*Journal of **AI** and **D**ata **M**ining*
Vol 7, No 3, 2019, 451-466

DOI: 10.22044/JADM.2018.4328.1517

External Plagiarism Detection based on Human Behaviors in Producing Paraphrases of Sentences in English and Persian Languages

A. Shojaei[1] and F. Safi-Esfahani[2*]

[1,2]Faculty of Computer Engineering, Najafabad Branch, Islamic Azad University, Najafabad, Iran.
[1,2]Big Data Research Center, Najafabad Branch, Islamic Azad University, Najafabad, Iran.

Received 21 July 2016; Revised 30 May 2017; Accepted 04 November 2017
**Corresponding author: fsafi@iaun.ac.ir (F. Safi-Esfahani).*

Abstract
With the advent of the internet and easy access to digital libraries, plagiarism has become a major issue. Applying search engines is one of the plagiarism detection techniques that converts plagiarism patterns to search queries. Generating suitable queries is the heart of this technique, and the existing methods suffer from the lack of producing accurate queries, Precision, and Speed of retrieved results. This research work proposes a framework called ParaMaker. It generates accurate paraphrases of any sentence, similar to human behaviors, and sends them to a search engine to find the plagiarism patterns. For the English language, ParaMaker is examined against six known methods with standard PAN2014 datasets. The results obtained show an improvement of 34% in terms of the Recall parameter, while the parameters Precision and Speed are maintained. In the Persian language, statements of suspicious documents are examined compared to an exact search approach. ParaMaker shows an improvement of at least 42% in Recall, while Precision and Speed are maintained.

Keywords: *Plagiarism Detection, External Plagiarism Detection, Resource Retrieval, Sentence Paraphrase Producing.*

1. Introduction

A plagiarizer uses the ideas of others and attributes them to himself without referencing to their names [1]. In the digital age, many existing resources have been added to the web, which makes the plagiarism problem even worse. For a small set of documents, a one by one comparison of suspicious documents with each source document seems reasonable but this approach is not applicable to a large set of documents on the web. An existing solution for this problem is to use the resource retrieval techniques that apply search engines to retrieve the potential sources of plagiarism for a suspicious document. The plagiarism detection methods mostly use the parameters Precision and Recall for evaluation. Recall mentions that if a document is theft, it is surely retrieved, while Precision says that if a document is retrieved, then it must be theft.

The most important matter is to produce and submit suitable queries in order to find plagiarism patterns and obtain accurate results [2]. Producing inefficient queries is a common problem among the methods that use the retrieving techniques that result in obtaining a weak Recall parameter. Some methods try to improve the Recall parameter; however, this improvement leads to a sharp reduction in the parameters Precision and Speed. In this research work, the hypothesis is that producing paraphrases of sentences based on human behaviors generates more accurate query results in retrieving more suitable documents by search engines. As a result, the Recall parameter is improved, while the parameters Precision and Speed are maintained. This article aims at improving the quality of queries to improve the plagiarism detection. A framework is presented, namely ParaMaker, along with several algorithms to make suitable queries. The results obtained show that the Recall parameter is improved up to 34% compared to the highest values in the PAN2014 competition, while the parameters Precision and

Speed are maintained in acceptable levels. The scope of this paper is limited to the English and Persian documents. Moreover, a source document must be inspected in terms of plagiarism, whether the sentences are marked by references or not.

The rest of this paper is organized as what follows. Section 2 introduces the research works done in the field of resource retrieval. Section 3 describes the proposed method in detail. Section 4 provides information on the conducted experiments. The outcomes of the experiment are reported in Section 5. Section 6 includes the concluding remarks.

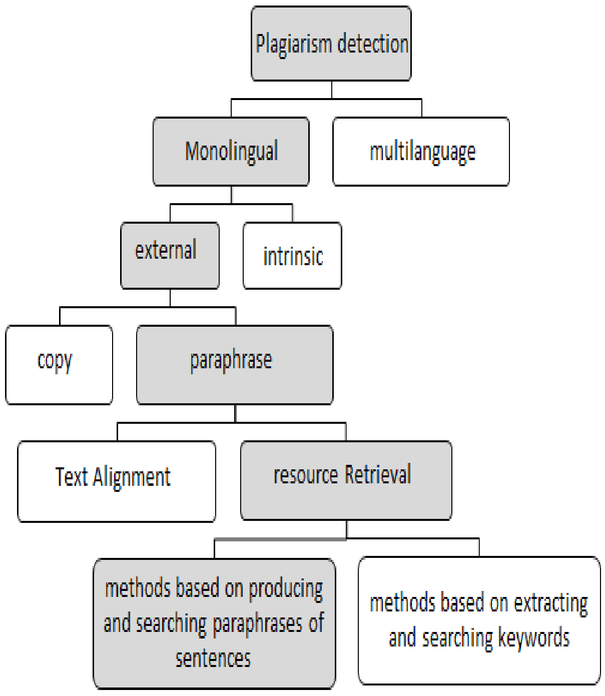
Plagiarism detection
Monolingual
multilanguage
external
intrinsic
copy
paraphrase
Text Alignment
resource Retrieval
methods based on producing and searching paraphrases of sentences
methods based on extracting and searching keywords

Figure 1. Taxonomy of plagiarism detection techniques.

2. Literature review

Many techniques have been developed for plagiarism detection in natural languages, classified in figure 1. The highlighted area illustrates the mainstream of this research work.
Monolingual plagiarism detection refers to identifying plagiarism in a homogeneous language environment, while multilingual plagiarism detection means the plagiarism between two or more languages. In external plagiarism, suspicious documents are compared with a collection of documents, while intrinsic plagiarism detects plagiarism against checking only one document itself [3]. Moreover, plagiarism is classified into literal and intelligent based on the behavior of plagiarizers. In the literal plagiarism, a plagiarizer does not spend a lot of time to hide his academic offense, for instance, simply copying the text from the Internet, while in the intelligent plagiarism, a plagiarizer hides, obfuscates, and changes the original work with intelligent methods such as manipulating and translating texts in order to deceive readers [3, 4].

One of the most important types of plagiarism is paraphrasing, which means expressing the same ideas or contents with words different from the source documents [5]. There are several methods available to detect paraphrasing known as text alignment techniques. They compare a suspicious document with the source documents one by one. Applying these methods is not justified for large datasets such as the Internet. Recently, search engines are used for comparison of a suspicious document with the source documents. Table 1 shows the main differences between the two groups. The research works carried out on plagiarism detection based on resource retrieval are presented as what follow. In [6], a ranking system is presented, in which each word is assigned a weight (tf.idf[1]) that is the frequency of each word in a document. In addition, three different strategies are presented in making queries: 1) The words in a 50-line chunk are scored by tf.idf, and then the top ten words are picked up to make a search query. 2) In each 50-line chunk, the first 8-gram that includes at least three words of the top ten words are chosen to make the search query. 3) Headers are considered as independent chunks [7]. Noun phrases in headers are then scored and ranked. Subsequently, the fifteen top ranked phrases are used in making the search query. This method focuses on the top ten results obtained from the search query. Only those documents are retrieved that 90% of their 4-grams in a 500-character chunk are available in a suspicious document.

[1]Term frequency–inverse document frequency

Table 1. Comparison between resource retrieval and text alignment techniques.

Resource retrieval	Text alignment
Applies search engines	Applies methods to find similarity of texts
Used on the web and large-scale datasets	Used only on a small scale and local datasets
Faster	Slower
More error prone	Less error prone

Authors in [8] have selected five-sentence sections and have chosen ten phrases with the highest ranking. In this way, there are four methods to extract the keywords that include BM25[1] [9], tf[2], tf.idf, and EW[3] [10]. In this method, for ranking the extracted keywords, a SVM-based ranking model is used. Finally, to build queries for each section of the suspicious document, the best group of keywords is selected. Then for each query, three results with the highest ranks are retrieved.

Authors in [11] have broken down a document into (100 or 200)-word sections. There exists the following rules to choose suitable terms in making search queries: 1) five top-rank terms ranked by term frequency[4] in the document level, 2) five top-rank terms ranked by term frequency in the paragraph level, 3) sub-group names for each sentence. The keyword extraction method is based upon two well-known strategies including term frequency and co-occurrence words. The role of words is determined based on the maximum entropy part-of-speech tagger [12], and all names are extracted as keywords. For each section, at least four queries are made. Before sending the query, it must be ensured that 60% of the queries are different from the previously submitted queries.

Method [13] divides a text into five-sentence sections. This method uses a very simple strategy to extract the key phrases so that only nouns, adjectives, and verbs are included. This method classifies the above sections based on various features in order to retrieve the results by several search engines.

The research work [14] divides a suspicious document into weighted sentences. This weight is determined according to the amount of overlaps with other sentences. One problem with the weighted sentences is that the selected sentences are not distributed in the whole document; therefore, certain parts of the document may be neglected. Based upon the first seven pieces returned by the search engine and the similarity of available sentences, the similarity of sentences in a suspicious document is calculated. Finally, the document with the highest similarity is retrieved. After removing the articles, pronouns, prepositions, and high-frequency words from the highest ranked sentence, six entities (phrases and words) of the most weighted entities in each sentence are extracted and used in making queries.

In [25], three types of text chunking are applied: sentence and word chunking for keyword extraction, paragraph chunking, and header chunking. Several types of queries are eventually prepared, which are keyword-based, paragraph-based, and header-based queries. All queries are processed according to their priority. By processing the retrieved results, the positions of the discovered similarities are stored. Prior to the full document download, a snippet can be retrieved. It contains a portion of the document up to 500 characters around a given textual string.

Table 2 illustrates a comparison between the resource retrieval methods presented in the shaded area of figure 1. The methods are compared in terms of the segmentation method, the way of making queries, search control, and retrieval filters along with mentioning advantages and disadvantages of each method.

3. Proposed approach

Document retrieval techniques have a higher speed, and are more scalable than the other paraphrase detection methods. The Recall parameter plays an important role in the evaluation of these techniques, and negatively document retrieval methods show a weak Recall either. The ParaMaker framework presented in this section tries to make paraphrases of a sentence based on

[1]BM25 ranking function is based on a 2-Poisson model of term frequencies in documents
[2]Term frequency
[3]Evaluating words
[4]Number of times a term occurs in a document is recalled its term frequency

the human behaviors in generating new sentences from one sentence. After generating paraphrases from one sentence, the search queries are made and submitted to a search engine. It is expected that this method will increase the Recall parameter along with maintaining the Speed and Precision quality factors. According to figure 2, the ParaMaker framework includes five phases: preprocessing, segmentation, query generation, searching control, and filtering of retrieved documents. Innovations of this research work fall in the query generation and search control phases.

Table 2. Comparison of resource retrieval methods.

Research	Text segmentation method	Query making method	Search control and retrievals filter method	Advantages and Disadvantages
Elizalde [6]	-Divides the text into sections of 50 linear	-Using Coefficient (tf.idf) document frequency or identifying nominal phrases in making queries	- Focuses on the top ten results of each query	-Very good runtimes - Relatively low Recall
Kong [8]	- Divides the text into sections of 5 sentences	-Selection of the ten best phrases in each section based on the keywords extraction method BM25 and weighted tf.idf -A rating model based on SVM ranking for grouping keywords in ranking queries	-retrieve stop three results per query	- Low Precision - High number of generated queries -High Recall
Prakash [11]	- Divides the text into sections of 100 words based on title detection	-Selects five terms with the highest rank in the document level as well as five terms in the paragraph level. - Extracts keywords based on two well-known strategies, long term frequency and co-occurrence words	-Retrieve a document from the ten highest results when at least one to five-grams of 500 characters are in a suspected document	-Relatively good runtime - Relatively low Recall
Suchomel [25]	-Considers titles as separate sections - Divides the documents into sections using titles	-Using three different strategies to generate queries based on keywords, paragraphs, and titles	-Deletes duplicated similar positions -Uses snippet of 500 characters for each query	-Very low Precision -High number of generated queries
Williams [13]	- Divides a suspected documents into paragraphs with 5sentences	-The formation of key phrases by nouns, adjectives, and verbs	-Trains a classifier to retrieve features using several search engines	-The highest Precision -High number of generated queries
Zubarev [14]	- Divides the suspected document to formatted sentences	-Forms queries by deleting articles, pronouns, prepositions, and repeated words -Choices six entities (phrases or words) from the main (most weighted) entities	- Removes queries that are potentially mapped to retrieved resources	-Low number of generated queries
ParaMaker (This Paper)	-Divides the text into sentences	- Use paraphrases of a sentence based on human behavior in making queries	- Exact sentence searching -Use proximity parameter for words of each sentence - Retrieve the highest result per query	-Very high Recall parameter -Acceptable Precision and speed parameters -Generating more efficient and more precise queries -Retrieving relevant documents -High number of generated queries

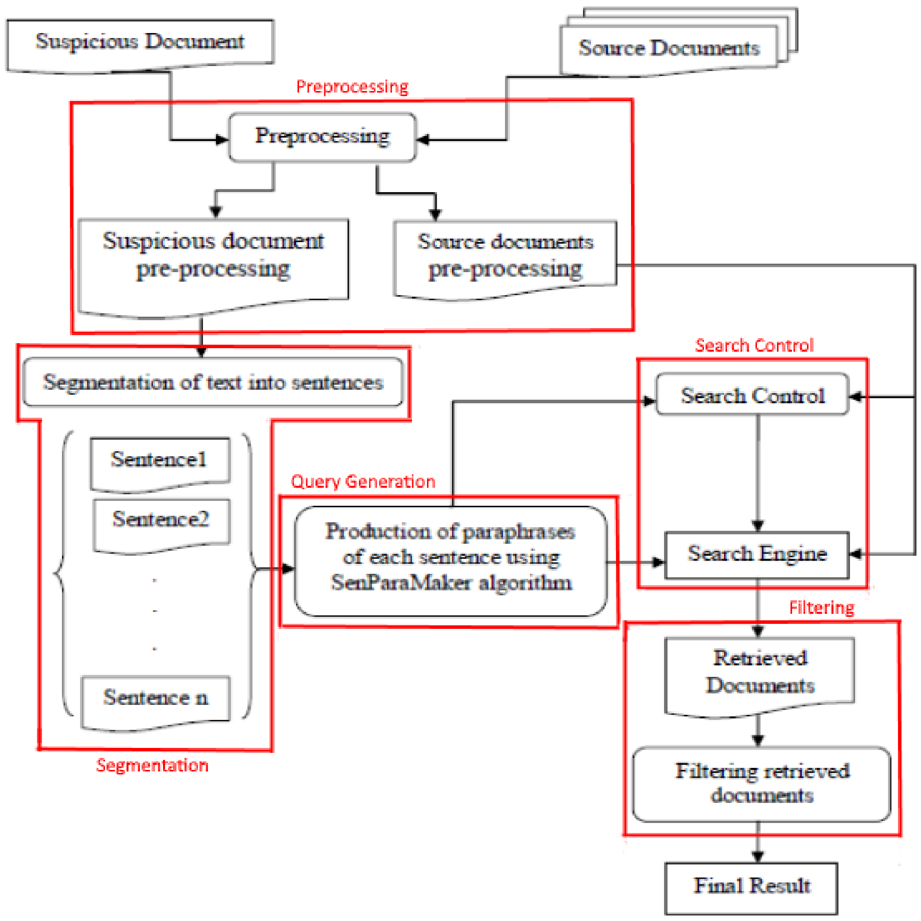
Suspicious Document
Source Documents
Preprocessing
Preprocessing
Suspicious document pre-processing
Source documents pre-processing
Segmentation of text into sentences
Sentence1
Sentence2
Sentence n
Segmentation
Query Generation
Production of paraphrases of each sentence using SenParaMaker algorithm
Search Control
Search Control
Search Engine
Filtering
Retrieved Documents
Filtering retrieved documents
Final Result

Figure 2. Proposed ParaMaker framework.

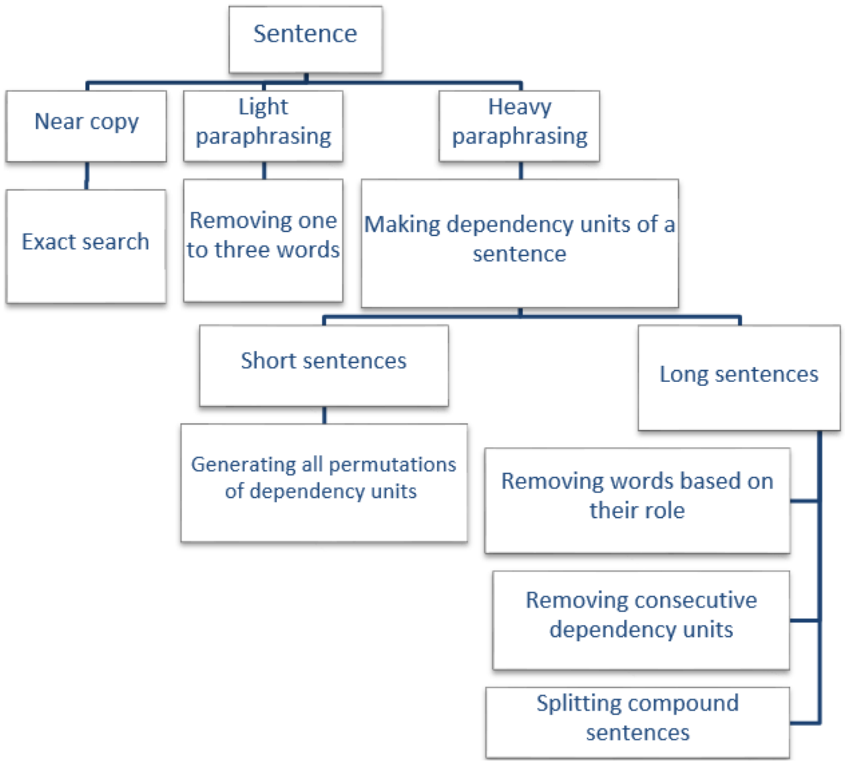
Sentence
Near copy
Light paraphrasing
Heavy paraphrasing
Exact search
Removing one to three words
Making dependency units of a sentence
Short sentences
Long sentences
Generating all permutations of dependency units
Removing words based on their role
Removing consecutive dependency units
Splitting compound sentences

Figure 3. Various Paraphrases of sentences in ParaMaker.

3.1. Phase I: Preprocessing

Preprocessing performs suitable operations to enhance a text so that the plagiarism detection algorithms take better results (precision, speed, etc.). The preprocessing phase for the English language involves the following steps [15, 16]: 1) dividing the text into segments; 2) replacing the numbers; 3) replacing tabs and new lines with space characters; 4) keeping letters and converting all notations, numbers, etc. to spaces; 5) converting all letters to lowercase; 6) replacing the juxtaposed space characters with one space character; 7) deleting the words that are of less than three letters; 8) removing common words; 9) stemming the remaining words using an algorithm such as the Porter's stemming algorithm [17].

3.2. Phase II: Segmentation

In this phase, the context of a suspicious document is divided into its constituent sentences. Sentence diagnostic tools should pay attention to the separator characters in order to recognize sentences in an input text. The precise output of algorithms in many languages depends upon the tools applied in this phase.

3.3. Phase III: Query generation based on human behaviors

In this phase, various paraphrases of a sentence are generated and submitted to a search engine as several search queries. As it is usually the case, a plagiarizer, as a human, follows several steps in paraphrasing, as follows: 1) rearranging or changing the order of words in sentences or phrases; 2) removing words or phrases in sentences; 3) adding words or phrases; 4) replacing words or phrases with their synonyms; 5) splitting or combining sentences [18].

Producing paraphrases of sentences: According to figure 3, the paraphrasing of a sentence falls into the following categories 1) near copy; 2) light paraphrasing; 3) heavy paraphrasing.

Near copy includes nuance changes in a sentence that is the result of the preprocessing phase. These changes include removing or adding general words, replacing synonyms of general words, partial changes in words and verbs or adding/removing symbols/marks in a sentence.

Light paraphrasing includes removing one/more words, adding one/more sequential words, replacing words with their synonyms, and changing the order of words or phrases. In the recommended SenParaMaker algorithm, generating light paraphrasing is analogous to removing at most three sequential words provided that the remaining words in a sentence do not become less than three words. A sentence with less than three words is a short phrase, and searching for short phrases causes a wrong detection. Replacing the synonym words is also done by search engines in the search control phase.

Heavy paraphrasing includes splitting complex sentences, deleting one or more word(s)/phrase(s), adding one/more word(s)/phrase(s), and permutation of word(s)/phrase(s). Initially, after using a dependency parser [19], a sentence is analyzed and partitioned into its dependency units that will be explained more in the sequel. From now on, the removal operation is performed on the dependency units instead of words. Removing each word from a sentence may result in producing trivial sentences. It prevents producing meaningless paraphrases, and bypasses unnecessary processing, which, consequently, speeds up the detection process.

According to [20], basically, there are two assumptions in producing the dependency unit: 1) each sentence includes one central verb; 2) based on the type and number of mandatory/arbitrary complements of the central verb, determining the fundamental structures that are built on this verb is possible. In connecting two words by a dependency relation, one word is root and another one is dependent. In order to pars the dependency of phrases, each element in a sentence must be tagged as a root or dependent element. It means that the dependents of each root element and the root element of each dependent must be determined. Figure 4 shows the root/dependent words in the sentence "economic news had little effects on financial markets".

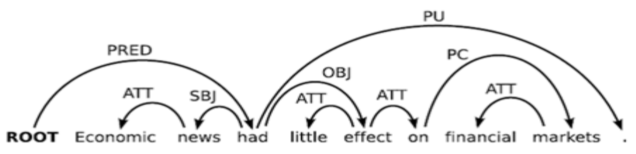

Figure 4. Root/Dependent elements in a sample sentence.

At this stage, as long as the sentence length is not less than half, removing the sequential dependency units is continued. The pseudo-code of the heavy paraphrase detection algorithm is shown in figure 5. At first, a sentence is broken into its dependency units. If the number of dependency units is less than or equal to five, the sentence is considered as a short sentence and the function smallSentence is invoked, while the sentences with a dependency unit larger than five are considered as long sentences and function largeSentence is called, consequently. A long sentence might be a compound sentence that is analogous to dividing the sentence into its constituent sentences S1 and S2. If the number of words in S1 or S2 is less than five, the sentences are considered as short phrases.

```
Function HeavyParaphrase (sentence);
dpnUnit ←DependencyUnit (Sentence);
if count of dpnUnit > 6 then
      LargeSentence (dpnUnit);
      isCompnd ←isCompund(sentence);
      if isCompnd = true then
         S1, S2 ← Split(Sentence);
            if S1.size() < 5 OR S2.size() < 5 then
            delete S1 OR S2;
      else
         if count of S1.dpnUnit > 6 then
            LargeSentence(S1.dpnUnit);
         else
            SmallSentence(S1.dpnUnit);
         end else;
         if count of S2.dpnUnit > 6 then
            LargeSentence(S2.dpnUnit);
         else
            SmallSentence(S2.dpnUnit);
         end else;
      end else;
   end if;
else
   SmallSentence (dpnUnit);
end else;
end function;
```

Figure 5. Heavy Paraphrase detection algorithm.

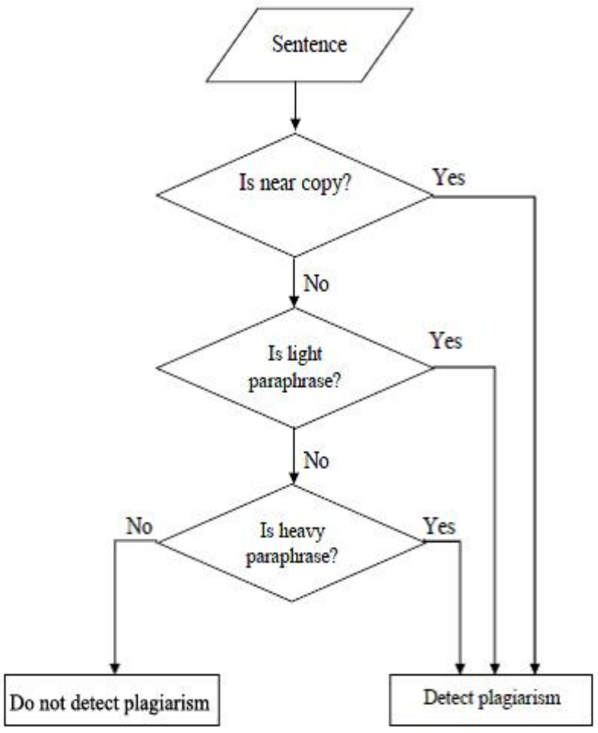

Figure 6.Cancel SenParaMaker flowchart to detect plagiarism of sentences

Figure 6 shows the whole SenParaMaker algorithm that detects the plagiarism of a sentence. Producing paraphrases by changing the order of words in sentences occurs by setting the proximity parameter, which is described in the next phase. Synonym replacements can be automatically done by the search engine during the search operation.

3.4. Phase IV: Search control

In this phase, the generated paraphrases for each sentence are submitted to the search engine as queries along with the proximity parameter. This parameter determines irregularities of words and phrases to the sentences of an original document. It finds words that are within a certain distance from each other, regardless of the order in which they appear. For example, an exact sentence match has a proximity parameter equal to zero, and a word transposition (such as “bar foo”) has a proximity parameter equal to one. Searching the term "foo bar" with the proximity parameter equal to one has the capability of detecting the phrase "bar foo" but is not able to detect the "bar text foo" because the words "foo" and "bar" are placed at least within two words of each other. If the proximity parameter is set to two or more, it is analogues to the term "foo bar". It is remarkable that query searching along with the proximity value is a type of exact search, and ensures that the words within the query terms are searched based upon their proximity value.

Replacing words with synonyms: People attempt to hide plagiarism by replacing the words with appropriate synonyms. Synonym detection is applied to handle this situation. In this phase, words in every sentence are replaced by their synonyms along with setting the proximity parameter of the search engine, simultaneously. The proximity parameter for each paraphrase is considered as equal as the length of sentences. Queries with the proximity parameter are expected to have a higher rank compared to the words that are merely close together. The synonyms of words (without stop words) are identified and replaced using a dictionary. In this research work, to achieve a higher precision, the synonyms of a word are replaced entirely.

3.5. Phase V: Filtering retrieved documents

It is usually the case that query searching retrieves several documents, some of which are not source documents, mistakenly. It drastically reduces the precision of plagiarism detection. This is the reason why ParaMaker framework considers the top retrieved document. In order to avoid creating intricate filters, ParaMaker tries to make more accurate queries. The default ranking algorithm considers the Cosine similarity between two documents. Certainly, setting up the proximity value between search queries and source documents affects ranking of results. In addition, documents included in the results of the previous search queries are not considered again.

4. Case study

Suppose a suspicious document that contains the following text:
"The main purpose of this study is to check acceptance factors of this advertisement through this new technology. Joan Bynvayt from America won the gold medal of women's marathon... | Cool running is a complete resource for runners, offering a race calendar, race results listings!"

Table 3 includes the original sentences along with their equivalent suspicious sentences and the plagiarism type as well (related to the above text).

Table 3. Case study of ParaMaker.

Sentence-1	
Original sentence	"checking and crititic of the acceptanance factors of advertisment through novel material, are purpose of this research."
Suspecious	"the main purpose of this study,is checking acceptance factors of this advertisment through this new technology."
Type of plagiarism	Light paraphrase
Sentence-2	
Original sentence	*"Joan Bynvayt won* gold *medal* of *marathon*."
Suspicious sentence	"*Joan Bynvayt from America won* gold *medal* of *women's marathon...*"
Type of plagiarism	Heavy paraphrase-short sentence
Sentence-3	
Original sentence	*"Cool Running is a complete resource for runners, offering a race calendar, race results listings*."
Suspicious sentence	"|*Cool running is a complete resource for runners, offering a race calendar, race results listings!*"
Type of plagiarism	Near Copy

The following sections illustrate how to detect plagiarized sentences as well as how to retrieve the resources that contain the original sentences.

Text preprocessing: Having been preprocessed, the above text may seem as follows: "main purpose study check acceptance factor advertisment technology. joan bynvayt america win gold medal women marathon. cool run complete resource run offer race calendar race result list."

Segmentation of text into its sentences: Table 4 shows the above text that is segmented into sentences.

Generating queries: Table 5 shows the queries generated for each sentence.

Search control: In this phase, the entire sentences of a suspicious document along with paraphrases of the sentences are sent to a search engine as queries. Replacing synonyms and setting the proximity parameter are of crucial importance in making suitable queries. Table 6 shows the Search control phase.

In Sentence-1 (suspicious text) the word "study" is replaced by "research", the word "technology" is replaced by "material", and the word "new" is replaced by "novel". Detection plagiarism requires replacing words with their synonyms, which is done in the search engine applying a dictionary during search operation for each sentence.

The term "the main purpose of this study" is the case that uses relocation in addition to synonym replacement. This relocation is also recognized in the search control phase by setting the proximity parameter of the search engine for each sentence, separately. Figure 7 shows steps taken in the case study of ParaMaker. The rest of the sentences in table 3 are processed like Sentence-1.

Table 4. Segmentation of text into its sentences.

Sentence-1	**"main purpose study check acceptance factor advertisment technology."**
Sentence-2	"joan bynvayt america win gold medal women marathon."
Sentence-3	"cool run complete resource run offer race calendar race result list."

Table 5. Generating queries.

Sentence-1	
Sentence-1	"main purpose study check acceptance factor advertisment technology"
Near copy	"main purpose study check acceptance factor advertisment technology"
Light paraphrases	"purpose study check acceptance factor advertisment technology", "study check acceptance factor advertisment technology", "check acceptance factor advertisment technology"
Sentence-2	
Sentence-2	"joan bynvayt america win gold medal women marathon"
Near copy	"joan bynvayt america win gold medal women marathon"
Light paraphrases	"bynvayt america win gold medal women marathon", "america win gold medal women marathon", "win gold medal women marathon", "joan america win gold medal women marathon", "joan win gold medal women marathon", "joan gold medal women marathon", "joan bynvayt win gold medal women marathon", "joan bynvayt gold medal women marathon", "joan bynvayt medal women marathon", "joan bynvayt america gold medal women marathon", "joan bynvayt america medal women marathon", "joan bynvayt america women marathon", "joan bynvayt america win medal women marathon", "joan bynvayt america win women marathon", "joan bynvayt america win marathon", "joan bynvayt america win gold women marathon", "joan bynvayt america win gold marathon", "joan bynvayt america win gold", "joan bynvayt america win gold medal marathon", "joan bynvayt america win gold medal"
Dependency units	joan bynvayt/america/win gold medal/women marathon
Heavy paraphrases (short sentence)	"america win gold medal women marathon", "win gold medal women marathon", "america win gold medal", "joan bynvayt win gold medal women marathon", "joan bynvayt women marathon", "joan bynvayt win gold medal", "joan bynvayt america women marathon", "joan bynvayt america win gold medal"
Sentence-3	
Sentence-3	"cool run complete resource run offer race calendar race result list"
Near copy	"cool run complete resource run offer race calendar race result list"

Table 6. Search control.

Sentence-1	**"the main purpose of this study, is checking acceptance factors of this advertisment through this new technology."**
Main sentence	"checking and crititic of the acceptanance factors of advertisment through novel material, are purpose of this research."

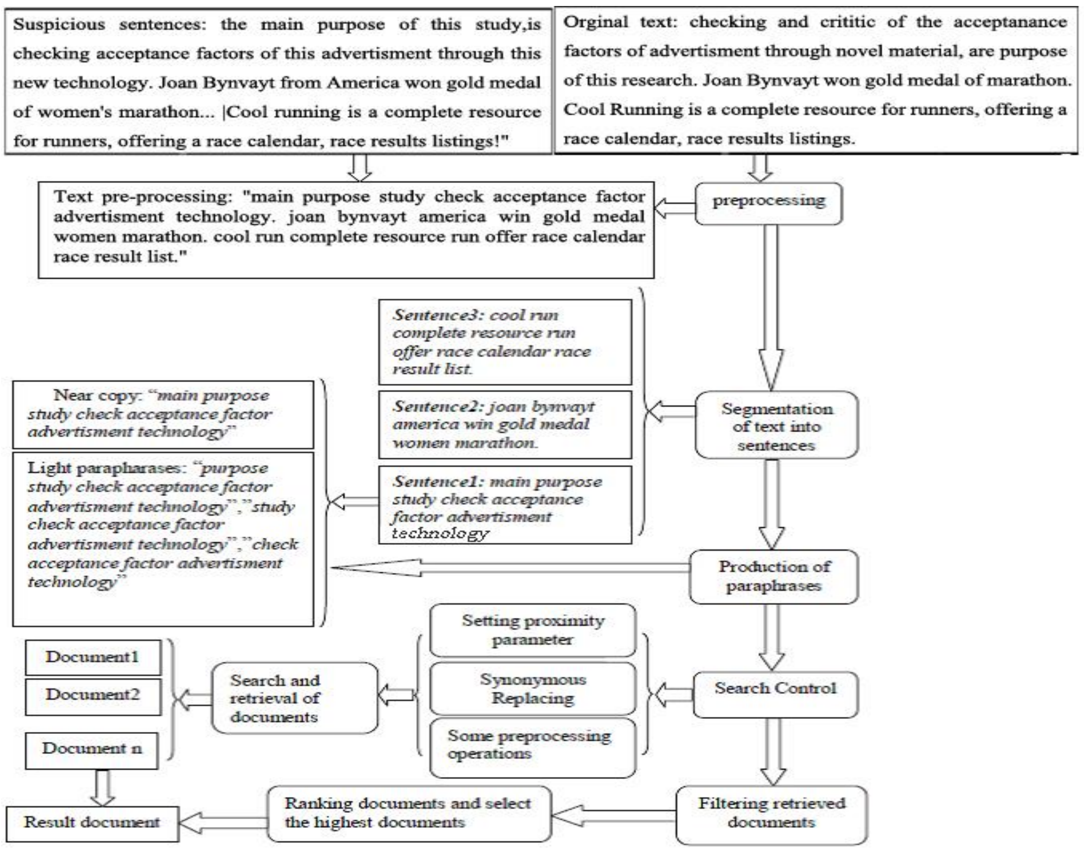
Suspicious sentences: the main purpose of this study,is checking acceptance factors of this advertisment through this new technology. Joan Bynvayt from America won gold medal of women's marathon... |Cool running is a complete resource for runners, offering a race calendar, race results listings!"
Orginal text: checking and crititic of the acceptanance factors of advertisment through novel material, are purpose of this research. Joan Bynvayt won gold medal of marathon. Cool Running is a complete resource for runners, offering a race calendar, race results listings.
Text pre-processing: "main purpose study check acceptance factor advertisment technology. joan bynvayt america win gold medal women marathon. cool run complete resource run offer race calendar race result list."
preprocessing
Sentence3: cool run complete resource run offer race calendar race result list.
Sentence2: joan bynvayt america win gold medal women marathon.
Sentence1: main purpose study check acceptance factor advertisment technology
Segmentation of text into sentences
Near copy: "main purpose study check acceptance factor advertisment technology"
Light paraphrases: "purpose study check acceptance factor advertisment technology","study check acceptance factor advertisment technology","check acceptance factor advertisment technology"
Production of paraphrases
Setting proximity parameter
Synonymous Replacing
Some preprocessing operations
Search Control
Search and retrieval of documents
Document1
Document2
Document n
Result document
Ranking documents and select the highest documents
Filtering retrieved documents

Figure 7. Case study of ParaMaker.

5. Evaluation

In this section, the datasets applied in the experiments for both the English and Persian languages are introduced first, and then two categories of experiments are examined for both languages.

5.1. Datasets

Datasets for English language experiments: The evaluations of this study are based upon PAN2014 competition that is accessible from Webis-TRC-13[21]. A collection of web documents in ClueWeb2009 is provided in 145 topics that are manually searched. Each of the produced suspicious documents has a set of source documents (D_{src}). Source documents include several series of phrases so that any phrase contains at least 50 long words, and each document contains at least one term. In addition, the terms are not repetitive, and each term is also extracted from a set of terms.

In order to mimic human behaviors in paraphrasing, each term is obfuscated, and then a suspicious document is provided by attaching the obfuscated terms entirely. Suspicious documents are finally paired with their peers in the source documents.

Some documents are produced in order to form the sample documents without plagiarism. The dataset contains 3653 suspicious documents as well as 4774 source documents. Suspicious documents are divided into three categories: 1) no obfuscation that applies exact or near copies, 2) random obfuscation, and 3) no plagiarism. The aim of random obscurity is to study whether or not the detection algorithm is able to detect the reused terms.

Datasets for Persian language experiments: In order to evaluate ParaMaker for the Persian language, three datasets are used, as follow: 1) TMC (Tehran Monolingual Corpus) has been created by University of Tehran, and includes the news extracted from Hamshahri newspaper and ISNA news agency. TMC is a huge monolingual corpus that contains 1000 source documents and 400 suspicious documents; 2) IranDoc that is an Iranian research center responsible for gathering scientific documents especially students' theses. IranDoc has prepared 230 documents along with 220 suspicious documents; 3) Prozhe.com is a website in which 440 source documents along with 160 suspicious documents are chosen and created based on several scientific reports and students' papers.

According to [22], two document sets are created for detecting plagiarism, specifically. Five categories of queries are produced in these datasets, which are featured as follow: 1) synonym replacement; 2) structural changing; 3) synonym replacement and structural changing; 4) removing words; and 5) adding words.

5.2. Evaluation criteria

Performance of resource retrieval algorithms for any suspicious document is evaluated using the following five criteria [2, 23]: 1) number of submitted queries; 2) number of retrieved documents; 3) Precision; 4) Recall, and 5) average runtime in minutes. Suppose that a suspicious document contains phrases from a text including the terms that have been plagiarized from a set of source documents D_{src}, and D_{ret} represents a set of retrieved documents returned by a retrieval algorithm. Precision and Recall are calculated based on the following equations (1) and (2). Finally, in order to measure the cost-effectiveness of a source retrieval algorithm in retrieving D_{ret} set, workload is also calculated in terms of the number of queries and retrievals.

$$\text{Precision} = \frac{|D_{\text{ret}} \cap D_{\text{src}}|}{|D_{\text{ret}}|} \quad (1)$$

$$\text{Recall} = \frac{|D_{\text{ret}} \cap D_{\text{src}}|}{|D_{\text{src}}|} \quad (2)$$

6. Experiments and experimental setup

The presented ParaMaker framework is examined for both the English and Persian languages. For retrieving documents, Apache Solr[5] that is an open source search engine is applied. PAN2014 uses the Indri[6] search engine for searching queries; while ParaMaker applies Solr. In [24], both Solr and Indri were compared in 2009 and 2012. According to table 7, indexing time in Indri is quicker than Solr, and size of produced index file is less in Indri. Precision of retrieving and ranking of documents in Indri is better than precision of Solr. However, for heavy queries, the speed of searching documents in

[5] http://lucene.apache.org/solr/

[6] http://www.lemurproject.org/indri/

Solr is ten times quicker than Indri. That is the reason why the experiments that compare searching time are ten folded to be comparable with Indri.

Table 7. Comparison of search time in Solr and Indri.

	2009		2012	
Search Engine	Indri4.1	Solr1.4	Indri5.3	Solr3.6
Light Queries (Second)	10	13	10	13
Number of Light Queries/Second	0.07	0.09	0.07	0.09
Heavy Queries (Second)	200	25	251	25
Number of Heavy Queries/Second	1.33	0.16	1.67	0.16

An illustration of experiment environment is shown in figure 8, in which every component of the ParaMaker framework is implemented by a suitable tool. In addition, several experiments are considered for evaluating the ParaMaker framework in both the English and Persian languages, as shown in table 8. Experiment-1 studies the effects of preprocessing on ParaMaker framework. Experiment-2 measures Precision and Recall parameters for exact, light, and heavy paraphrasing. Experiment-3 studies the relation between the number of retrieved documents and Precision and Recall parameters for each query. Expriment-4 compares the performance of the ParaMaker and PAN2014 participants. Finally, Experiment-5 is performed on three datasets in the Persian language, and studies the parameters Precision and Recall.

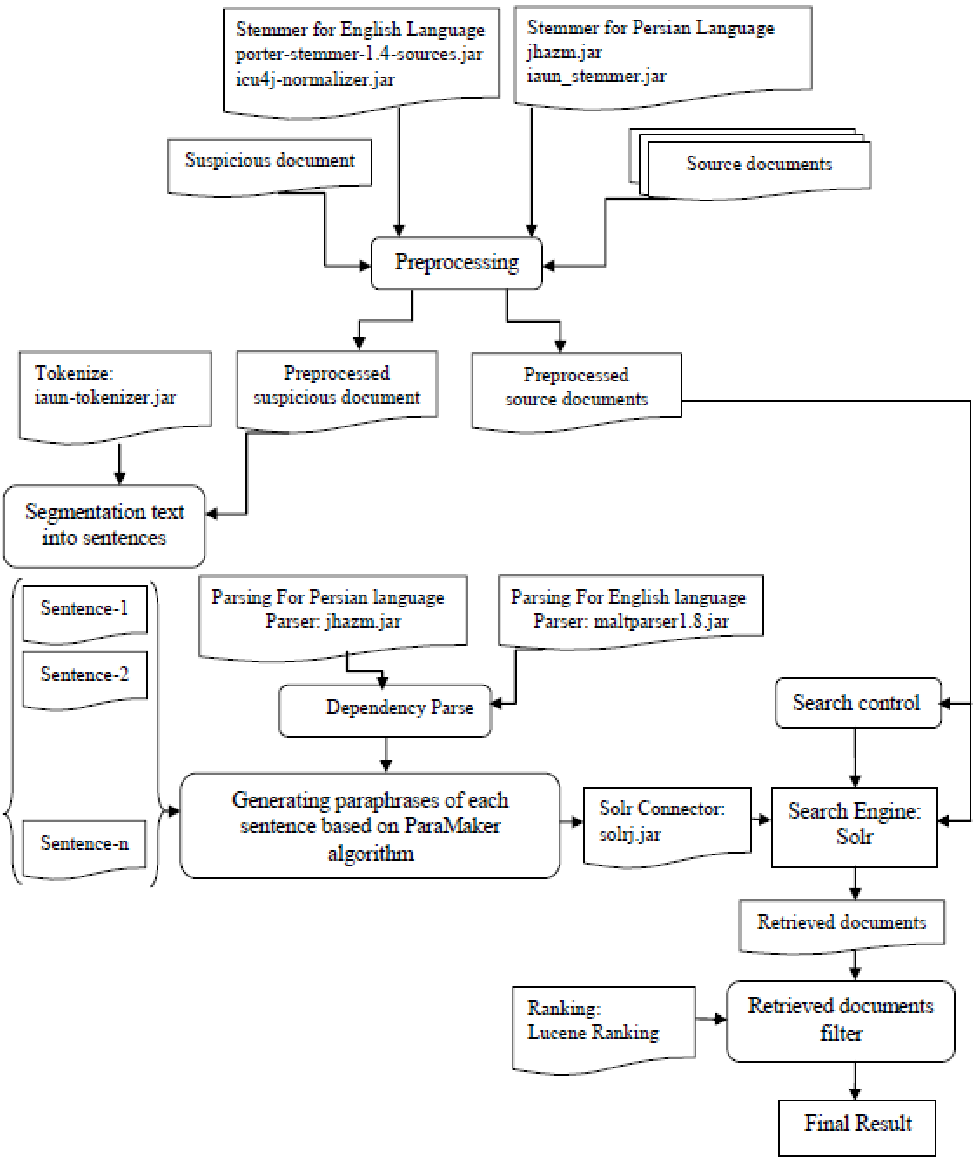
Stemmer for English Language
porter-stemmer-1.4-sources.jar
icu4j-normalizer.jar
Stemmer for Persian Language
jhazm.jar
iaun_stemmer.jar
Suspicious document
Source documents
Preprocessing
Tokenize:
iaun-tokenizer.jar
Preprocessed suspicious document
Preprocessed source documents
Segmentation text into sentences
Sentence-1
Sentence-2
Sentence-n
Parsing For Persian language
Parser: jhazm.jar
Parsing For English language
Parser: maltparser1.8.jar
Dependency Parse
Search control
Generating paraphrases of each sentence based on ParaMaker algorithm
Solr Connector:
solrj.jar
Search Engine:
Solr
Retrieved documents
Ranking:
Lucene Ranking
Retrieved documents filter
Final Result

Figure 8. ParaMaker evaluation environment.

Table 8. Design of experiments

Experiment-1	Evaluation of preprocessing.
Experiment-2	Evaluation of paraphrasing types.
Experiment-3	Evaluating number of retrievals for each query.
Experiment-4	Comparison of ParaMaker against PAN2014's best detectors.
Experiment-5	Measuring evaluation criteria for ParaMaker in Persian language in comparison with exact search.

6.1. Experiments for English and Persian Languages

6.1.1. Experiment-1: Evaluation of preprocessing

The experiment shown in figure 9 shows the effects of preprocessing on the parameters Precision and Recall. These parameters are measured with/without preprocessing, and the results obtained are finally compared. Preprocessing includes removing the stop words, replacing synonyms, and stemming. Compared to the time when there is no preprocessing, Recall parameter increased to 16% after removing the stop words, with synonym replacement to 5% and with stemming to 7%. On the other hand, Precision parameter decreases to 3%, 12%, and 8%, respectively.

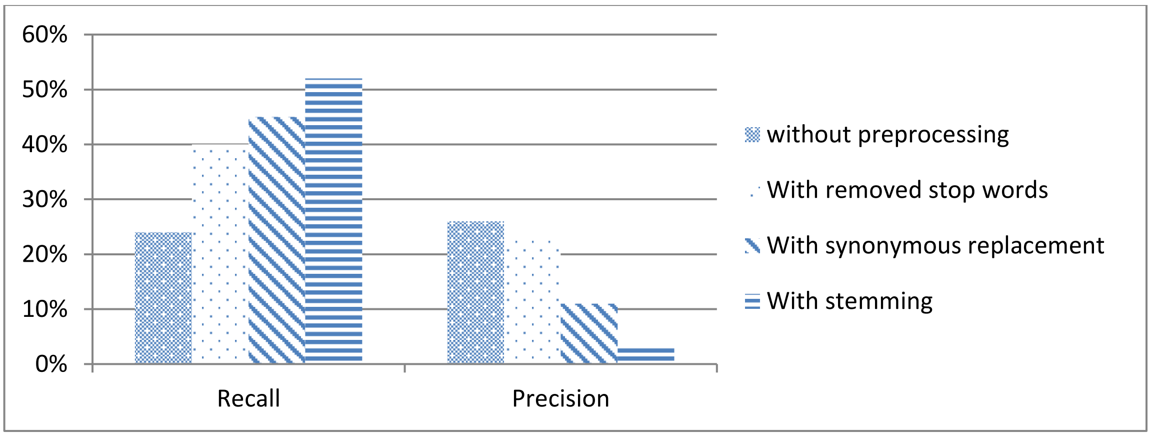

Figure 9. Evaluations of preprocessing

6.1.2. Experiment-2: Evaluating various types of paraphrasing

This experiment aims at evaluating the recognition algorithm. According to figure 10, compared to the near copy, the Recall parameter increased to 12% for light paraphrasing and 32% for heavy paraphrasing. Having a more number of retrieved documents in the collection, D_{ret} increases the Recall parameter due to having more number of common documents in both the source and retrieved documents. Increasing Recall is analogous to decreasing the Precision criterion. However, D_{ret} is in the denominator part of the Precision equation, and as a result, increasing the number of retrieved documents affects the Precision parameter inversely. In order to solve this problem, the retrieved documents must be mostly from the source documents. It can be realized that when downloads are constrained, higher ranked documents are retrieved.

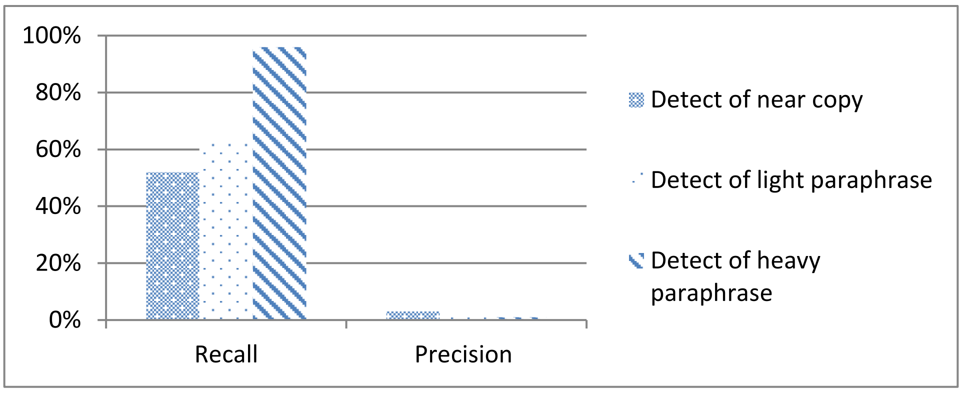

Figure 10. Evaluation of various paraphrasing types.

6.1.3. Experiment-3: Evaluating number of downloads for each query

As shown in figure 11, selecting the top three documents from the results returned by the search engine instead of considering the whole results improves the Precision parameter about 11% and decreases the Recall parameter about 5%. If a document is selected with the highest rank, Precision is increased significantly to 18%, while Recall decreases lightly about 6% that is negligible.

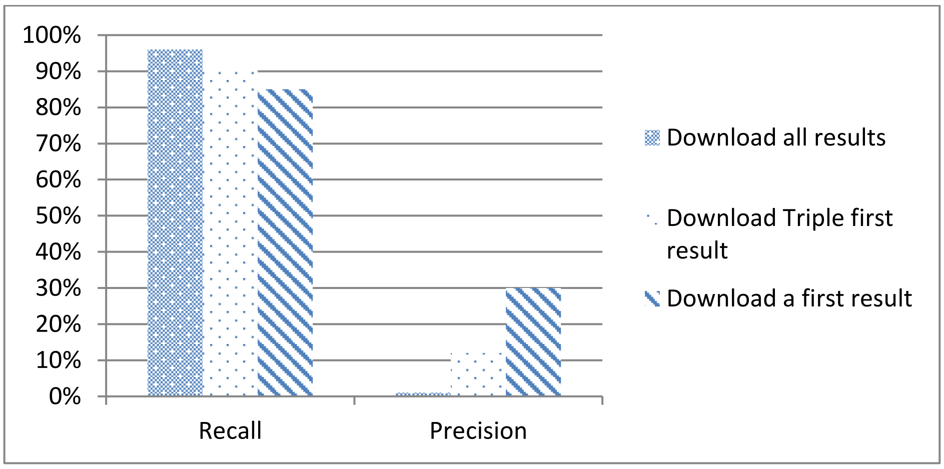

Figure 11: Evaluation of number of downloads for each query.

6.1.4. Experiment-4: Comparison of ParaMaker against PAN2014's best detectors

In this experiment, the performance of the six plagiarism detectors participated in the PAN2014 competition is studied against the proposed system. Comparison of the results in table 9 shows that the Recall parameter has improved 34% compared to Prakash [11] that had the highest Recall previously.

Table 9. Comparison of ParaMaker against PAN2014's best detectors.

Ref.	Software Team/	Submission Year	Downloaded Sources		Workload		Run time
			Rec.	Prec.	Queries	Retrieval	
	Proposed Method	2016	0.85	0.30	149.2	123.6	**20:49:12**
[11]	Prakash	2014	0.51	0.38	60.0	38.8	**19:47:45**
[8]	Kong	2014	0.48	0.08	83.5	207.1	**24:03:31**
[13]	Williams	2014	0.48	0.57	117.1	14.4	**39:44:11**
[14]	Zubarev	2014	0.45	0.54	37.0	18.6	**40:42:18**
[25]	Suchomel	2014	0.40	0.08	19.5	237.3	**45:42:06**
[6]	Elizalde	2014	0.39	0.40	54.5	33.2	**04:02:00**

6.1.5. Experiment-5: Measuring evaluation criteria for ParaMaker in Persian language in comparison with exact search

The same experiments similar to the experiments for the English language were repeated for the Persian language. The results obtained showed that ParaMaker increased the Recall parameter in the Persian language as well. Moreover, in comparison with increasing the Recall parameter, decreasing Precision can be waived. According to the previous experiments, the evaluation criteria are measured for exact searching by a search engine. After preprocessing, the results obtained are compared with the exact search, searching light paraphrasing of sentences, and finally, heavy paraphrasing of sentences. The whole evaluations are performed on three Persian datasets including TMC, IranDoc, and Prozhe.com. The evaluation results showed that the Recall parameter for the TMC dataset is improved around 45% in figure 12, 42% in figure 13 on IranDoc Dataset, and 61% in figure 14 on Prozhe.com dataset. Reducing Precision is due to the increase in the number of retrieved documents.

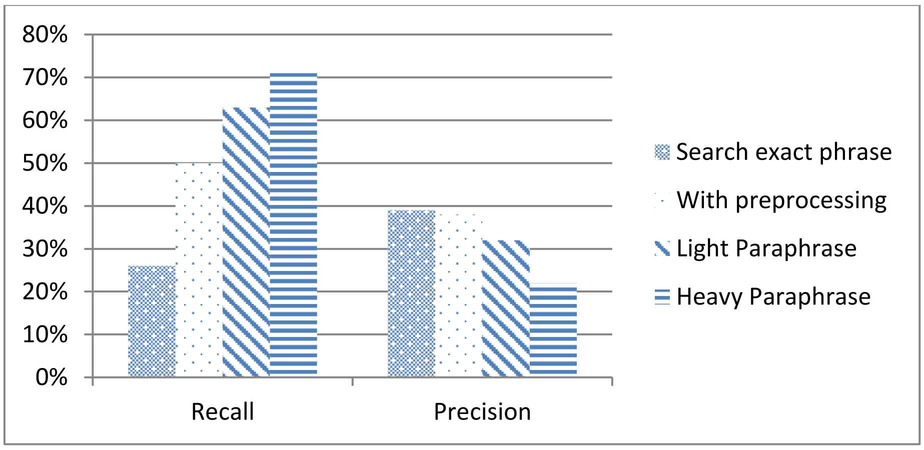

Figure 12. Evaluating ParaMaker in Persian language on TMC dataset.

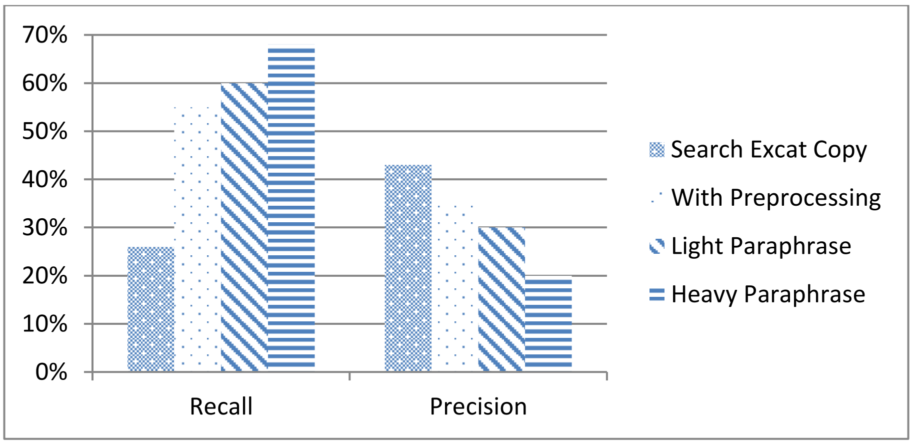

Figure 13.Evaluating ParaMaker in Persian language on IranDoc dataset.

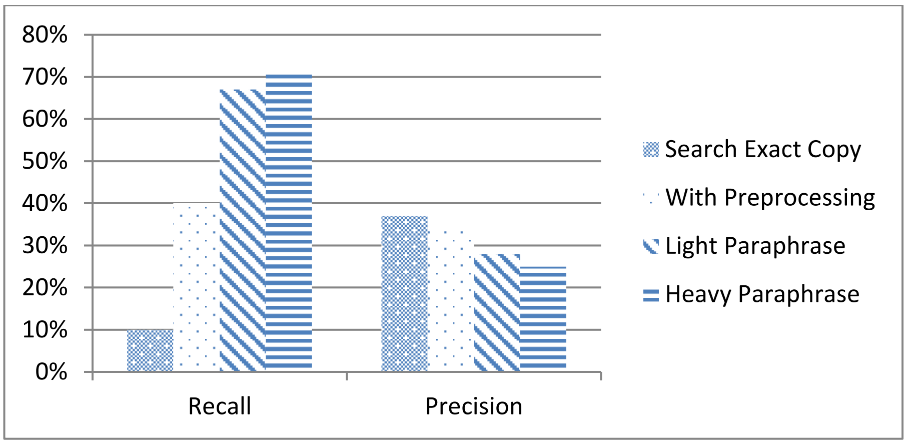

Figure 14. Evaluating ParaMaker in Persian language on Prozhe.com dataset.

6.2. Discussion

The preprocessing operation has a positive impact on improving the performance of detection algorithms. Whatever the number of generated paraphrases is greater, the number of queries increases. Increasing the number of queries results in increasing 1) the number of retrieved documents, 2) possibility of retrieved resources, and 3) Recall parameter. If the selected document was of the highest rank, Precision increased greatly; however, the Recall parameter reduced a negligible amount. Using the natural language processing techniques and producing correct paraphrase for each sentence in a suspicious document, more efficient queries were generated in the query construction phase. Having retrieved the best related documents for each sentence, the plagiarism detection algorithm would be able to improve the Recall parameter along with maintaining Precision and Speed of plagiarism detection.

7. Conclusions and future works

The plagiarism detection methods should look at the available resources on the Internet. Resource retrieval inspects suspicious documents and produces queries, which are subsequently submitted to search engines. A common issue in the plagiarism detection methods that use the resource retrieval techniques is having a low Recall parameter. Several methods try to improve this parameter, which lead to a sharp Precision and a Speed reduction.

This research work aimed at presenting a plagiarism detection approach that made paraphrases of sentences based on human behaviors in order to improve the Recall parameter, while maintained Precision and Speed. The ParaMaker framework was presented along with several algorithms to mimic the human behavior in making the paraphrases of a sentence. The paraphrased sentences were submitted to a search engine to retrieve the suspected documents. The comparison of the proposed system with six methods in PAN2014 showed an improvement of 34%, while the proposed method maintained the average of Precision and Speed.

As recommendations for future works: 1) Text summarization is mostly used in the idea of plagiarism; it is suggested that our idea get extended in the idea of paraphrasing. 2) Moreover, in the query generation phase, paraphrases of a sentence can be combined with the key phrases to generate more precise queries. 3) It is also suggested that in order to generate more precise queries, the paraphrased sentences must be considered for query generation. 4) Control search and/or filtering documents may also return more accurate results and reduce the number of retrieved resources instead of retrieving the highest rank documents.

8. References

[1] Maurer, H. A., Kappe, F. & Zaka, B. (2006). Plagiarism-A Survey. J. UCS, vol. 12, no. 8, pp. 1050-1084.

[2] Potthast, M., Hagen, M. A., Beyer, M., Tippmann, M., Busse, P., Rosso, & Stein, B. (2014). Overview of the 6th international competition on plagiarism detection. In CLEF Conference on Multilingual and Multimodal Information Access Evaluation, pp. 845-876.

[3] Alzahrani S. M., et al. (2012). Understanding Plagiarism linguistic patterns, textual features, and detection methods. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, vol. 42, pp. 133-149, 2012.

[4] Ceska Z., et al. (2008). Multilingual plagiarism detection. Artificial Intelligence: Methodology, Systems, and Applications, pp. 83-92, 2008.

[5] Ho C., et al. (2012). Extracting lexical and phrasal paraphrases: a review of the literature. Artificial Intelligence Review, pp. 1-44.

[6] Elizalde, V. (2014). Using Noun Phrases and tf-idf for Plagiarized Document Retrieval. Notebook for PAN at CLEF2014.

[7] Barker, K. & Cornacchia, N. (2000). Using Noun Phrase Heads to Extract Document Key phrases. In Hamilton, H.J. (ed.) Advances in Artificial Intelligence, 13th Biennial Conference of the Canadian Society for Computational Studies of Intelligence, AI 2000, Montréal, Quebec, Canada, May 14-17, 2000, Proceedings. Lecture Notes in Computer Science, vol. 1822, pp. 52-40.

[8] 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—Notebook for PAN at CLEF 2014.

[9] Robertson, S., et al. (2004). Simple BM25 extension to multiple weighted fields. In Proc. of the CIKM'04, 2004.

[10] Gaston H. Gonnet, Ricardo A. Baeza-yates & Snider, T. (1992). New Indices for Text: Pat Trees and Pat Arrays. Information Retrieval Data Structures & Algorithms, Prentice Hall, pp. 66-82.

[11] Prakash, A. & Saha, S. (2014). Experiments on Document Chunking and Query Formation for Plagiarism Source Retrieval—Notebook for PAN at CLEF2014.

[12] Toutanova, K. & Manning, C. D. (2000). Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics, vol. 13, pp. 63-70.

[13] Williams, K. H., Chen, & Giles, C. (2014). Supervised Ranking for Plagiarism Source Retrieval. Notebook for PAN at CLEF2014.

[14] Zubarev, D. & Sochenkov, I. (2014). Using Sentence Similarity Measure for Plagiarism Source Retrieval. Notebook for PAN at CLEF 2014.

[15] Chong, M., Specia, L. & Mitkov, R. (2011). Using Natural Language Processing for Automatic Detection of Plagiarism, In: Proceedings of the 4th International Plagiarism Conference, Newcastle-upon-Tyne, UK.

[16] Ceska, Z. & Fox, C. (2011). The Influence of Text Pre-processing on Plagiarism Detection. International Conference on Recent Advances in Natural Language Processing 2009. Association for Computational Linguistics, pp. 55-59.

[17] Willett, P. (2006). The Porter stemming algorithm: then and now. Program, vol. 40, pp. 219-223.

[18] Clough, P. (2000). Plagiarism in natural and programming languages: an overview of current tools and technologies, Department of Computer Science, University of Sheffield.

[19] Kübler, S., McDonald, R. & Nivre, J. (2009). Dependency parsing. Synthesis Lectures on Human Language Technologies, vol. 1, pp. 1-127.

[20] Nivre, J. (2005). Dependency grammar and dependency parsing, MSI report, vol. 5133, pp. 1-32.

[21] Potthast, M., Hagen, M., Völske, M. & Stein, B. (2013). Exploratory Search Missions for TRECTopics. In: Wilson, M.L., Russell-Rose, T., Larsen, B., Hansen, P., Norling, K. (eds.) 3rd European Workshop on Human-Computer Interaction and Information Retrieval (EuroHCIR2013). pp. 11–14. CEUR-WS.org.

[22] Rakian, S., Safi-Esfahani, F. & Rastegari, H. (2015). A Persian Fuzzy Plagiarism Detection Approach. Journal of Information Systems and Telecommunication, vol. 3, no. 3, pp. 182-190.

[23] Potthast, M., Hagen, M., Gollub, T., Tippmann, M., Kiesel, J., Rosso, P., Stamatatos, E. & Stein, B. (2013). Overview of the 5th international competition on plagiarism detection. In CLEF Conference on Multilingual and Multimodal Information Access Evaluation, pp. 301-331.

[24] Turtle, H., Hegde, Y. & Rowe, S. (2012). Yet another comparison of lucene and indri performance. In SIGIR2012 Workshop on Open Source Information Retrieval, pp. 64-67.

[25] Suchomel, Š. & Brandejs, M. (2014). Heterogeneous Queries for Synoptic and Phrasal Search. Notebook for PAN at CLEF2014.

نشریه هوش مصنوعی و داده کاوی

تشخیص سرقت علمی-ادبی بیرونی بر اساس رفتارهای انسانی در بازگویی جملات در زبان‌های انگلیسی و فارسی

فرامرز صافی‌اصفهانی *و شیما راکیان

دانشکده کامپیوتر، واحد نجف‌آباد، دانشگاه آزاد اسلامی، نجف‌آباد، ایران.

ارسال ۲۰۱۶/۰۷/۲۱؛ بازنگری ۲۰۱۷/۰۵/۳۰؛ پذیرش ۲۰۱۷/۱۱/۰۴

چکیده:

با ظهور اینترنت و سـادگی دسـترسی به منبع دیجیتالی، سرقت علمی-ادبی به یک مشکل جدی تبدیل شده است. بکارگیری موتورهای جستجو یکی از تکنیک‌های تشـخیص سرقت علمی-ادبی می‌باشد که در آن الگوهای سرقت به پرسش‌های موتور جستجو تبدیل می‌شود. ایجاد پرسش‌های مناسب قلب این تکنیک می‌باشـد. مشـکل روش‌های موجود دقیق و صـحت پایین پرســش‌های جســت‌وجو و همچنین سـرعت پایین اســتخراج نتایج مربوط به آنها می‌باشد. این تحقیق به ارائه یک چارچوب بنام ParaMaker می‌پردازد که شبیه رفتار انسان در بازگویی جملات به ساخت پرسش‌های جستجو پرداخته و با ارسـال آنها به موتور جســتجو موارد تقلب را پیدا می‌کند. در ارزیابی چارچوب پیشنهادی در زبان انگلیسی از دیتاست استاندارد PAN2014 استفاده شـده‌اسـت که با حفظ دقت و صـحت باعث بهبود ۳۴ درصدی پارامتر Recall می‌گردد. همچنین در زبان فارسی در مقایسه با یک موتور جستجو که بر اساس کپی دقیق کار می‌کند با حفظ دقت و صحت پارامتر Recall ۴۲ درصد بهبود داشته‌است.

کلمات کلیدی: تشخیص سرقت علمی-ادبی، تشخیص سرقت علمی-ادبی بیرونی، بازیابی منابع، بازگویش جملات.