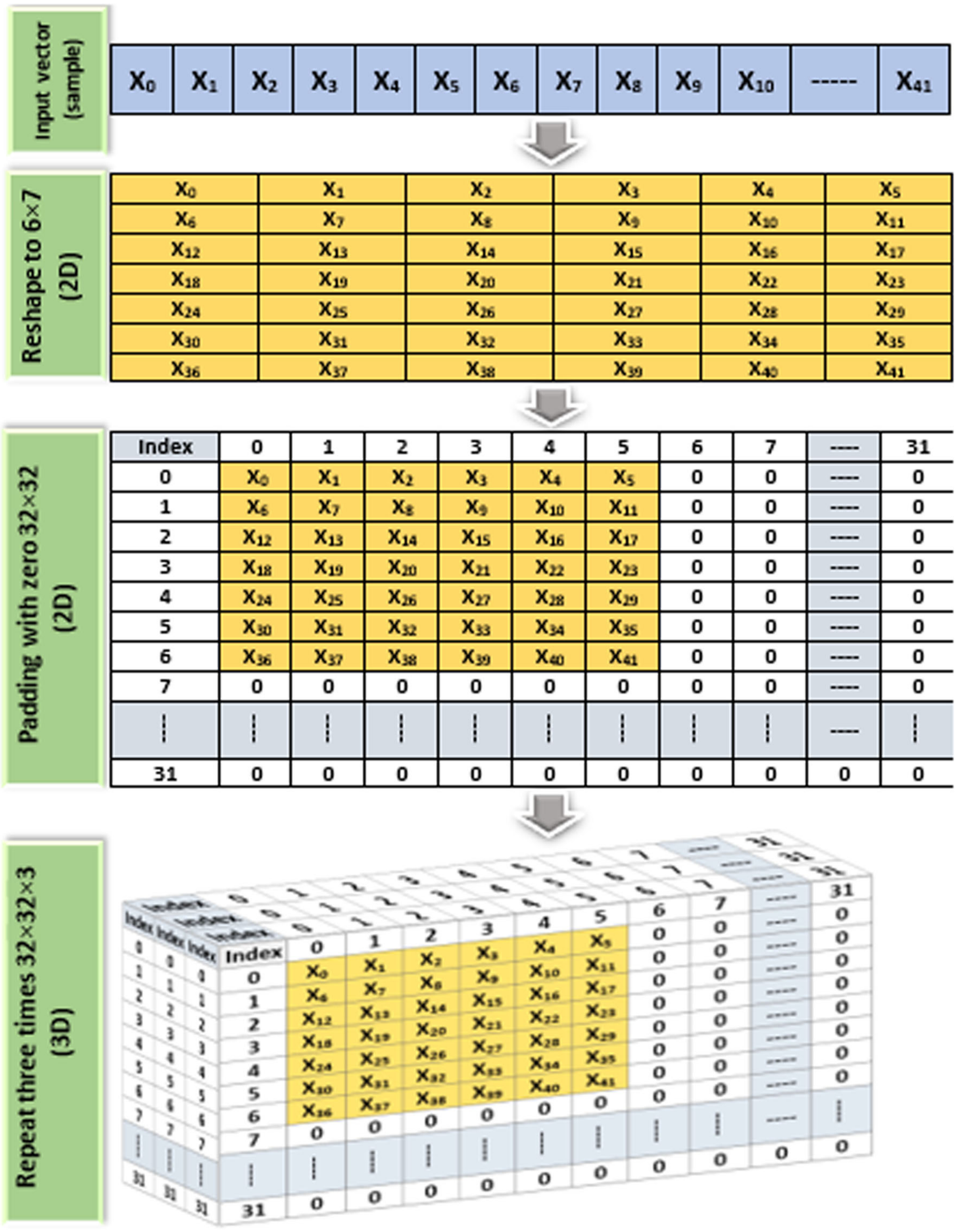

Fig. 6: 1D vector conversion to 3D image

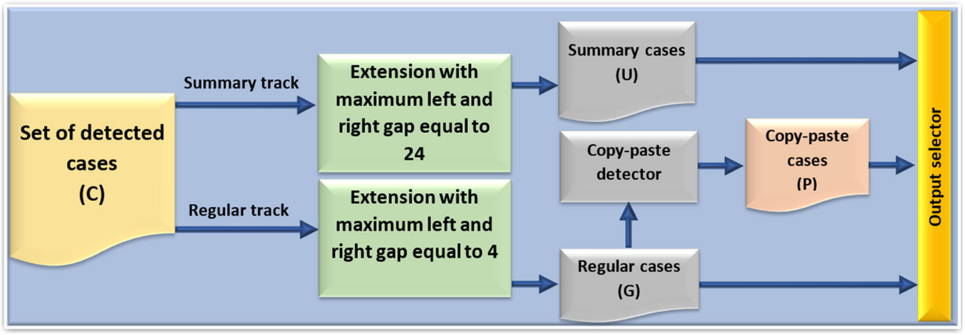
Set of detected cases (C)
Summary track
Regular track
Extension with maximum left and right gap equal to 24
Extension with maximum left and right gap equal to 4
Summary cases (U)
Copy-paste detector
Copy-paste cases (P)
Regular cases (G)
Output selector

Fig. 7 Workflow of adaptive behavior

Table 2 Setting parameters

Parameter	Value	Parameter	Value
th_{upper}	0.4	$max_{right\text{-}gap\text{-}regular}$	4
th_{low}	0.125	$max_{left\text{-}gap\text{-}summary}$	24
δ	0.5	$max_{right\text{-}gap\text{-}summary}$	24
γ	0.8	$min_{left\text{-}gap}$	0
seq_len	42	$min_{right\text{-}gap}$	0
time_steps	1	Len_{left}	100
batch_size	1	Len_{right}	200
Sim_1	0.34	th_{copy}	256
$max_{left\text{-}gap\text{-}regular}$	4		

3.4 Post-processing

In this step, the main objectives were to solve the overlapping problem, merge adjacent detected cases, and discard the short cases, which extract the best-plagiarized segment between the suspicious and source documents. These objectives were achieved through three operations: Filter *I*, adaptive behavior, and Filter *II*.

3.4.1 Filter *I*

The second phase of the proposed system may discover similar cases with multiple source sentences. Therefore, the proposed system supposes that only one suspicious sentence is plagiarized with one source sentence, but not vice versa to solve the overlapping problem. This process allows the pair of suspicious and corresponding source sentences with the maximum meteor score to pass the next steps.

3.4.2 Adaptive behavior

This behavior supports the proposed system in terms of dealing with various types of obfuscation, as in PAN Workshop series [18, 19]. Several cases exist, including copy–paste, random obfuscation, translation obfuscation, and summary obfuscation. Each type needs a special method with specific parameters to deal with it and adapt to get the best performance. However, this type of obfuscation remains unknown. Therefore, the proposed system should be adapted with different types using the following methods illustrated in Fig. 7.

Adaptive extension The input of cases C passed from Filter *I* has been defined as the pairs (x, y) of the detected plagiarized sentences. These sentences need a technique to merge with their adjacent sentences to form the largest text segments similar to the two documents. This technique is called extension, and it is divided into two sub-processes [4], that is, clustering and validation.

Clustering process The cases that are not separated by more gaps of sentences are grouped. This process was implemented by sorting and clustering the set of cases by x (left or suspicious document) such that $x_n - x_{n+1} \leq max_{left\text{-}gap}$. Then, the sorting and clustering processes were used for each resulting cluster based on y through a $max_{right\text{-}gap}$ threshold (right or source document).

Validation process However, the proposed system uses the parameters $max_{left\text{-}gap}$ and $max_{right\text{-}gap}$ to cluster the cases into the largest text segments. Some sentences in these segments may have no similarity to any sentences in the corresponding segment. Therefore, to avoid adding noise in the clustering step, we have validated the similarity between the text segments of the remaining clusters using a threshold. If the similarity is less than the given threshold (Sim_1), it then applies the extension stage using $max_{left\text{-}gap} - 1$ and $max_{right\text{-}gap} - 1$ for this particular cluster, therefore reducing the gaps to $min_{left\text{-}gap}$ and $min_{right\text{-}gap}$ values, respectively. If any minimum values are reached and the validation condition is not met, the cluster is then discarded. Given a cluster integrated by the cases of the form (x, y), the text segment in the suspicious document segment seg_{left} collects all the sentences from the smallest to the largest x in the cluster. Similarly, the corresponding text segment in the source document seg_{right} collects all the sentences from the smallest to the largest y in the cluster. This proposed system to measure the similarity between the segments is based on gross similarity in

Table 3 The description of TSF training and testing subsets datasets

Types	Training			Testing					
	PAN 2013			PAN 2013			PAN 2014		
	# of Pairs	Positive cases	Negative cases	# of Pairs	Positive cases	Negative cases	# of Pairs	Positive cases	Negative cases
No-plagiarism	1000	–	10,384	1000	–	9464	1600	–	13,033
No-obfuscation	1000	5305	–	1000	5082	–	1600	8637	–
Random obfuscation	1000	4906	–	1000	5160	–	1600	7773	–
Translated	1000	5055	–	1000	5515	–	–	–	–
Summary	1185	980	–	1185	977	–	–	–	–

Eq. (20). The extension technique was used twice with different left and right gap parameters for the adaptive extension.

The adaptive extension uses the extension technique with two tracks upon different $\max_{\text{left-gap}}$ and $\max_{\text{right-gap}}$ values to achieve the best result. The first track is a regular track, where the maximum left and right gaps are equal to 4 sentences which are found to realize the best performance to copy–paste, random, and translation subsets. The second is a summary track, where the maximum left and right gaps equal 24 sentences, which are the best for the summary obfuscation subset.

Copy–paste detector This detector obtains copy–paste segments from the regular case. The longest common substring algorithm is used with more than or equals a certain threshold (th_{copy}) measured in characters.

Output selector After applying adaptive extension and copy–paste detector methods, three outputs are extracted and are named as follows: regular plagiarism cases (Cases G), summary plagiarism cases (Cases U), and copy–paste plagiarism cases (Cases P). The output selector selects copy–paste cases as a priority. Then, if the length of the source segment is more than or equal to three times the length of the corresponding suspicious segment, it is then called summary cases, otherwise regular cases.

Filter II Small pairs of plagiarized segments are removed in this step. The pair of segments is considered small if the suspicious segment's length is less than Lenleft or the length of the source segment is less than Lenright.

4 Experimental setup

This proposed system was developed using Python programming language. It was implemented on an Intel®–CoreTMi5-4210U CPU @ 2.40 GHz computer with 8.00 GB RAM. Table 2 indicates the setting of the parameters used in the proposed system.

4.1 Datasets

The proposed system was evaluated using two benchmark datasets, that is, PAN 2013 and PAN 2014. These datasets were provided for text-alignment sub-tasks in the plagiarism detection depending on various types of obfuscation such as:

- No-obfuscation: This type is known as copy–paste, where the source text and reused text are identical.
- Random obfuscation: The source text is developed using various plagiarism techniques to change the sentences' composition and semantics or both. These techniques are applied using automatic software tools or manually.
- Cyclic translation: Machine translators convert the source text into other languages and then return it into the original language.
- Summary: This is considered the most complex among the other types because it depends on a good understanding of the subject or text to be plagiarized manually.

PAN 2014 is a supplement to PAN 2013, and it consists of only two forms of obfuscation: no-obfuscation and random obfuscation. Number of documents for each obfuscation type in the training and testing corpus of PAN 2013 and PAN 2014 is presented in Table 3.

4.2 TSF database creation

The proposed system depends on building a supervised training database to train the intelligent classification models. Therefore, TSF database was created based on the available benchmark datasets PAN 2013 and PAN 2014. Forty-two values of the sentences similarity features are computed and recorded aggregating with the class label for each extracted case to build TFS database. The positive and negative cases of the TSF database were extracted from the

Table 4 Description of TSF database features

No	Feature Name	Description	No	Feature Name	Description
1	Cosine Semantic—Depth Estimation	Calculation semantic similarity by Cosine measure between two cases based on Depth Estimation criterion	23	Fuzzy Semantic—Depth Estimation	Calculation semantic similarity by Fuzzy measure between two cases based on Depth Estimation criterion
2	Dice Semantic—Depth Estimation	Calculation semantic similarity by Dice measure between two cases based on Depth Estimation criterion	24	Syntactic—JCN	Calculation syntactic variations between two cases based on JCN criterion
3	Jaccard Semantic—Depth Estimation	Calculation semantic similarity by Jaccard measure between two cases based on Depth Estimation criterion	25	Dice Semantic—JCN	Calculation semantic similarity by Dice measure between two cases based on JCN criterion
4	Cosine Semantic—Hybrid	Calculation semantic similarity by Cosine measure between two cases based on Hybrid criterion	26	Jaccard Semantic—JCN	Calculation semantic similarity by Jaccard measure between two cases based on JCN criterion
5	Dice Semantic—Hybrid	Calculation semantic similarity by Dice measure between two cases based on Hybrid criterion	27	Jaccard Semantic—Depth	Calculation semantic similarity by Jaccard measure between two cases based on Depth criterion
6	Jaccard Semantic—Hybrid	Calculation semantic similarity by Jaccard measure between two cases based on Hybrid criterion	28	Dice Semantic—Depth	Calculation semantic similarity by Dice measure between two cases based on Depth criterion
7	Gross similarity	Calculation semantic similarity by combining between Dice measure and Syntactic based on Hybrid criterion	29	Cosine Semantic—Depth	Calculation semantic similarity by Cosine measure between two cases based on Depth criterion
8	Cosine Semantic—Path	Calculation semantic similarity by Cosine measure between two cases based on Path criterion	30	Cosine Semantic—RES	Calculation semantic similarity by Cosine measure between two cases based on RES criterion
9	Jaccard Semantic—Path	Calculation semantic similarity by Jaccard measure between two cases based on Path criterion	31	Optimized Fuzzy Semantic—Depth	Calculation semantic similarity by Optimized fuzzy measure between two cases based on Depth criterion
10	Dice Semantic—Path	Calculation semantic similarity by Dice measure between two cases based on Path criterion	32	Fuzzy Semantic—Depth	Calculation semantic similarity by Fuzzy measure between two cases based on Depth criterion
11	Fuzzy Semantic—Path	Calculation semantic similarity by Fuzzy measure between two cases based on Path criterion	33	Cosine Semantic—LCH	Calculation semantic similarity by Cosine measure between two cases based on LCH criterion
12	Syntactic—Path	Calculation syntactic variations between two cases based on Path criterion	34	Jaccard Semantic—LCH	Calculation semantic similarity by Jaccard measure between two cases based on LCH criterion
13	Syntactic—Depth Estimation	Calculation syntactic variations between two cases based on Depth Estimation criterion	35	Dice Semantic—LCH	Calculation semantic similarity by Dice measure between two cases based on LCH criterion
14	Syntactic—Hybrid	Calculation syntactic variations between two cases based on Hybrid criterion	36	Dice Semantic—RES	Calculation semantic similarity by Dice measure between two cases based on RES criterion
15	Syntactic—LCH	Calculation syntactic variations between two cases based on LCH criterion	37	Jaccard Semantic—RES	Calculation semantic similarity by Jaccard measure between two cases based on RES criterion
16	Syntactic—RES	Calculation syntactic variations between two cases based on RES criterion	38	Fuzzy Semantic—Hybrid	Calculation semantic similarity by Fuzzy measure between two cases based on Hybrid criterion
17	Syntactic—Depth	Calculation syntactic variations between two cases based on Depth criterion	39	Fuzzy Semantic—LCH	Calculation semantic similarity by Fuzzy measure between two cases based on LCH criterion
18			40		

Table 4 (continued)

No	Feature Name	Description	No	Feature Name	Description
	Jaccard Semantic—LIN	Calculation semantic similarity by Jaccard measure between two cases based on LIN criterion		Fuzzy Semantic—JCN	Calculation semantic similarity by Fuzzy measure between two cases based on JCN criterion
19	Cosine Semantic—JCN	Calculation semantic similarity by Cosine measure between two cases based on JCN criterion	41	Fuzzy Semantic—RES	Calculation semantic similarity by Fuzzy measure between two cases based on RES criterion
20	Cosine Semantic—LIN	Calculation semantic similarity by Cosine measure between two cases based on LIN criterion	42	Fuzzy Semantic—LIN	Calculation semantic similarity by Fuzzy measure between two cases based on LIN criterion
21	Dice Semantic—LIN	Calculation semantic similarity by Dice measure between two cases based on LIN criterion	43	Output	1: represents a positive case 0: represents a negative case
22	Syntactic—LIN	Calculation syntactic variations between two cases based on LIN criterion			

benchmark datasets as explained in Sect. 3.2. The training cases of the TSF database were extracted from a training corpus of PAN 2013, and the test cases were selected from a testing corpus of PAN 2013 and PAN 2014. Table 3 shows the number of samples of TSF training and testing databases.

The training database was used to train the intelligent classification models, and the testing database was used to evaluate the constructed intelligent classifiers. The order of TFS features is effective in the construction process of DenseNet and LSTM classifiers. Therefore, information gain [31] was applied to rank the features. Table 4 shows the description of the ranked features of TSF database.

The statistical analysis of TSF database was developed and is shown in Table 5 to explain the importance of each created feature. The range of values for all sentence similarity features of TSF 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, 95% confidence limits, and P-value scores were calculated for each sentence similarity feature.

4.3 Evaluation metrics

Potthast et al. [32] introduced a novel score called Plagdet to assess the effectiveness of various plagiarism detection techniques. This score combines three measurements factors: recall, precision, and granularity. These measurements are computed as follows:

$$Rec(A,B) = \frac{1}{A} \times \sum_{a \in A} \frac{\left|\bigcup_{b \in B}(a \cap b)\right|}{|a|} \tag{22}$$

$$Prec(A,B) = \frac{1}{B} \times \sum_{b \in B} \frac{\left|\bigcup_{a \in A}(a \cap b)\right|}{|b|} \tag{23}$$

where $a \cap b$

$$= \begin{cases} a \cap b & \text{if} b \text{ detects} a (\text{number of overlapping characters}) \\ \emptyset & \text{otherwise} \end{cases}$$

where $A, a, B,$ *and* b are the actual collection of all plagiarism cases, plagiarism case, detected collection of all plagiarism instances, and detected case using the proposed system, respectively. Granularity is determined in Eq. (23) to handle overlapping or multiple detections for one plagiarism instance. This measure assesses the ability of the detection system to detect each instance of plagiarism as a single piece.

$$\text{Gran}(A,B) = \frac{1}{|A_B|} \times \sum_{a \in A_B} |B_A| \tag{24}$$

where $A_B \subseteq A$ is the instances detected in A and $B_A \subseteq A$ is all the detections of instance a.

The Plagdet score is calculated as:

$$\text{Plagdet}(A,B) = \frac{F-\text{measure}}{\log_2(1+\text{Gran}(A,B))} \tag{25}$$

where the F-measure is the harmonic mean of recall and precision and is estimated as follows:

$$F-\text{measure} = \frac{2 \times \text{Rec}(A,B) \times \text{Prec}(A,B)}{\text{Rec}(A,B) + \text{Prec}(A,B)} \tag{26}$$

4.4 Results and discussions

The proposed system depends on two paths to detect the text plagiarism. The first path is based on traditional

Table 5 Statistical analysis of TSF database features

Sentence Similarity Feature	The Extracted Cases		95% Confidence Limits		P-value
	Negative N = 10,384 Mean ± Standard Deviation	Positive N = 16,246 Mean ± Standard Deviation	Lower limit	Upper limit	
Cosine Semantic—Depth Estimation	0.346 ± 0.099	0.835 ± 0.183	0.641	0.648	0.841
Dice Semantic—Depth Estimation	0.331 ± 0.100	0.827 ± 0.196	0.630	0.637	0.911
Jaccard Semantic—Depth Estimation	0.202 ± 0.075	0.747 ± 0.256	0.530	0.539	0.815
Cosine Semantic—Hybrid	0.450 ± 0.089	0.858 ± 0.158	0.696	0.702	0.878
Dice Semantic—Hybrid	0.430 ± 0.091	0.849 ± 0.173	0.683	0.689	0.777
Jaccard Semantic—Hybrid	0.279 ± 0.076	0.772 ± 0.235	0.576	0.583	0.828
Gross similarity	0.442 ± 0.084	0.840 ± 0.173	0.682	0.688	0.942
Cosine Semantic—Path	0.543 ± 0.082	0.879 ± 0.137	0.746	0.750	0.969
Jaccard Semantic—Path	0.357 ± 0.079	0.797 ± 0.214	0.622	0.629	0.888
Dice Semantic—Path	0.521 ± 0.085	0.869 ± 0.153	0.731	0.736	0.828
Fuzzy Semantic—Path	0.664 ± 0.135	0.927 ± 0.115	0.822	0.827	0.668
Syntactic—Path	0.486 ± 0.124	0.807 ± 0.199	0.679	0.685	0.886
Syntactic—Depth Estimation	0.455 ± 0.128	0.796 ± 0.211	0.660	0.666	0.960
Syntactic—Hybrid	0.487 ± 0.124	0.807 ± 0.198	0.679	0.685	0.888
Syntactic—LCH	0.487 ± 0.125	0.806 ± 0.198	0.678	0.684	0.844
Syntactic—RES	0.486 ± 0.123	0.803 ± 0.200	0.677	0.682	0.734
Syntactic—Depth	0.498 ± 0.123	0.809 ± 0.196	0.685	0.691	0.969
Jaccard Semantic—LIN	0.586 ± 0.113	0.854 ± 0.174	0.747	0.752	0.708
Cosine Semantic—JCN	0.287 ± 0.223	0.623 ± 0.419	0.487	0.497	0.949
Cosine Semantic—LIN	0.750 ± 0.087	0.917 ± 0.108	0.851	0.854	0.907
Dice Semantic—LIN	0.733 ± 0.094	0.910 ± 0.120	0.839	0.842	0.773
Syntactic—LIN	0.506 ± 0.123	0.807 ± 0.196	0.687	0.692	0.774
Fuzzy Semantic—Depth Estimation	0.401 ± 0.147	0.793 ± 0.235	0.637	0.643	0.951
Syntactic—JCN	0.496 ± 0.127	0.804 ± 0.199	0.681	0.687	0.905
Dice Semantic—JCN	0.273 ± 0.214	0.616 ± 0.420	0.478	0.487	0.840
Jaccard Semantic—JCN	0.176 ± 0.144	0.571 ± 0.419	0.412	0.422	0.948
Jaccard Semantic—Depth	0.730 ± 0.095	0.903 ± 0.131	0.834	0.837	0.705
Dice Semantic—Depth	0.840 ± 0.067	0.943 ± 0.086	0.902	0.904	0.790
Cosine Semantic—Depth	0.852 ± 0.061	0.948 ± 0.076	0.910	0.912	0.642
Cosine Semantic—RES	0.384 ± 0.092	0.659 ± 0.312	0.548	0.555	0.792
Optimized Fuzzy Semantic—Depth	0.838 ± 0.104	0.960 ± 0.084	0.911	0.914	0.561
Fuzzy Semantic—Depth	0.849 ± 0.100	0.963 ± 0.081	0.917	0.920	0.458
Cosine Semantic—LCH	0.741 ± 0.046	0.855 ± 0.140	0.809	0.812	0.555
Jaccard Semantic—LCH	0.574 ± 0.050	0.756 ± 0.214	0.683	0.687	0.877
Dice Semantic—LCH	0.728 ± 0.040	0.844 ± 0.144	0.797	0.800	0.636
Dice Semantic—RES	0.366 ± 0.078	0.626 ± 0.319	0.521	0.528	0.813
Jaccard Semantic—RES	0.227 ± 0.057	0.542 ± 0.377	0.415	0.423	0.941
Fuzzy Semantic—Hybrid	0.652 ± 0.141	0.839 ± 0.198	0.764	0.769	0.913
Fuzzy Semantic—LCH	0.501 ± 0.204	0.785 ± 0.239	0.671	0.678	0.787
Fuzzy Semantic—JCN	0.667 ± 0.141	0.821 ± 0.207	0.762	0.767	0.787
Fuzzy Semantic—RES	0.638 ± 0.177	0.821 ± 0.212	0.747	0.752	0.749
Fuzzy Semantic—LIN	0.810 ± 0.128	0.894 ± 0.154	0.860	0.863	0.705

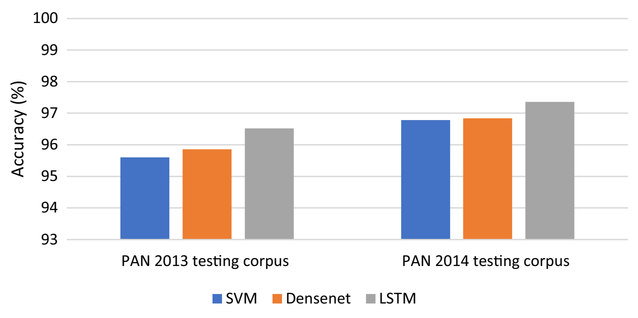

Fig. 8 Comparison between SVM, DenseNet, and LSTM models on the TSF testing dataset

paragraph-level comparison, and the second path is based deep learning approach. The second path is used, if the first path didn't able to discover the text similarity; it is based on constructing deep learning classifier that has the ability to detect all the confused types of lexical, syntactic, and semantic plagiarism cases. As shown in Table 5, all the sentence similarity features have closer positive and negative mean values, 95% confidence limits values of the positive cases are far from the one value and closer to the positive and negative mean values, and P-value is more than 0.05, this indicates that relying with each feature will cause a confusion in the decision. Therefore, the previous researches [3–5, 7, 11, 12, 20] 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 not discriminative as shown in Table 5. Therefore, the proposed system takes into consideration all different possible types of the sentence similarity features by creating TSF database to train intelligent classification algorithms.

A comparative study was implemented to determine the most convenient model to fit the TSF training dataset. It was also used to detect text plagiarism in the TSF testing dataset with the highest accuracy. This study was conducted on different intelligent learning models: SVM, DenseNet, and LSTM. According to the experiments conducted with these three models, the results showed that the

Table 6 Performance of SVM classifier on the TSF testing dataset

Dataset	Types	Recall (%)	Precision (%)	F-score (%)	Gran	PlagDet (%)
PAN 2013 test	No-obfuscation	96.73	99.49	98.09	1.0	98.09
	Random obfuscation	88.54	85.03	86.75	1.00251	86.60
	Translated	90.16	79.79	84.66	1.00161	84.56
	Summary	36.54	90.37	52.04	1.20192	45.70
	All obfuscation types	88.47	87.78	88.12	1.01227	87.35
	No-plagiarism	96.62	94.15	95.37	–	–
PAN 2014 test	No-obfuscation	96.29	99.52	97.88	1.0	97.88
	Random obfuscation	87.18	89.79	88.46	1.0	88.46
	All obfuscation types	91.83	94.85	93.31	1.0	93.31
	No-plagiarism	98.31	97.35	97.83	–	–

Table 7 Performance of DenseNet classifier on the TSF testing dataset

Dataset	Types	Recall (%)	Precision (%)	F-score (%)	Gran	PlagDet (%)
PAN 2013 test	No-obfuscation	96.77	99.46	98.10	1.0	98.10
	Random obfuscation	88.80	86.29	87.53	1.00251	87.37
	Translated	90.14	81.81	85.77	1.00160	85.67
	Summary	37.80	94.40	53.98	1.18009	48.00
	All types	88.63	89.15	88.89	1.01121	88.18
	No-plagiarism	96.71	94.19	95.43	–	–
PAN 2014 test	No-obfuscation	96.31	99.50	97.88	1.0	97.88
	Random obfuscation	87.71	90.74	89.20	1.0	89.20
	All types	92.09	95.31	93.68	1.0	93.68
	No-plagiarism	98.61	97.44	98.02	–	–

Table 8 Performance of LSTM classifier on the TSF testing dataset

Dataset	Types	Recall (%)	Precision (%)	F-score (%)	Gran	PlagDet (%)
PAN 2013 test	No-obfuscation	96.90	99.49	98.18	1.0	98.18
	Random obfuscation	88.74	89.04	88.89	1.00251	88.73
	Translated	90.34	86.36	88.31	1.00160	88.20
	Summary	42.39	95.60	58.74	1.11905	54.22
	All types	88.99	91.69	90.32	1.00781	89.81
	No-plagiarism	98.12	94.39	96.21	–	–
PAN 2014 test	No-obfuscation	96.23	99.49	97.83	1.0	97.83
	Random obfuscation	87.68	91.92	89.75	1.0	89.75
	All types	92.04	95.88	93.92	1.0	93.92
	No-plagiarism	99.11	97.57	98.33	–	–

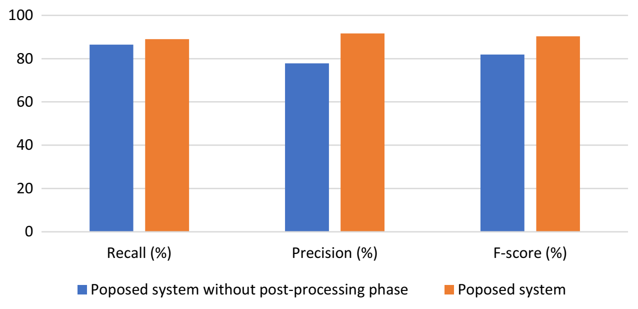

Fig. 9 Effectiveness of post-processing step in the proposed system's performance on detecting the all types of PAN 2013 obfuscations

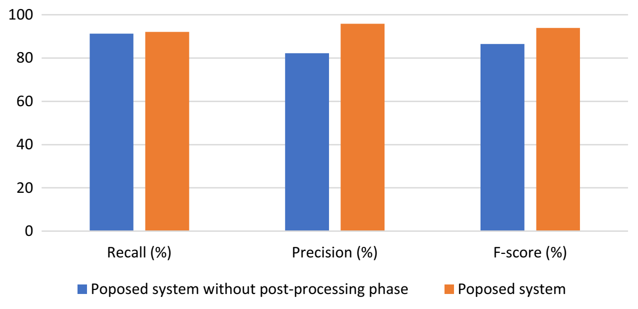

Fig. 10 Effectiveness of post-processing step in the proposed system's performance on detecting the all types of PAN 2014 obfuscations

LSTM model was able to achieve the highest accuracy using different test datasets. These results are presented in Fig. 8 and Tables 6, 7, and 8. The LSTM can weigh the fluctuations between the values of 42 features. This ability has helped discover the highest accuracy of the lexical, syntactic, and semantic similarity cases compared with SVM and DenseNet classifiers.

Based on the previous results, the proposed system is based on the constructed LSTM classifier to detect text plagiarism. This system depends on n-grams and meteor score techniques to detect the simple cases and the constructed LSTM for the complex cases. As explained in Sect. 3.2, the meteor score was computed for each pair of sentences. If the meteor score value is more than or equal to th_{upper}, it is considered a plagiarized case. Else if the meteor score value is less than th_{low}, the case is then discarded. Additionally, if neither the first nor second conditions are fulfilled, the case is analyzed using the constructed LSTM. After the classification process.

The post-processing phase of the proposed system was applied as has been explained in Sect. 3.3 to find the best largest plagiarized segment by solving the overlapping issues, merging adjacent cases, and removing small cases. To study the contribution of post-processing phase in the proposed system's performance, the proposed system without post-processing phase was applied to discover all the obfuscation types in the testing corpus of PAN 2013 and PAN 2014. The results demonstrated the effectiveness of post-processing phase to improve the proposed system's performance as shown in Figs. 9 and 10.

The proposed system was evaluated on the PAN 2013 and PAN 2014 datasets and compared with recent research systems. From Tables 9, 10, and 11, it is evident that the suggested system is efficient and accurate for all forms of plagiarism. It surpasses the competition on the PAN 2013 random obfuscation subset, entire PAN 2013 dataset, and entire PAN 2014 dataset, which achieved the highest accuracy compared to up-to-date ranking research in this field.

As per the previous results, the proposed system could balance the evaluation criteria, precision, and recall plus reduce the granularity, where all steps of the proposed system contributed to making it reliable and robust:

- Preprocessing: Contributed to reducing the incidence of false-positive cases and the size of comparable texts.

Table 9 The proposed system's performance compared to the prior systems on PAN 2013 random obfuscation subset

Team	Recall (%)	Precision (%)	F-measure (%)	Gran	PlagDet (%)
Proposed system with LSTM	**88.74**	89.04	**88.89**	1.00251	**88.73**
Sanchez-Perez et al. [4]	86.07	91.02	88.48	1.00086	88.42
PlagLinSVM [20]	84.72	**92.13**	88.27	1.00000	88.27
PlagRbfSVM [20]	85.17	89.58	87.32	1.00000	87.32
Oberreuter and Eiselt [33]	83.25	90.61	86.77	1.00000	86.78
Shrestha et al. [34]	83.16	91.10	86.95	1.00630	86.56
Palkovskii and Belov [35]	82.24	91.45	86.60	1.00176	86.50
Vani and Gupta [36]	79.92	87.88	83.71	1.0010	83.65
Kong et al. 1 [37]	77.90	89.37	83.24	1.00000	83.24
Kong et al. 2 [38]	78.08	87.00	82.30	1.00000	82.30
Kong et al. 3 [39]	78.68	86.22	82.28	1.00000	82.28
Glinos [40]	72.48	96.95	82.95	1.04037	80.62
Gross and Modaresi [41]	71.88	96.00	82.21	1.03336	80.29
Palkovskii and Belov [42]	75.13	84.84	79.69	1.00000	79.69
Rodríguez Torrejón and Martín Ramos [43]	62.99	93.84	75.38	1.00000	75.38
Suchomel et al. [44]	68.89	82.97	75.28	1.00000	75.28
Oberreuter et al. [45]	65.32	87.92	74.95	1.00000	74.96
Rodríguez Torrejón and Martín Ramos [46]	63.37	91.00	74.71	1.00000	74.71
Gharavi et al. [11]	72.90	75.10	73.99	1.00000	73.99
Daud et al. [47]	64.12	81.76	71.86	1.00000	71.86
Rodríguez Torrejón and Martín Ramos [48]	60.28	83.88	70.15	1.00000	70.15
Shrestha and Solorio [49]	71.46	92.34	80.57	1.30962	66.71
Saremi and Yaghmaee [50]	68.88	91.81	78.71	1.29511	65.67
Suchomel et al. [51]	51.95	87.58	65.22	1.00000	65.21
Küppers and Conrad [52]	36.87	89.89	52.29	1.01847	51.60
Alvi et al. [53]	36.60	94.79	52.81	1.07203	50.25
Palkovskii and Belov [54]	36.42	93.14	52.36	1.06785	49.96
Abnar et al. [55]	35.36	82.99	49.59	1.01509	49.06
Sánchez-Vega et al. [56]	43.50	49.52	46.32	1.02200	45.60
Nourian [57]	23.61	96.27	37.92	1.11558	35.08
Jayapal and Goswami [58]	18.18	92.31	30.38	2.19096	18.15

Bold values indicate to show the highest values

- Detailed Analysis: Participated in extracting the most accurate cases of plagiarism between the suspicious and original documents.
- TSF database creation: Included linguistic analysis of the texts and extracting linguistic features that contain all the aspects that could be applied to change the structure of the plagiarized text and can distinguish the classified cases.
- LSTM classification construction relies on deep learning techniques, which weigh the fluctuations between the linguistic features instead of the experimental setting of the similarity criterion parameters in the previous research.
- Post-processing: Improved the precision and reduced the granularity without significant impact on the recall.

5 Conclusion

This paper has constructed a new database for intelligent learning purposes to detect text plagiarism. It had 42 features for each similarity case. The database values were computed using the words and sentences similarity metrics that reflect all different lexical, syntactic, and semantic text plagiarism types. Each case of the created database was

Table 10 The proposed system's performance compared to the prior systems on the complete PAN 2013 dataset

Team	Recall (%)	Precision (%)	F-measure (%)	Gran	PlagDet (%)
Proposed System with LSTM	**88.99**	91.69	**90.32**	1.00781	**89.81**
Sanchez-Perez et al. [4]	87.90	88.17	88.03	1.00344	87.82
Oberreuter and Eiselt [33]	85.78	88.60	87.17	1.00369	86.93
Palkovskii and Belov [35]	82.64	92.23	87.17	1.00580	86.81
PlagLinSVM [20]	85.01	88.72	86.83	1.00609	86.45
Glinos [40]	79.33	**96.25**	86.97	1.01695	85.93
PlagRbfSVM [20]	86.00	84.87	85.43	1.00791	84.95
Rodríguez Torrejón and Martín Ramos [48]	76.90	90.43	83.12	1.00278	82.95
Shrestha et al. [34]	83.78	85.91	84.83	1.00701	84.40
Gross and Modaresi [41]	76.62	93.27	84.13	1.02514	82.64
Rodríguez Torrejón and Martín Ramos [46]	76.19	89.48	82.30	1.00141	82.22
Kong et al. 1 [37]	80.75	84.01	82.35	1.00309	82.16
Kong et al. 2 [38]	81.34	82.86	82.09	1.00336	81.90
Gharavi et al. [11]	76.71	83.36	79.90	1.00000	79.90
Suchomel et al. [44]	76.59	72.51	74.49	1.00028	74.48
Saremi and Yaghmaee [50]	77.12	86.51	81.55	1.24450	69.91
Shrestha and Solorio [49]	73.81	87.46	80.06	1.22084	69.55
Abnar et al. [55]	61.16	77.33	68.30	1.02245	67.22
Alvi et al. [53]	55.07	93.38	69.28	1.07111	65.95
Palkovskii and Belov [54]	53.56	81.70	64.70	1.07295	61.52
Nourian [57]	43.38	94.71	59.50	1.04343	57.72
Gillam [59]	25.89	88.49	40.06	1.00000	40.06
Gillam and Notley [60]	16.84	88.63	28.30	1.00000	28.30
Jayapal and Goswami [58]	38.19	87.90	53.25	2.90698	27.08

Bold values indicate to show the highest values

Table 11 The proposed system's performance compared to the prior systems on the complete PAN 2014 dataset

Team	Recall (%)	Precision (%)	F-measure (%)	Gran	PlagDet (%)
Proposed System with LSTM	**92.04**	95.88	**93.92**	1.00000	**93.92**
Palkovskii and Belov [35]	88.92	92.76	90.80	1.00027	90.78
PlagLinSVM [20]	90.55	89.75	90.15	1.00210	90.01
Oberreuter and Eiselt [33]	91.54	87.17	89.30	1.00051	89.27
Sanchez-Perez et al. [4]	91.98	86.61	89.21	1.00026	89.20
Glinos [40]	84.51	**96.01**	89.89	1.01761	88.77
PlagRbfSVM [20]	91.49	85.52	88.40	1.00209	88.27
Shrestha et al. [34]	89.84	84.42	87.05	1.00381	86.81
Gross and Modaresi [41]	81.82	92.52	86.84	1.02187	85.50
Rodríguez Torrejón and Martín Ramos [48]	80.27	90.03	84.87	1.00000	84.87
Kong et al. [39]	84.16	82.88	83.52	1.00000	83.51
Alvi et al. [53]	67.28	90.08	77.03	1.06943	73.42
Abnar et al. [55]	84.78	54.83	66.59	1.00455	66.38
Gillam and Notley [60]	29.66	85.74	44.07	1.00000	44.08

Bold values indicate to show the highest values converted to a 3D image and signal to appropriate the inputs of different deep learning architectures. The proposed intelligent system pre-processes the documents, selects the possible plagiarized cases, and computes the lexical, syntactic, and semantic similarity criterion values for each extracted case to create the training database. It also constructs the deep learning classifier and extracts the better segments of the plagiarized text by filtering and merging adjacent detected seeds of similar text sentences. A comparative study was implemented on the two benchmark datasets, that is, PAN 2013 and PAN 2014, to determine the most convenient deep learning architecture that can detect text plagiarism with the highest accuracy by fitting the values of the created training database. As per our findings, it was determined that the proposed system based on LSTM architecture achieved the highest accuracy compared to the state-of-the-art text plagiarism systems, which can weigh the fluctuations between the values of lexical, syntactic, and semantic similarity criteria of the suspicious cases.

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. "Council of Writing Program Administrators. (2003). Defining and avoiding plagiarism: The WPA statement on best practices. In Council of Writing Program Administrators. Retrieved from http://wpacouncil.org/files/wpaplagiarism-statement.pdf".
2. Stamatatos and Efstathios (2011) Plagiarism detection using stopword n-grams. J Am Soc Inform Sci Technol 62(12):2512–2527
3. Sánchez-Vega F, Villatoro-Tello E, Montes-y-Gómez M, Rosso P, Stamatatos E, Villaseñor-Pineda L (2019) Paraphrase plagiarism identification with character-level features. Pattern Anal Appl 22(2):669–681
4. Sanchez-Perez M, Sidorov G, and Gelbukh A, (2014) A winning approach to text alignment for text reuse detection at PAN 2014–notebook for PAN at CLE", 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, UK, pp 1004–1011
5. Roostaee M, Fakhrahmad SM, Sadreddini MH (2020) Cross-language text alignment: A proposed two-level matching scheme for plagiarism detection. Expert Syst Appl 160:113718
6. Sahi M, Gupta V (2017) A novel technique for detecting plagiarism in documents exploiting information sources. Cogn Comput 9(6):852–867
7. Ahuja L, Gupta V, Kumar R (2020) A new hybrid technique for detection of plagiarism from text documents. Arab J Sci Eng 45(12):9939–9952
8. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J. (2013) Distributed representations of words and phrases and their compositionality. Adva Neural Inform Proc Syst 26.
9. Conneau A, Douwe K, Holger S, Loic B, Antoine B. (2017) Supervised learning of universal sentence representations from natural language inference data, arXiv preprint arXiv:1705.02364
10. Pennington, J, Richard S, Christopher D Manning. (2014) Glove: Global vectors for word representation, In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1532–1543
11. 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 Appl 32(14):10593–10607
12. van Son N, Huong LT, Thanh NC (2021) A two-phase plagiarism detection system based on multi-layer lstm networks. IAES Int J Artif Intel 10(3):636–648
13. Frank M, Drikakis D, Charissis V (2020) Machine-learning methods for computational science and engineering. Computation 8(1):15
14. Song YL, Chen SS (2009) Text mining biomedical literature for constructing gene regulatory networks. Interdiscip Sci Comput Life Sci 1:179–186
15. Aggarwal CC (2015) Data mining. Springer, Cham
16. Kavitha T, Mathai PP, Karthikeyan C et al (2022) Deep learning based capsule neural network model for breast cancer diagnosis using mammogram images. Interdiscip Sci Comput Life Sci 14:113–129
17. Sah M, Direkoglu C (2022) A survey of deep learning methods for multiple sclerosis identification using brain MRI images. Neural Comput Appl. https://doi.org/10.1007/s00521-022-07099-3
18. Potthast M, Gollub T, Hagen M, Tippmann M, Kiesel J, Rosso P, Stamatatos E, and 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
19. Potthast M, Hagen M, Beyer A, Busse M, Tippmann M, Rosso P, and 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
20. Altheneyan AS, Menai MEB (2020) Automatic plagiarism detection in obfuscated text. Pattern Anal Appl 23(4):1627–1650
21. Miller GA (1995) WordNet: a lexical database for English. Commun ACM 38(11):39–41
22. Sapkota, U, Steven B, Manuel M, Thamar S, (2015) Not all character n-grams are created equal: A study in authorship

attribution, In: Proceedings of the 2015 conference of the North American chapter of the association for computational linguistics: Human language technologies, pp 93–102
23. Leacock C, Chodorow M (1998) Combining local context and WordNet similarity for word sense identification. WordNet Electron Lexical Database 49(2):265–283
24. Wu Z, Palmer M, Verbs Semantics and Lexical Selection, In: Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics, pp 133–138, 1994.
25. Resnik P (1995) Using information content to evaluate semantic similarity in a taxonomy, In: Proceedings of the 14th international joint conference on artificial intelligence, Vol. 1, 448–453, Montreal
26. Lin D (1998) An information-theoretic definition of similarity. In Icml 98(1998):296–304
27. Jay J, David CW (1997) Semantic similarity based on corpus statistics and lexical taxonomy, In: Proceedings of the 10th research on computational linguistics international conference, pp 19–33
28. Huang G, Zhuang L, Laurens Van Der M, Weinberger KQ (2017) Densely connected convolutional networks, In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4700–4708
29. Graves, A, Abdel-rahman Mohamed, Geoffrey H (2013) Speech recognition with deep recurrent neural networks, In: 2013 IEEE international conference on acoustics, speech and signal processing, pp 6645–6649. IEEE
30. Singh PK, Rahim N (2016) Normalization and transformation technique based privacy preservation in data mining. Int J Res 3:10–17
31. Chaising S, Temdee P, Prasad R (2021) Individual attribute selection using information gain based distance for group classification of elderly people with hypertension. IEEE Access 9:82713–82725
32. Potthast M, Stein B, Barrón-Cedeño A, Rosso P (2010) An evaluation framework for plagiarism detection. In: Coling 2010: Posters pp 997–1005
33. Oberreuter G, Eiselt A (2014) Submission to the 6th international competition on plagiarism detection, From Innovand. io, Chile, Available: https://www.uni-weimar.de/medien/webis/events/pan-14/pan14-web/
34. Shrestha P, Maharjan S, Solorio T (2014) Machine translation evaluation metric for text alignment, CLEF (working notes), pp 1012–1016, Available: https://pan.webis.de/downloads/publications/papers/shrestha_2014.pdf
35. Palkovskii Y, Belov A (2014) Developing high-resolution universal multi-type n-gram plagiarism detector", Conference and Labs of the Evaluation Forum and Workshop (CLEF'14), pp 984–989, Available: http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-PalkovskiiEt2014.pdf.
36. Vani K, Gupta D (2018) Unmasking text plagiarism using syntactic-semantic based natural language processing techniques: Comparisons, analysis and challenges. Inf Process Manage 54(3):408–432
37. 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
38. Leilei K, Haoliang Q, Cuixia D, Mingxing W, Zhongyuan H (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
39. Kong L, Han Y, Han Z, Yu H, Wang Q, Zhang T and Qi H, (2014) Source retrieval based on learning to rank and text alignment based on plagiarism type recognition for plagiarism detection, CLEF (working notes), pp 973–976, Available: http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-KongEt2014.pdf.
40. Glinos DG (2014) A hybrid architecture for plagiarism detection, CLEF (working notes). pp 958–965, Available: http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-Glinos2014.pdf
41. Gross P, Modaresi P (2014) Plagiarism alignment detection by merging context seeds, CLEF (working notes), pp 966–972, Available: https://pan.webis.de/downloads/publications/papers/gross_2014.pdf
42. Palkovskii Y, Belov A (2021) Applying specific clusterization and fingerprint density distribution with genetic algorithm overall tuning in external plagiarism detection, CLEF (working notes), Available: http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-PalkovskiiEt2012.pdf.
43. Rodríguez Torrejón D, Martín RJ (2014) CoReMo 2.3 plagiarism detector text alignment module, CLEF (working notes), pp 997–1003, Available: http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-RodriguezTorrejonEt2014.pdf
44. Suchomel Š, Kasprzak J, Brandejs M "Diverse (2013) Queries and feature type selection for plagiarism discovery, CLEF (working notes), Available: http://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-SuchomelEt2013.pdf
45. Oberreuter G, Carrillo-Cisneros D, Scherson I, Velásquez J, (2012) Submission to the 4th international competition on plagiarism detection, Available: http://www.uni-weimar.de/medien/webis/events/pan-12
46. Diego A. Rodríguez T, José M, Martín R (2013) Text alignment module in CoReMo 2.1 plagiarism detector, CLEF (working notes), Available: http://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-RodriguezTorrejonEt2013.pdf
47. Daud A, Khan JA, Nasir JA, Abbasi RA, Aljohani NR, Alowibdi JS (2019) Latent dirichlet allocation and POS tags based method for external plagiarism detection: LDA and POS tags based plagiarism detection. In: scholarly ethics and publishing: breakthroughs in research and practice, pp 319–336. IGI Global
48. Torrejón DA, Ramos JM (2012) Detailed Comparison Module In CoReMo 1.9 Plagiarism Detector, CLEF (working notes), Available: http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-RodriguezTorrejonEt2012.pdf
49. 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
50. Saremi M and Yaghmaee F (2013) Submission to the 5th international competition on plagiarism detection, Available: http://www.uni-weimar.de/medie n/webis/events/pan-13
51. Suchomel S, Kasprzak J, Brandejs M (2012) Three way search engine queries with multi-feature document comparison for plagiarism detection, CLEF (working notes), Available: http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-SuchomelEt2012.pdf
52. Robin K, Conrad S (2012) A set-based approach to plagiarism detection, CLEF (working notes), Available: http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-KuppersEt2012.pdf
53. Alvi F, Stevenson M, Clough P (2014) Hashing and merging heuristics for text reuse detection, CLEF (working notes), pp 939–946, Available: http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-AlviEt2014.pdf
54. Yurii P, Alexei B (2013) Using hybrid similarity methods for plagiarism detection, CLEF (working notes), Available: http://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-PalkovskiiEt2013.pdf
55. Abnar S, Dehghani M, Zamani H and Shakery A (2014) Expanded N-grams for semantic text alignment, CLEF (working notes), pp 928–938, Available: http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-AbnarEt2014.pdf

56. Sánchez-Vega F, Montes-y-Gómez M, Pineda LV (2012) Optimized fuzzy text alignment for plagiarism detection, CLEF (working notes), Available: http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-SanchezVegaEt2012.pdf
57. Nourian A (2013) Submission to the 5th international competition on plagiarism detection, Available: http://www.uni-weimar.de/medien/webis /events/pan-13
58. Jayapal A, Goswami B, Vector space model and overlap metric for author identification, CLEF (working notes), 2013, Available: http://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-JayapalEt2013.pdf
59. Lee G (2013) Guess again and see if they line up: Surrey's runs at plagiarism detection, CLEF (working notes), Available: http://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-Gillam2013.pdf
60. Gillam L, Notley S (2014) Evaluating robustness for 'IPCRESS': Surrey's text alignment for plagiarism detection, CLEF (working notes), pp 951–957, Available: http://ceur-ws.org/Vol-1180/CLEF2014wn-Pan-GillamEt2014.pdf.