Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

A machine learning approach to query generation in plagiarism source retrieval[*]

Lei-lei KONG[†1,2], Zhi-mao LU[1], Hao-liang QI[†‡2,3], Zhong-yuan HAN[2]
([1]*College of Information and Communication Engineering, Harbin Engineering University, Harbin 150001, China*)
([2]*School of Computer Science and Technology, Heilongjiang Institute of Technology, Harbin 150050, China*)
([3]*State Key Laboratory of Digital Publishing Technology, Beijing 100871, China*)
[†]E-mail: kongleilei1979@gmail.com; haoliang.qi@gmail.com
Received June 17, 2016; Revision accepted Dec. 12, 2016; Crosschecked Nov. 1, 2017

Abstract: Plagiarism source retrieval is the core task of plagiarism detection. It has become the standard for plagiarism detection to use the queries extracted from suspicious documents to retrieve the plagiarism sources. Generating queries from a suspicious document is one of the most important steps in plagiarism source retrieval. Heuristic-based query generation methods are widely used in the current research. Each heuristic-based method has its own advantages, and no one statistically outperforms the others on all suspicious document segments when generating queries for source retrieval. Further improvements on heuristic methods for source retrieval rely mainly on the experience of experts. This leads to difficulties in putting forward new heuristic methods that can overcome the shortcomings of the existing ones. This paper paves the way for a new statistical machine learning approach to select the best queries from the candidates. The statistical machine learning approach to query generation for source retrieval is formulated as a ranking framework. Specifically, it aims to achieve the optimal source retrieval performance for each suspicious document segment. The proposed method exploits learning to rank to generate queries from the candidates. To our knowledge, our work is the first research to apply machine learning methods to resolve the problem of query generation for source retrieval. To solve the essential problem of an absence of training data for learning to rank, the building of training samples for source retrieval is also conducted. We rigorously evaluate various aspects of the proposed method on the publicly available PAN source retrieval corpus. With respect to the established baselines, the experimental results show that applying our proposed query generation method based on machine learning yields statistically significant improvements over baselines in source retrieval effectiveness.

Key words: Plagiarism detection; Source retrieval; Query generation; Machine learning; Learning to rank
https://doi.org/10.1631/FITEE.1601344 **CLC number:** TP391.3

1 Introduction

In recent years, plagiarism has become easier as there are more web resources and powerful search engines. The problem of plagiarism has been aggravating because of the digital resources available on the World Wide Web (Alzahrani *et al.*, 2012). The increasingly serious problem of plagiarism has motivated and accelerated the development of plagiarism detection technology. Plagiarism and its automatic detection have gained extensive attention from academia to industry: an increasing number of studies have been performed in the field and many plagiarism detection software programs have been developed (Potthast *et al.*, 2012a). In particular, a considerable attention in this field is focused on the plagiarism detection algorithm evaluations organized by the Cross-Language Evaluation Forum (CLEF), which has accelerated the development of plagiarism detection algorithms and their related works. Plagiarism detection, with author identification and author profiling, becomes known as PAN (International

‡ Corresponding author
* Project supported by the National Social Science Foundation of China (No. 14CTQ032) and the National Natural Science Foundation of China (No. 61370170)
ORCID: Lei-lei KONG, http://orcid.org/0000-0002-4636-3507

Evaluation Competition on Uncovering Plagiarism, Authorship, and Social Software Misuse) in CLEF (http://pan. webis.de).

Plagiarism source retrieval is one of the core tasks of plagiarism detection. PAN@CLEF features a track, annotates the corpora of source retrieval, and introduces the task of source retrieval; i.e., given a suspicious document and a plagiarism source document collection, the task of source retrieval is to retrieve the plagiarized sources for the suspicious document (Potthast *et al.*, 2013a). In this case, the suspicious document contains plagiarized passages, the source document collection is the set of plagiarized source documents, and the acquisition of plagiarism sources is generally referred to as source retrieval in plagiarism detection.

In early plagiarism detection research, the source retrieval task on relatively small corpora consisting of only hundreds to a few thousands of documents was carried out (Prakash and Saha, 2014). Obviously, this would also be workable by comparing each suspicious document to its potential source documents in corpora of these sizes. However, there is a deep gap between a small collection and the real environment where humans reuse or plagiarize text from the web. Currently, sources for plagiarism come from the whole network, and plagiarists use a search engine to search for appropriate sources on a given topic. The traditional detailed comparison approaches are no longer viable given such large-scale document collections. Researchers now consider the whole web as a possible plagiarism source. Thus, the current trend is for plagiarism detection to generate queries from suspicious documents and exploit a search engine to retrieve the sources of plagiarized text (Potthast *et al.*, 2013a). Against this background, query generation is regarded as the most important step for the source retrieval algorithm, since the decisions made at this stage directly affect the overall performance (Potthast *et al.*, 2013a; 2014): the better the queries, the more the chances of retrieving source documents matching the suspicious document.

Considerable research on query generation for source retrieval has been carried out in recent years. To summarize, the current methods for query generation depend mainly on heuristic methods. For example, the following are extracted to generate queries: document-level term frequency (TF) ranked terms or paragraph-level TF-ranked terms (Prakash and Saha, 2014), terms with a high TF–inverse document frequency (TF–IDF) (Kong *et al.*, 2012; Elizalde, 2013; Suchomel *et al.*, 2015), name entities (Elizalde, 2013), terms with rarest frequency at the document level (Haggag and El-Beltagy, 2013), or only nouns, adjectives, and verbs (Jayapal, 2012; Williams *et al.*, 2013; 2014a; Zubarev and Sochenkov, 2014).

Heuristic-based methods have achieved a good performance; e.g., the Williams method (Williams *et al.*, 2013) achieved 0.47 on F-score (the highest F-score of PAN@CLEF 2013 (Potthast *et al.*, 2013a)). We find that each heuristic-based method has its own advantages and no one statistically outperforms the others among the classical query generation methods (Section 3.1). Our objective is to find a new method that is a statistical improvement over all the existing methods. Further improvement based on heuristic methods for source retrieval may rely on only the experience of experts, such as modifying the parameters of the original methods or attempting to combine and develop new heuristic methods. As we know, this undertaking could prove to be laborious. In this study, we pave the way for a new statistical machine learning method that can generate the best queries from the candidates by classical heuristic-based methods.

Certainly, statistical machine learning methods have achieved a great success in various fields. However, we find that little research has focused on machine learning methods directed toward the problem of query generation for source retrieval. From our analysis, resolving the query generation problem via statistical machine learning methods within a typical source retrieval environment is difficult, mainly because there is no training data. Given a suspicious document and a large collection of source documents, there is no direct indication which queries for the suspicious document may be the best for source retrieval, or which queries would be better than other queries. Facing the absence of training data, only heuristic methods can be used to extract the queries. In this study, we address the problem of query generation for source retrieval by means of machine learning. Our objectives focus on the following two questions:

1. How can machine learning based methods be applied effectively to the query generation problem?

2. How should training datasets be constructed for the existing source retrieval corpus?

For the first problem, a query generation method for source retrieval based on machine learning is introduced to analyze which queries are the best for source retrieval when given different candidate queries within the same suspicious document segment. This leads to a problem of learning with preference examples like: for suspicious document segment s_k, on performance metrics $\boldsymbol{m}$ of source retrieval, should query $Q_{(\mathrm{a})}$ be ranked higher than query $Q_{(\mathrm{b})}$? We will formalize the issue of query generation as a problem of learning to rank and choose the most superior queries when given the different candidate queries.

For the second problem, certainly if we know the set of queries that are actually relevant to the plagiarism source documents, we could use them as training examples to learn the query generation models. Unfortunately, we do not have any information besides the suspicious document and its plagiarism sources. However, if we have many different queries on a suspicious document segment (i.e., exploiting the existing heuristic query generation methods to produce these queries), we can use them to retrieve plagiarism sources. In terms of the performance metrics of source retrieval, these different queries can be compared with the others. By exploring these queries with labels of retrieval performance, the training samples can be built. In Section 3.3, we introduce a detailed generation process for data training.

Using the publicly available PAN@CLEF 2013 training and testing collections (http://pan.webis.de/clef13/pan13-web/plagiarism-detection.html), we rigorously evaluated various aspects of our query generation method. The experimental results show that the proposed machine learning method can successfully resolve the problem of query generation for source retrieval in plagiarism detection.

2 Related work

Our approach relates to the following three research areas: plagiarism detection, source retrieval, and classical works on query generation methods. In this section, we briefly describe the related works.

2.1 Generic retrieval process to detect plagiarism

Fig. 1 shows a generic retrieval process in plagiarism detection when given a suspicious document d_{susp} and a (very large) potential plagiarism source document collection D (Potthast *et al.*, 2013a).

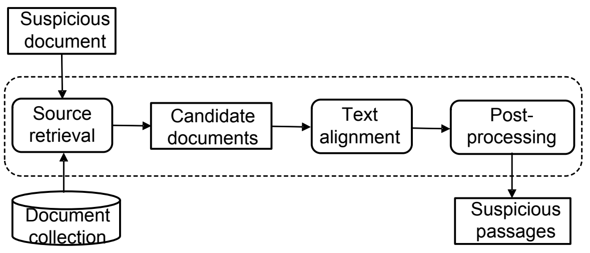

Fig. 1 Generic retrieval process to detect plagiarism (Potthast *et al.*, 2013a)

Fig. 1 was introduced by Potthast *et al.* (2013a). It summarizes the typical process that is implemented in most plagiarism detectors. The process is divided into three basic steps: First, source retrieval (which is called 'candidate retrieval' by PAN (Potthast *et al.*, 2012a)) identifies a small set of candidate documents $D_{\mathrm{src}} \subseteq D$ that are likely sources for plagiarism regarding d_{susp}. Second, by text alignment, where each candidate document $d_{\mathrm{src}} \in D_{\mathrm{src}}$ is compared to d_{susp}, all passages of the texts that are highly similar are extracted. Third, by post-processing the extracted passage pairs are filtered to determine the final plagiarized text passages. This study revolves around the source retrieval task.

2.2 Source retrieval

For source retrieval, if the document collection is small, then the algorithms for source retrieval, which resort to comparing all suspicious documents exhaustively to the available source documents, are feasible. In a realistic setting, however, the source collection could be very large and probably contains the entire web such that it becomes impractical to compare exhaustively. For the sake of realism, using web search engines to retrieve plagiarism sources in web-sized document collections may be one of the best solutions.

The task of source retrieval was proposed by PAN (Potthast *et al.*, 2012a). Potthast *et al.* (2012a; 2013a; 2014) depicted the basic process of source retrieval, which can be summarized in Fig. 2.

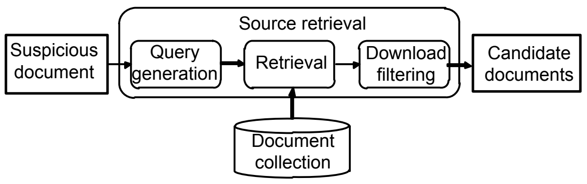

Fig. 2 Generic process of source retrieval

Source retrieval follows three main steps, as outlined in Fig. 2:

1. Query generation. Suspicious document d_{susp} is divided into small passages first. Then, given a passage in a suspicious document, words are extracted to construct the queries for this passage.

2. Retrieval. Given a set of queries, the algorithm for source retrieval schedules the submissions of the queries to the search engine and directs the download of search results.

3. Download filtering. Using a filtering algorithm, the search results are filtered to decide which downloaded documents will be further compared in detail with the suspicious document. This study is devoted to the issue of query generation based on the statistical machine learning method.

2.3 Heuristic methods for query generation of source retrieval

Queries for a suspicious document are the set of words or phrases extracted from the suspicious document (or suspicious document segments). These queries will be submitted to a search engine to retrieve the plagiarism sources. The rationale for query generation is to select only the queries that maximize the chance of retrieving source documents matching the suspicious document (Potthast *et al.*, 2013a; 2014). In the existing methods for query generation of source retrieval, heuristic methods remain the primary way to generate queries.

For example, Williams *et al.* (2013; 2014a; 2014b) removed the stop words and tagged each word using the Stanford tagger (Toutanova *et al.*, 2003), and then only the words tagged as nouns, verbs, and adjectives were retained as queries. Zubarev and Sochenkov (2014) followed a similar strategy: they formed queries by ignoring articles, pronouns, prepositions, and repeated words. Haggag and El-Beltagy employed combinations of rarest words (frequency at the document level) as queries (Haggag and El-Beltagy, 2013). Kong *et al.* (2012), Suchomel and Brandejs (2015), Suchomel *et al.* (2015), and Rafiei *et al.* (2015) used those rarest words with high TF–IDF values. Lee *et al.* (2013) exploited the Google books *n*-grams to determine the most unique eight-gram of each paragraph as the queries.

Recently, some researchers have tried to combine different heuristic-based query generation methods to perform source retrieval. For example, Elizalde (2013) applied three different strategies to generate the queries: the top 10 words scored by TF–IDF values, the named entities, and the noun phrases. Prakash and Saha (2014) used the top-5 document-level TF-ranked terms, the top-5 paragraph-level TF-ranked terms, and the nouns to form queries. However, no matter how complex the combinations of heuristic methods might be, they would still be interpreted as the heuristic-based methods, attempting to propose new heuristic methods or combining the existing heuristic methods. According to our knowledge, statistical machine learning methods have not been introduced in the existing investigations into query generation for source retrieval. In this study, we address the problems in exploiting a statistical machine learning method to generate the queries.

3 A machine learning method for query generation of source retrieval

In this section, we first analyze the performance of source retrieval when using different query generation methods. Then, a query generation framework and method based on statistical machine learning are proposed for source retrieval in plagiarism detection. Additionally, the construction of training samples is described in detail.

3.1 Which queries are the best ones for source retrieval?

We find that the different queries extracted by different methods for the same suspicious document segment can obtain different results in source retrieval; however, no method can always be superior to the other methods for all suspicious document segments.

To uncover various source retrieval performances when using different queries, we conduct an analysis on randomly selected data. As shown in Fig. 3, we take five classical query generation methods, which are TF (Prakash and Saha, 2014), TF–IDF (Kong *et al*., 2012; Elizalde, 2013; Suchomel and Brandejs, 2015; Suchomel *et al*., 2015), enhanced weirdness (EW) (Gillam, 2013), BM25 (Kong *et al*., 2012), and the Williams method (Williams *et al*., 2013; 2014a; 2014b), as examples. First, the suspicious document is divided into passages and we extract five groups of different queries for each segment by using the five different query generation methods above. Then we select 10 suspicious document segments at random. Using these queries to perform the retrieval, and taking F-score as the performance metric, we find that each one of the five query generation methods has opportunities to achieve the best F-score across the 10 suspicious document segments (Fig. 3).

Fig. 3 reveals that each heuristic-based method has its own advantages and no one outperforms the others statistically. In suspicious document segment 1, the query generated by the TF method achives the best F-score, in segments 2 and 8, the queries generated by the TF–IDF method obtain the highest F-score, while in segments 4, 7, and 9, the queries generated by the Williams method win the best F-score. In segment 4, the queries generated by the BM25 and the Williams methods reach the highest F-score at the same time.

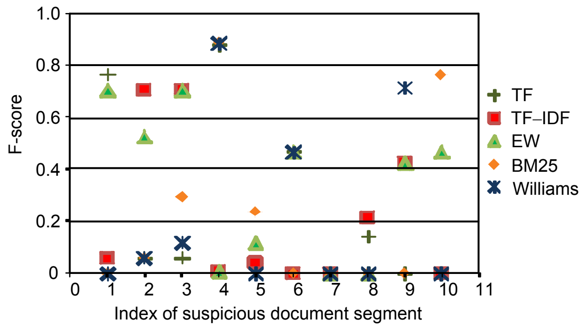

Fig. 3 Source retrieval results obtained by different queries extracted by different methods for the same suspicious document segment
TF: term frequency; TF–IDF: TF–inverse document frequency; EW: enhanced weirdness

This phenomenon occurs widely in source retrieval. Table 1 shows the statistics for the best F-score that each query extraction method achieves for 3500 suspicious document segments. The numbers for the best F-score contain duplicate statistics. In the same suspicious document segments, if the queries extracted by different methods are the same, or different queries retrieve the same plagiarism sources, or the numbers of plagiarism sources retrieved by different methods are the same, then the methods obtain the same F-score.

Table 1 Statistics of the best F-score

Method	Number of the best F-scores
TF	1260
TF–IDF	1391
EW	1285
BM25	1363
Williams	1445

TF: term frequency; TF–IDF: TF–inverse document frequency; EW: enhanced weirdness

Let us consider the examples in Fig. 3 again. In suspicious document segment 1, the query generated by the TF method obtains the best F-score, compared with the other queries. In suspicious document segment 2, the query generated by the TF–IDF method achieves the best F-score, and the query generated by EW is better than those generated by the BM25, TF, and Williams methods. Thus, TF–IDF and EW outperform the other three methods in suspicious document segment 2. When it is not possible to infer which method generates the best queries on an absolute scale, it is much more plausible to infer which ones are better than others in a suspicious document segment. Denoting the preferred ranking by $\gg$, we obtain the ranking information in suspicious document segment s_1 of the form:

$$
\begin{aligned}
&Q_{(s_1)}^{(\mathrm{TF})} \gg Q_{(s_1)}^{(\mathrm{EW})},\ Q_{(s_1)}^{(\mathrm{TF})} \gg Q_{(s_1)}^{(\mathrm{TFIDF})},\\
&Q_{(s_1)}^{(\mathrm{TF})} \gg Q_{(s_1)}^{(\mathrm{BM25})}, Q_{(s_1)}^{(\mathrm{TF})} \gg Q_{(s_1)}^{(\mathrm{Williams})},\\
&Q_{(s_1)}^{(\mathrm{EW})} \gg Q_{(s_1)}^{(\mathrm{TFIDF})},\ Q_{(s_1)}^{(\mathrm{EW})} \gg Q_{(s_1)}^{(\mathrm{BM25})},\\
&Q_{(s_1)}^{(\mathrm{EW})} \gg Q_{(s_1)}^{(\mathrm{Williams})},\ Q_{(s_1)}^{(\mathrm{TFIDF})} \gg Q_{(s_1)}^{(\mathrm{BM25})},\\
&Q_{(s_1)}^{(\mathrm{TFIDF})} \gg Q_{(s_1)}^{(\mathrm{Williams})},
\end{aligned}
$$

where $Q_{(s_1)}^{(i)}$ denotes the queries that are extracted by query generation method i in suspicious document segment s_1.

Now, our question is: which queries are the best ones for the suspicious document segment s_1 among all candidates when performing source retrieval? For suspicious document segment s_1, $Q_{(s_1)}^{(\mathrm{TF})}$ is the best choice because it obtains the highest F-score. For suspicious document segment 3, however, the queries extracted by EW and TF–IDF are all the best choices, since they achieve the same best F-score above 0.6. Yet, in this segment, if we choose the query extracted by the TF method, then F-score will be lower than 0.2. Thus, enormous gaps in source retrieval performance remains among the queries retrieved by different methods. Owing to the preference pairs, it is straightforward to formalize query generation as a ranking problem, and the objective of the algorithm is to achieve a rank for different queries in each suspicious document segment, and then the N best candidates in the ranking list will be chosen as the queries for the suspicious document segment.

3.2 Query generation framework

According to Section 3.1, the strategy for generating queries from candidates is summarized as follows:

Generating queries: For a set of candidate queries $\left\{Q_{(s_k)}^{(1)}, Q_{(s_k)}^{(2)}, \ldots, Q_{(s_k)}^{(n)}\right\}$ with a preference relationship and a suspicious document segment s_k containing all the candidates, the candidate queries are ranked to obtain a ranking list, and the top-N queries in the list are chosen as the queries for s_k.

To solve the above query generation problem, a framework for applying a machine learning method to generate queries for source retrieval is presented in Fig. 4. Given a collection of training data, a specific learning algorithm is employed to learn the ranking function. Candidate queries can be generated by some baseline systems. Then, in suspicious document segment s_k, the ranking function learned in the training phase is applied to return a corresponding query ranking list, and the top-m queries of the list will be selected as the queries of s_k.

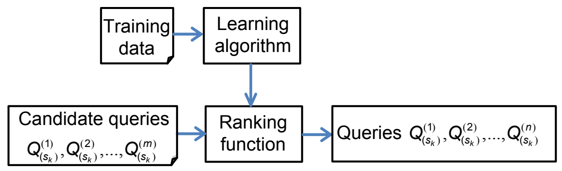
Training data
Learning algorithm
Candidate queries $Q_{(s_k)}^{(1)}, Q_{(s_k)}^{(2)}, \ldots, Q_{(s_k)}^{(m)}$
Ranking function
Queries $Q_{(s_k)}^{(1)}, Q_{(s_k)}^{(2)}, \ldots, Q_{(s_k)}^{(n)}$

Fig. 4 A framework for the proposed query generation based on machine learning for source retrieval in plagiarism detection

To achieve the framework above, we subscribe to the ranking view of information retrieval and view the query generation as a problem of learning to rank. We believe this view has certain inherent advantages. First, this framework mirrors the issues discussed in Section 3.1 very accurately: there are differences among different queries on the same suspicious document segments, and we are primarily concerned with how to rank these queries and choose the ones that might retrieve more plagiarism sources, and this is exactly what the model aims to quantify. Second, casting query generation as a learning-to-rank process allows us to leverage many features and methods developed in the machine learning domain.

Then the central issues focus on two aspects: how to generate the training datasets from the existing data for source retrieval (only suspicious documents and their plagiarism sources are available), and how to learn the ranking function based on the training datasets. We will describe each of these concerns in detail in Sections 3.3 and 3.4.

3.3 Creating training samples

To learn the ranking function, a training set T, which contains a number of queries with labels, is needed. Let s_k be the kth segment in suspicious document d_{susp}. The training samples $T_{(s_k)}$ on s_k is defined as

$$T_{(s_k)} = \left\{ t_{(s_k)}^{(i)} \mid t_{(s_k)}^{(i)} = \left(y_{(s_k)}^{(i)}, Q_{(s_k)}^{(i)} \right) \right\}, \tag{1}$$

where $Q_{(s_k)}^{(i)}$ is a candidate query that consists of terms extracted from suspicious document segment s_k, and $y_{(s_k)}^{(i)}$ is the ground truth labels of $Q_{(s_k)}^{(i)}$.

For source retrieval, the queries generated from a suspicious document segment will be submitted to a search engine. However, there is no direct way to show which queries are more plausible for source retrieval. However, for different queries, which are generated from the same suspicious document segment by different query-generation methods, we can evaluate their qualities according to the performance metrics of source retrieval. Then the values for the performance metrics can be regarded as the ground truth labels to indicate which queries are better than the others. In this way, we can obtain the queries with a preference relationship.

Let $Q^{(i)}_{(s_k)}$ be generated by function $\varphi^{(i)}$ as follows:

$$Q^{(i)}_{(s_k)} = \varphi^{(i)}(s_k), \tag{2}$$

where $\varphi^{(i)}$ can be any existing query-generation method (such as TF and TF–IDF) and s_k is the suspicious document segment.

We define ground truth label $y^{(i)}_{(s_k)}$ of a candidate query $Q^{(i)}_{(s_k)}$ as

$$\begin{cases} y^{(i)}_{(s_k)} = \mathrm{eval}\left(D^{(i)}_{(s_k)}\right), \\ D^{(i)}_{(s_k)} = \mathrm{sr}\left(Q^{(i)}_{(s_k)}\right), \end{cases} \tag{3}$$

where $y^{(i)}_{(s_k)}$ denotes the quality of the query, and eval(·) is a function of evaluating the source retrieval performance, which quantifies the quality of the queries. We choose the performance metrics of source retrieval (such as Recall, Precision, and F-score) to define eval(·). In this study, we use F-score as the function eval(·), which is described in Section 4.2. sr(·) is a retrieval function, which is defined by the search engine used to return the set of retrieved results $D^{(i)}_{(s_k)}$ by using $Q^{(i)}_{(s_k)}$ as the query.

In this study, we apply the retrieval function offered by ChatNoir described in Section 4.3.2 (Potthast *et al.*, 2012b). Training sample $T_{(s_k)}$ is generated in one suspicious document segment s_k, and all of the training datasets T can be constructed by using different suspicious document segments. Fig. 5 describes the process noted above.

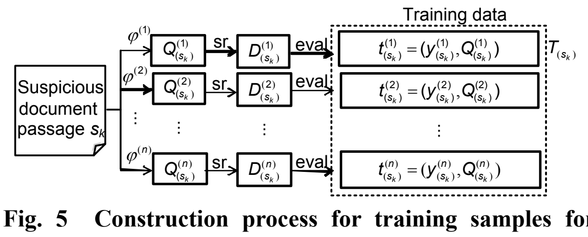
Training data
Suspicious document passage s_k
$\varphi^{(1)}$
$\varphi^{(2)}$
$\varphi^{(n)}$
$Q^{(1)}_{(s_k)}$
$Q^{(2)}_{(s_k)}$
$Q^{(n)}_{(s_k)}$
sr
$D^{(1)}_{(s_k)}$
$D^{(2)}_{(s_k)}$
$D^{(n)}_{(s_k)}$
eval
$t^{(1)}_{(s_k)} = (y^{(1)}_{(s_k)}, Q^{(1)}_{(s_k)})$
$t^{(2)}_{(s_k)} = (y^{(2)}_{(s_k)}, Q^{(2)}_{(s_k)})$
$t^{(n)}_{(s_k)} = (y^{(n)}_{(s_k)}, Q^{(n)}_{(s_k)})$
$T_{(s_k)}$

Fig. 5 Construction process for training samples for source retrieval in one suspicious document segment s_k

To see more clearly how training samples are built, consider the following example. In Fig. 3, the content of suspicious document segment s_2 is shown as follows:

In what follows, we give a detailed overview of Barack Obama's Family. We shed light on himself, his immediate and extended family, including maternal and paternal relations. Moreover, we give insights into the relations of Michelle Obama, Barack Obama's wife, as well as some distant relations of both. Barack Obama Barack Hussein Obama II is the 44th and current President of the United States. He is the first African American to hold the office. Obama was the junior United States Senator from Illinois from 2005 until he resigned following his election to the presidency.

First, we select five candidate query generation methods, TF, TF–IDF, EW, BM25, and the Williams methods, as functions $\varphi^{(i)}$ to obtain the following candidates:

$$\begin{cases} Q^{(1)}_{(s_2)} = \varphi^{(1)}(s_2) = \mathrm{TF}(s_2), \\ Q^{(2)}_{(s_2)} = \varphi^{(2)}(s_2) = \mathrm{TFIDF}(s_2), \\ Q^{(3)}_{(s_2)} = \varphi^{(3)}(s_2) = \mathrm{EW}(s_2), \\ Q^{(4)}_{(s_2)} = \varphi^{(4)}(s_2) = \mathrm{BM25}(s_2), \\ Q^{(5)}_{(s_2)} = \varphi^{(5)}(s_2) = \mathrm{Williams}(s_2). \end{cases}$$

With the above five methods, we obtain the following five candidate queries:

$$\begin{cases} Q^{(1)}_{(s_k)} = \text{(Obama Barack relations give family states president 44th American distant)}, \\ Q^{(2)}_{(s_k)} = \text{(Barack family paternal maternal president Hussein overview 44th African Obama)}, \\ Q^{(3)}_{(s_k)} = \text{(Barack family Obama paternal maternal give relations president Hussein overview)}, \\ Q^{(4)}_{(s_k)} = \text{(Obama relations Barack states detailed Michelle follows office united extended)}, \\ Q^{(5)}_{(s_k)} = \text{(Obama Barack relations give family American hold current light office)}. \end{cases}$$

Taking the source retrieval function which ChatNoir (Potthast *et al.*, 2012b) provids as retrieval function sr(·), with the evaluation metric F-score (Potthast *et al.*, 2013a) as evaluation function eval(·), we obtain $y^{(1)}_{(s_k)}$= 0.0588, $y^{(2)}_{(s_k)}$= 0.7058, $y^{(3)}_{(s_k)}$= 0.5209, $y^{(4)}_{(s_k)}$= 0.0588, and $y^{(5)}_{(s_k)}$= 0.0588. Combining each

$y_{(s_k)}^{(i)}$ with its corresponding $Q_{(s_k)}^{(i)}$, a group of training samples on s_2 can be produced.

Using the above training samples, seven reference relationships are also obtained on s_2, where $Q_{(s_2)}^{(i)} \gg Q_{(s_2)}^{(j)}$ means that query $Q_{(s_2)}^{(i)}$ is ranked ahead of query $Q_{(s_2)}^{(j)}$ according to their labels:

$$Q_{(s_2)}^{(2)} \gg Q_{(s_2)}^{(1)},\ Q_{(s_2)}^{(2)} \gg Q_{(s_2)}^{(3)},\ Q_{(s_2)}^{(2)} \gg Q_{(s_2)}^{(4)},$$
$$Q_{(s_2)}^{(2)} \gg Q_{(s_2)}^{(5)},\ Q_{(s_2)}^{(3)} \gg Q_{(s_2)}^{(1)},\ Q_{(s_2)}^{(3)} \gg Q_{(s_2)}^{(4)},$$
$$Q_{(s_2)}^{(3)} \gg Q_{(s_2)}^{(5)}.$$

hese preference relationships will then be given to a learning algorithm to learn a ranking function to obtain a ranking list of candidate queries and to select the best query for source retrieval.

3.4 Machine learning method for query generation

According to Section 3.3, we obtain the training samples with preference relationships, allowing us to formalize the query generation for source retrieval as a ranking task. In this subsection, the learning algorithm for ranking candidate queries is described. This method is built upon a ranking function, in which candidate queries are ranked and the queries for a suspicious document are generated by using the top-ranked queries. The features used by the ranking function are also introduced in detail.

3.4.1 Learning algorithm to rank candidate queries

Assume that $\boldsymbol{x}_{(s_k)}^{(i)}$ is the feature vector of candidate query $Q_{(s_k)}^{(i)}$ which will be defined in Section 3.4.2, and $y_{(s_k)}^{(i)}$ is the label of $Q_{(s_k)}^{(i)}$ evaluated by the evaluation metrics for source retrieval (Eq. (3)). The preference relationship $Q_{(s_k)}^{(i)} \gg Q_{(s_k)}^{(j)}$ between $Q_{(s_k)}^{(i)}$ and $Q_{(s_k)}^{(j)}$ described in Section 3.3 can be represented by using $\boldsymbol{x}_i$ and y_i as follows:

$$y_{(s_k)}^{(i)} > y_{(s_k)}^{(j)} \Rightarrow \boldsymbol{x}_{(s_k)}^{(i)} \gg \boldsymbol{x}_{(s_k)}^{(j)}. \tag{4}$$

Suppose that a set of ranking functions $f \in F$ exists and that each f can determine the preference relationships between samples:

$$\boldsymbol{x}_{(s_k)}^{(i)} \gg \boldsymbol{x}_{(s_k)}^{(j)} \Leftrightarrow f(\boldsymbol{x}_{(s_k)}^{(i)}) > f(\boldsymbol{x}_{(s_k)}^{(j)}). \tag{5}$$

By using f to rank the candidate queries on s_k, we can obtain a ranking list $r_{f(s_k)}$ on s_k. Then the goal of the learning algorithm is to learn a ranking function f from a family of ranking functions F to make ranking list $r_{f(s_k)}$ resemble its actual ordering $r_{(s_k)}^{*}$ most closely.

Furthermore, suppose that f is a linear function:

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = <\boldsymbol{w}, \boldsymbol{x}>, \tag{6}$$

where $\boldsymbol{w}$ is a weight vector adjusted by learning, $\boldsymbol{x}$ represents a feature vector of a candidate query, and $<\cdot,\cdot>$ stands for the inner product. Under the action of linear ranking function f, we have

$$\boldsymbol{x}_{(s_k)}^{(i)} \gg \boldsymbol{x}_{(s_k)}^{(j)} \Leftrightarrow \boldsymbol{w}\boldsymbol{x}_{(s_k)}^{(i)} > \boldsymbol{w}\boldsymbol{x}_{(s_k)}^{(j)}. \tag{7}$$

To select from F the best ranking function f that minimizes a given loss function with respect to the given ranked instances (Cao *et al.*, 2006), we need to find the weight vector so that the maximum number of the following inequalities is fulfilled on n suspicious document segments of training datasets T:

$$\begin{cases} \forall\left(Q_{(s_1)}^{(i)}, Q_{(s_1)}^{(j)}\right) \in r_{(s_1)}^{*}: \boldsymbol{w}\boldsymbol{x}_{(s_1)}^{(i)} > \boldsymbol{w}\boldsymbol{x}_{(s_1)}^{(j)}, \\ \forall\left(Q_{(s_2)}^{(i)}, Q_{(s_2)}^{(j)}\right) \in r_{(s_2)}^{*}: \boldsymbol{w}\boldsymbol{x}_{(s_2)}^{(i)} > \boldsymbol{w}\boldsymbol{x}_{(s_2)}^{(j)}, \\ \qquad\vdots \\ \forall\left(Q_{(s_n)}^{(i)}, Q_{(s_n)}^{(j)}\right) \in r_{(s_n)}^{*}: \boldsymbol{w}\boldsymbol{x}_{(s_n)}^{(i)} > \boldsymbol{w}\boldsymbol{x}_{(s_n)}^{(j)}. \end{cases} \tag{8}$$

The direct generalization of the above results is a non-deterministic polynomial-time (NP) hard problem (Höffgen *et al.*, 1995). By introducing (non-negative) slack variables $\xi_{i,j,k}$ ($\forall i, j, k$: $\xi_{i,j,k}$>0) (Joachims, 2002), we have

$$\begin{cases} \forall(Q_{(s_1)}^{(i)}, Q_{(s_1)}^{(j)}) \in r_{(s_1)}^{*}: \boldsymbol{w}\boldsymbol{x}_{(s_1)}^{(i)} > \boldsymbol{w}\boldsymbol{x}_{(s_1)}^{(j)} + 1 - \xi_{i,j,1}, \\ \forall(Q_{(s_2)}^{(i)}, Q_{(s_2)}^{(j)}) \in r_{(s_2)}^{*}: \boldsymbol{w}\boldsymbol{x}_{(s_2)}^{(i)} > \boldsymbol{w}\boldsymbol{x}_{(s_2)}^{(j)} + 1 - \xi_{i,j,2}, \\ \qquad\vdots \\ \forall(Q_{(s_n)}^{(i)}, Q_{(s_n)}^{(j)}) \in r_{(s_n)}^{*}: \boldsymbol{w}\boldsymbol{x}_{(s_n)}^{(i)} > \boldsymbol{w}\boldsymbol{x}_{(s_n)}^{(j)} + 1 - \xi_{i,j,n}. \end{cases} \tag{9}$$

Then, Eq. (8) can be rewritten as follows:

$$\begin{cases} \boldsymbol{w}\boldsymbol{x}_{(s_1)}^{(i)} - \boldsymbol{w}\boldsymbol{x}_{(s_1)}^{(j)} = \boldsymbol{w}\left(\boldsymbol{x}_{(s_1)}^{(i)} - \boldsymbol{x}_{(s_1)}^{(j)}\right) > 1 - \xi_{i,j,1}, \\ \boldsymbol{w}\boldsymbol{x}_{(s_2)}^{(i)} - \boldsymbol{w}\boldsymbol{x}_{(s_2)}^{(j)} = \boldsymbol{w}\left(\boldsymbol{x}_{(s_2)}^{(i)} - \boldsymbol{x}_{(s_2)}^{(j)}\right) > 1 - \xi_{i,j,2}, \\ \quad\vdots \\ \boldsymbol{w}\boldsymbol{x}_{(s_n)}^{(i)} - \boldsymbol{w}\boldsymbol{x}_{(s_n)}^{(j)} = \boldsymbol{w}\left(\boldsymbol{x}_{(s_n)}^{(i)} - \boldsymbol{x}_{(s_n)}^{(j)}\right) > 1 - \xi_{i,j,n}. \end{cases} \tag{10}$$

Thus, it becomes apparent that the ranking problem is equivalent to the problem of classification of pairwise vectors $\left(\boldsymbol{x}_{(s_k)}^{(i)}, \boldsymbol{x}_{(s_k)}^{(j)}\right)$, as proposed by Herbrich *et al.* (2000). Since $\boldsymbol{x}_{(s_k)}^{(i)}$ and $\boldsymbol{x}_{(s_k)}^{(j)}$ denote the *i*th and *j*th feature vectors in candidate queries $Q_{(s_k)}^{(i)}$ and $Q_{(s_k)}^{(j)}$, respectively, $y_{(s_k)}^{(i)}$ and $y_{(s_k)}^{(j)}$ denote the ranks of $\boldsymbol{x}_{(s_k)}^{(i)}$ and $\boldsymbol{x}_{(s_k)}^{(j)}$, respectively. Then we have

$$\left(\boldsymbol{x}_{(s_k)}^{(i)} - \boldsymbol{x}_{(s_k)}^{(j)}, z_{i,j,k}\right), \tag{11}$$

where

$$z_{i,j,k} = \begin{cases} +1, & y_{(s_k)}^{(i)} > y_{(s_k)}^{(j)}, \\ -1, & y_{(s_k)}^{(i)} < y_{(s_k)}^{(j)}. \end{cases}$$

This means that any vector $\boldsymbol{x}_{(s_k)}^{(i)} - \boldsymbol{x}_{(s_k)}^{(j)}$ is given a positive label $z=+1$ or a negative label $z=-1$ according to the ranks of $\boldsymbol{x}_{(s_k)}^{(i)}$ and $\boldsymbol{x}_{(s_k)}^{(j)}$. Following the method proposed by Cao *et al.* (2006), if having constructed training sample set T, we can create a new training dataset T' by using the above pairwise vectors:

$$T' = \left\{\left(\boldsymbol{x}_{(s_k)}^{(i)} - \boldsymbol{x}_{(s_k)}^{(j)}\right), z_{i,j,k}\right\}. \tag{12}$$

Taking T' as the classification data, a classification model can be used to learn weight vector $\boldsymbol{w}$.

Considering the sizes of the training datasets (only thousands of effective preferred pairs), we adopt the classic small sample learning algorithm, support vector machine (SVM) (Cortes and Vapnik, 1995), to maximize the margin of the positive and negative samples. Following the work of Joachims (2002), adding SVM regularization for margin maximization to the objective, constructing the SVM model is equivalent to solving the following quadratic optimization problem on pairwise vectors:

$$\begin{aligned} &\min_{\boldsymbol{w}} V(\boldsymbol{w}, \boldsymbol{\xi}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + c\sum \xi_{i,j,k} \\ &\text{subject to} \\ &z_{i,j,k}\left\langle \boldsymbol{w}, \boldsymbol{x}_{(s_k)}^{(i)} - \boldsymbol{x}_{(s_k)}^{(j)}\right\rangle \geq 1 - \xi_{i,j,k}, \\ &\forall i,j,k: \xi_{i,j,k} > 0, \end{aligned} \tag{13}$$

where c is a tunable parameter that allows a trading-off margin size against the training error and $\xi_{i,j,k}$ are the slack variables.

According to Hastie *et al.* (2001), when $\lambda=1/2c$, the optimization of problem (13) is equivalent to that in

$$\min_{\boldsymbol{w}} \sum\left[1 - z_{i,j,k}\left\langle \boldsymbol{w}, \boldsymbol{x}_{(s_k)}^{(i)} - \boldsymbol{x}_{(s_k)}^{(j)}\right\rangle\right]_+ + \frac{1}{2}\|\boldsymbol{w}\|^2, \tag{14}$$

where subscript '+' indicates the positive part. The first element is the hinge loss defined in our query pairs, which means that if $\boldsymbol{w}\boldsymbol{x}_{(s_k)}^{(i)}$ is larger than $\boldsymbol{w}\boldsymbol{x}_{(s_k)}^{(j)}$ by a margin of one, there is no loss; otherwise, the loss will be $\xi_{i,j,k}$. The second element is the regularizer.

Supposing that $\boldsymbol{w}^*$ are the optimal weights learned by SVM, final ranking function f can be achieved by using $\boldsymbol{w}^*$:

$$f_{\boldsymbol{w}^*}(\boldsymbol{x}) = \left\langle \boldsymbol{w}^*, \boldsymbol{x}\right\rangle. \tag{15}$$

3.4.2 Features

We rely on the classic features of information retrieval to define our features to learn ranking function *f*.

Supposing $\boldsymbol{x}_{(s_k)}^{(i)} \in \mathbb{R}^N$ is a feature vector of candidate query $Q_{(s_k)}^{(i)}$, which is defined as follows:

$$\boldsymbol{x}_{(s_k)}^{(i)} = \Phi\left(Q_{(s_k)}^{(i)}, s_k, d_j\right), \tag{16}$$

where $\Phi\left(Q_{(s_k)}^{(i)}, s_k, d_j\right)$ is a mapping onto the features that describe the match among query $Q_{(s_k)}^{(i)}$, the *k*th suspicious document segment s_k from which $Q_{(s_k)}^{(i)}$ is extracted, and the jth suspicious document d_j to which s_k belongs.

In our method, only the statistics extracted from query-document pairs (such as TF and IDF) and their combinations are chosen as features. In information retrieval, these features are usually used to describe the relevance between the query and the document.

From an information retrieval perspective, viewing $Q_{(s_k)}^{(i)}$ as the query, and s_k or d_j, from which $Q_{(s_k)}^{(i)}$ is extracted, as the document, we defined 12 features to represent a feature vector $\boldsymbol{x}_{(s_k)}^{(i)}$ according to Nallapati (2004) and Cao *et al.* (2006). We took a logarithm on some feature values to reduce the effects of large numbers. For all the features described in Table 2, k_m denotes one term of $Q_{(s_k)}^{(i)}$, tf(k, d) represents the raw count of term k in document d, C represents a document collection, idf(k) is the inverse document frequency of term k, $|d|$ is the number of terms in d, avdl and avsl are the average term numbers of d_j and s_k, respectively, and k_1 and b are two constants of Okapi BM25 (Robertson, 1997). Note that we use only the statistical features to define the feature vectors of candidate queries. The other types of features, such as the syntactic or semantic features, can also be defined to describe characteristics of different candidate queries to improve the source retrieval performance. We believe that more features can depict the candidate queries more accurately and flexibly, and such investigation will be our future work.

Table 2 Features used for ranking

Rank	Feature
1	$\sum_{k_m \in Q_{(s_k)}^{(i)}} \log(\mathrm{tf}(k_m, s_k)+1)$
2	$\sum_{k_m \in Q_{(s_k)}^{(i)}} \log(\mathrm{tf}(k_m, d_j)+1)$
3	$\sum_{k_m \in Q_{(s_k)}^{(i)}} \log\left(\frac{\lvert C\rvert}{\mathrm{tf}(k_m, C)}+1\right)$
4	$\sum_{k_m \in Q_{(s_k)}^{(i)}} \log(\mathrm{idf}(k_m))$
5	$\sum_{k_m \in Q_{(s_k)}^{(i)}} \log\left(\frac{\mathrm{tf}(k_m, s_k)}{\lvert s_k\rvert}+1\right)$
6	$\sum_{k_m \in Q_{(s_k)}^{(i)}} \log\left(\frac{\mathrm{tf}(k_m, d_j)}{\lvert d_j\rvert}+1\right)$
7	$\sum_{k_m \in Q_{(s_k)}^{(i)}} \log\left(\frac{\mathrm{tf}(k_m, s_k)}{\lvert s_k\rvert}\cdot\frac{\lvert C\rvert}{\mathrm{tf}(k_m, C)}+1\right)$
8	$\sum_{k_m \in Q_{(s_k)}^{(i)}} \log\left(\frac{\mathrm{tf}(k_m, d_j)}{\lvert d_j\rvert}\cdot\frac{\lvert C\rvert}{\mathrm{tf}(k_m, C)}+1\right)$
9	$\sum_{k_m \in Q_{(s_k)}^{(i)}} \log\left(\frac{\mathrm{tf}(k_m, s_k)}{\lvert s_k\rvert}\cdot\mathrm{idf}(k_m)+1\right)$
10	$\sum_{k_m \in Q_{(s_k)}^{(i)}} \log\left(\frac{\mathrm{tf}(k_m, d_j)}{\lvert d_j\rvert}\cdot\mathrm{idf}(k_m)+1\right)$
11	$\sum_{k_m \in Q_{(s_k)}^{(i)}} \frac{\mathrm{tf}(k_m, s_k)\cdot\mathrm{idf}(k_m)\cdot(k_1+1)}{\mathrm{tf}(k_m, s_k)+k_1\cdot\left(1-b+b\cdot\frac{\mathrm{LEN}(s_k)}{\mathrm{avsl}}\right)}$
12	$\sum_{k_m \in Q_{(s_k)}^{(i)}} \frac{\mathrm{tf}(k_\mathrm{m}, d_j)\cdot\mathrm{idf}(k_m)\cdot(k_1+1)}{\mathrm{tf}(k_m, d_j)+k_1\cdot\left(1-b+b\cdot\frac{\mathrm{LEN}(d_j)}{\mathrm{avdl}}\right)}$

4 Experiments

4.1 Benchmark datasets

The evaluation corpus employed for source retrieval is based on the Webis Text Reuse Corpus 2012 (Webis-TRC-2012) (Potthast *et al.*, 2013b). A detailed description of Webis-TRC-2012 can be found in Potthast *et al.* (2013b). We evaluated the proposed method on the training data and testing data from the source retrieval task in PAN 2013. These datasets are all parts of Webis-TRC-2012 (Potthast *et al.*, 2013a; 2014), including 40 and 58 suspicious documents, respectively. The statistics information on the data corpus is shown in Table 3.

Table 3 Statistics for experimental corpora

Corpus	Number of suspicious documents	avg(#) of words of d_{susp}	avg(#) of plagiarism segments of d_{src}	avg(#) of plagiarism segments of d_{susp}	Jaccard (d_{susp}, d_{src})
PAN 2013 training corpus	40	5495	10.22	18.43	0.1978
PAN 2013 test corpus 2	58	4588	9.32	28.46	0.1822

avg(#) of words of d_{susp} represents the average word number of suspicious documents, avg(#) of plagiarism segments of d_{src} or d_{susp} is the average number of segments in d_{src} or d_{susp}, and Jaccard (d_{susp}, d_{src}) is the average degree of similarity between the whole suspicious document d_{susp} and its plagiarism source document d_{src} calculated by the Jaccard coefficient

Following PAN, we used ClueWeb09 as the plagiarism source document datasets, which consists of 1 040 809 705 web pages in 10 languages (Potthast *et al.*, 2013a). It has been adopted widely to evaluate retrieval models and search engines within the annual Text REtrieval Conference (TREC) (http://trec.nist.gov).

4.2 Performance measures

Given a suspicious document d_{susp} consisting of passages that have been reused from a set of source documents D_{src}, PAN measures the retrieval performance of a source retrieval algorithm in terms of precision, recall, and F-score of the set of retrieved documents D_{ret}.

For evaluation, considering the near-duplicate web documents, which are almost but not exactly the same as some source documents, PAN presents the following method to detect the near-duplicate document (Potthast *et al.*, 2013a; 2014): For any $d_{\text{ret}}\in D_{\text{ret}}$, the evaluation algorithm employs a near-duplicate detector to judge whether it is a true positive detection, e.g., whether there is a $d_{\text{src}}\in D_{\text{src}}$ of d_{susp} that is a near duplicate of d_{ret}. d_{ret} is a true positive detection for a given pair of d_{src} and d_{ret}, if (1) $d_{\text{ret}}=d_{\text{src}}$, or (2) the Jaccard similarity of the word n-grams in d_{ret} and d_{src} is above 0.8 for n=3, above 0.5 for n=5, and above 0 for n=8, or (3) the passages in d_{susp} known to be reused from d_{src} are contained in d_{ret}. More details can be found in Hagen *et al.* (2015). In this study, if d_i is a true positive detection of d_j, it is marked as positiveDet(d_i, d_j).

Considering the near duplicates, let D_{dup} denote the set of all near duplicates of a given set of source documents D_{src} of d_{susp}, and D'_{ret} denote the subset of D_{src} that has at least one corresponding true positive detection in D_{ret} (Potthast *et al.*, 2013a). Then we define

$$D_{\text{dup}}=\{d_{\text{dup}}|\exists \text{d}_{\text{src}}\in D_{\text{src}}: \text{positiveDet}(d_{\text{dup}}, d_{\text{src}})\},$$
$$D'_{\text{ret}}=\{d_{\text{src}}|d_{\text{src}}\in D_{\text{src}} \text{ and } \exists d_{\text{ret}}\in D_{\text{ret}}: \text{positiveDet}(d_{\text{ret}}, d_{\text{src}})\}.$$

Based on these sets, PAN defines the Precision, Recall, and F-score as

$$\text{Precision}=\frac{|D_{\text{ret}}\cap D_{\text{dup}}|}{|D_{\text{ret}}|}, \tag{17}$$

$$\text{Recall}=\frac{|D'_{\text{ret}}\cap D_{\text{src}}|}{|D'_{\text{ret}}|}, \tag{18}$$

$$\text{F-score}=\frac{2\cdot\text{Recall}\cdot\text{Precision}}{\text{Recall}+\text{Precision}}, \tag{19}$$

where Precision depicts the passages of the documents retrieved that are actual sources of plagiarism to the suspicious document. It measures how good the algorithm is at correctly identifying true sources of plagiarism. Recall describes the passages of the documents that are actually sources of plagiarism of the suspicious document that are retrieved successfully. Note that a high Precision can be achieved only by retrieving the documents with such a high confidence that they may be plagiarized sources, though a high Precision can be achieved at the expense of a low Recall. Similarly, a high Recall can be achieved by retrieving a large set of documents with low Precision. F-score is commonly used to represent the tradeoff between Precision and Recall.

In this study, following PAN, performance metric F-score is used as function eval(·) (Eq. (3)) to evaluate label $y^{(i)}_{(s_k)}$ of training example $t^{(i)}_{(s_k)}$.

To measure the cost effectiveness of a source retrieval algorithm in retrieving D_{ret}, PAN counts the numbers of queries and downloads until the first true positive detection is made. Four cost-effectiveness evaluation metrics are denoted as total workload of queries, total workload of downloads, workload to the first detection of queries, and workload to the first downloads. From a source-retrieval cost point of view, these performance metrics should be as low as possible.

4.3 Experimental setup

4.3.1 Baselines

We selected the Williams method (Williams *et al.*, 2014b) as our strong baseline to evaluate the performance of the proposed query generation method based on machine learning. The Williams method is one of the most successful plagiarism detection systems from PAN. It achieved the best F-score in PAN 2013 and PAN 2014 (Potthast *et al.*, 2013a; 2014).

The hypothesis of the Williams method is that verb, noun, and adjective words are more likely to be the keywords of a segment text; thus, Williams *et al.*

(2013) restricted their queries to only verbs, nouns, and adjectives. The Williams method first partitions the suspicious document into segments that are made up of five sentences. For each segment, only the nouns, verbs, and adjectives extracted by the Stanford Part-of-Speech (POS) Tagger (Toutanova *et al.*, 2003) are retained. Then, the first three queries are generated via combining every non-overlapping sequence of 10 words. We denoted this baseline as Williams.

Meanwhile, we chose TF–IDF, which is a very common strategy for query generation in source retrieval (Potthast *et al.*, 2013a; 2014), as another baseline. We denoted it as TF–IDF. TF–IDF calculates the product of TF and IDF to evaluate how important a term is. For comparison, we also used five sentences as one segment. The difference between Williams and TF–IDF methods lies in the query generation process. TF–IDF uses the words ranked by their TF–IDF values, instead of nouns, verbs, and adjectives as queries. For each segment, we removed the stop words, and the terms were ranked by their TF–IDF values. The document frequency of a term is obtained from an external corpus (we used the *Wall Street Journal* corpus in our experiments) (Potthast *et al.*, 2013a; 2014). Then, followed by the Williams method, the top-30 words were selected to generate three queries, each of which contains 10 words.

Although TF–IDF is a simple query generation method, when we replaced the queries generated by the Williams method with those generated by the TF–IDF method, we found that the latter performed better than the former in terms of source retrieval. Considering its representativeness and better performance, we leveraged TF–IDF as our another baseline.

4.3.2 Process for source retrieval

Except for query generation, the other processes for source retrieval in our method and the baselines all followed the Williams method (Williams *et al.*, 2013; 2014b).

The queries generated by the above two baselines were used as candidate queries. To rank the candidate queries, we trained a ranking function on a training corpus using the ranking SVM (http://www.cs.cornell.edu/People/tj/svm_light/svm_rank.html) with all the other parameters set to their default values except parameter c (which denotes the tradeoff between the training error and margin). By using the learned ranking function, we ranked all six candidate queries in each segment. Similar to the baselines, we selected the first three queries from the candidates as the queries for this segment.

ChatNoir (Potthast *et al.*, 2012b) (offered and suggested by PAN) was used as the search engine to retrieve the plagiarism sources in our evaluation. ChatNoir is one of the publicly available search engines that index the ClueWeb09 corpus (Potthast *et al.*, 2012b). It implements the classic BM25F model and incorporates pagerank and spamrank scores. The queries selected by our method were submitted to ChatNoir according to their extracted schedules. For each query, not only the baselines, but also our method, remained in the top-three search results, and the search results of each query were merged into the final result set of each suspicious document.

Following the Williams method (Williams *et al.*, 2013; 2014b), we trained a classifier using linear discriminant analysis (LDA) (http://scikit-learn.org) on the training corpus to filter the search results. The details of the classification and features were described in Williams *et al.* (2013; 2014b).

4.3.3 Training of parameters

Using the standard settings for PAN, the PAN 2013 plagiarism source retrieval training corpus was used as the training data to tune the parameters, and the PAN 2013 plagiarism source retrieval test corpus 2 was used as the test data for evaluation. The results reported in Section 4.4 are the ones run on the testing data whose parameters were tuned on the training data. The experimental results were achieved on the test corpus (Tables 4 and 5). As an additional benefit, the results can be directly compared with the results from the PAN plagiarism source retrieval track. To better evaluate the performance of the proposed method, we performed another experiment: We took the PAN 2013 test corpus 2 as our training data and the PAN 2013 training corpus as the test data to evaluate the proposed method (Tables 6 and 7). Our method is denoted as query generation based on machine learning (QGML).

When training the ranking function, we chose the main evaluation metric F-score as the evaluation function eval$(\cdot)$ to label the performance of the source retrieval of each query $Q_{(s_k)}^{(i)}$.

Table 4 Experimental results on the PAN 2013 testing corpus 2 with no download filtering

Method	Downloaded source			Total workload		Workload to first detection	
	F-score	Precision	Recall	Query	Download	Query	Download
Williams	0.1332 (+45.95%)	0.0780 (+58.33%)	0.6428 (−9.80%)	150.24	255.71	14.74	39.72
TF–IDF	0.1491 (+30.38%)	0.0918 (+34.53%)	0.5692 (+11.82%)	148.94	185.95	25.38	32.05
QGML	0.1944*	0.1235*	0.6365	150.24	165.45*	14.52	26.24*

TF–IDF: term frequency–inverse document frequency; QGML: query generation based on machine learning. Figures in brackets indicate the relative improvements in QGML over the baselines; * indicates that the experimental results are statistically significant at the level of $p<0.05$ using a one-sided paired *t*-test

Table 5 Experimental results on the PAN 2013 testing corpus 2 with download filtering

Method	Downloaded source			Total workload		Workload to first detection	
	F-score	Precision	Recall	Query	Download	Query	Download
Williams	0.4798 (+7.12%)	0.4834 (+27.58%)	**0.5338** (−5.47%)	150.24	31.79	15.02	6.02
TF–IDF	0.4823 (+6.55%)	0.5407 (+14.06%)	0.4946 (+2.02%)	**148.94**	**18.57**	24.41	5.29
QGML	**0.5139***	**0.6167***	0.5046	150.24	22.47	**14.52**	**4.84**

TF–IDF: term frequency–inverse document frequency; QGML: query generation based on machine learning. Bold values represent the best results per category; figures in brackets indicate the relative improvements in QGML over the baselines; * indicates that the experimental results are statistically significant at the level of $p<0.05$ using a one-sided paired *t*-test

Table 6 Experimental results on the PAN 2013 training corpus with no download filtering

Method	Downloaded sources			Total workload		Workload to first detection	
	F-score	Precision	Recall	Query	Download	Query	Download
Williams	0.1485 (+28.21%)	0.0892 (+36.88%)	0.6878 (−3.77%)	170.98	267.33	15.40	45.53
TF–IDF	0.1762 (+8.06%)	0.1095 (+11.51%)	0.6425 (+3.02%)	172.38	197.64	22.95	43.15
QGML	0.1904*	0.1221*	0.6619	173.95	213.20	19.13	36.50*

TF–IDF: term frequency–inverse document frequency; QGML: query generation based on machine learning. Figures in brackets indicate the relative improvements in QGML over the baselines; * indicates that the experimental results are statistically significant at the level of $p<0.05$ using a one-sided paired *t*-test

Table 7 Experimental results on the PAN 2013 training corpus with download filtering

Method	Downloaded sources			Total workload		Workload to first detection	
	F-score	Precision	Recall	Query	Download	Query	Download
Williams	0.5199 (+8.39%)	0.5390 (+23.78%)	**0.5664** (−5.91%)	170.98	37.85	**15.40**	7.13
TF–IDF	0.5304 (+6.24%)	0.5912 (+12.86%)	0.5292 (+0.70%)	**172.38**	**24.33**	22.95	**5.03**
QGML	**0.5635***	**0.6672***	0.5329	173.95	28.08	19.13	5.65

TF–IDF: term frequency–inverse document frequency; QGML: query generation based on machine learning. Bold values represent the best results per category; figures in brackets indicate the relative improvements in QGML over the baselines; * indicates that the experimental results are statistically significant at the level of $p<0.05$ using a one-sided paired *t*-test

4.4 Experimental results and analysis

The experimental results are shown for two cases: when no download filtering was used and when download filtering was used. The former addresses finding out whether the improvements in the queries can boost the source retrieval performance when no additional algorithm is acting on the search results, while the latter focuses on examining whether the overall source retrieval performance can be boosted using the proposed query generation methods.

In the following experimental results, we report the per-category F-scores for the baselines and our proposed method. Precision, Recall, total workload of queries and downloads, and workload to the first detection of queries and downloads are also listed as reference.

4.4.1 With no download filtering

The goal of source retrieval is to use a search engine to retrieve the plagiarism sources by exploiting the constructed queries from the suspicious document. The objective of the following two experiments is to prove whether the improvement in queries can boost the source retrieval performance directly. As discussed in Section 2.2, there are three core processes in source retrieval: query generation, retrieval, and download filtering. In our experiments, since we chose ChatNoir as our search engine, the retrieval ranking model is decided by the selected search engine. No download filtering being used means that we did not leverage any additional information, except for the search results, to evaluate the source retrieval performance. In other words, once the set of queries is submitted to the search engine, the search result documents can then be returned according to the ChatNoir retrieval model.

Since we do nothing about the retrieved results, the source retrieval performance depends entirely on the quality of the queries: better queries will retrieve more plagiarism sources. Tables 4 and 6 show the performance when no download filtering is used for the proposed methods and the baselines. The comparisons indicate that the proposed method leads to an improvement in performance over the baseline. As can be seen from Tables 4 and 6, in the two datasets, the proposed method achieves significantly better results than the baselines, while the total workload of downloads and the workload to first detection of queries and downloads are roughly equal to their counterparts in the baselines, and some evaluation metrics are even lower than their counterparts in the baselines. Taking the experimental results with the PAN 2013 testing corpus 2 as an example, the relative improvements in QGML over the Williams and TF–IDF methods on F-score are 45.95% and 30.38%, respectively, while the corresponding Precision improvements in QGML over the Williams and TF–IDF methods are 58.33% and 34.53%, respectively. However, the relative decline in QGML over the Williams method is less than 10% on Recall, while for TF–IDF, QGML achieves a relative improvement of 11.82% on Recall. These figures may suggest that QGML is more effective than the baselines.

No download filtering being used means that we do not leverage any additional information except for the search results to evaluate the source retrieval performance.

In the two experiments above, we extracted on average 299.18 queries from 40 suspicious documents, and 343.36 queries from 58 suspicious documents to construct the training samples. The numbers of effective preferred pairs are only 2403 and 3016. In such a scaled training set, the experimental results show that our method could learn an effective ranking function to select better queries from the candidates to improve the source retrieval performance.

Note that the performance measures F-score and Precision in the experimental results above are lower than those reported by the PAN competition. This is mainly due to no download filtering being performed in these two experiments. Our retrieved document collection $|D_{\mathrm{ret}}|$ contains all the top-n retrieval results, resulting in a larger value of $|D_{\mathrm{ret}}|$ and a lower value on the evaluation metric Precision. The lower Precision is responsible for the lower F-score. The Precision would be improved when a download filtering algorithm is used on the original search results.

4.4.2 With download filtering

Download filtering performs a more accurate matching to further check if the retrieved results are plagiarism sources. At this point, we can obtain more information, such as a snippet of the search results or the data returned by the search engine (i.e., the PageRank score of the result and the number of words in the result), to filter the results for improving the Precision. Adopting the same download filtering method as in the Williams method, Tables 5 and 7 show the performances when download filtering is used to test whether the overall source retrieval performance can be boosted using the proposed query generation methods.

As shown in Tables 5 and 7, download filtering improves the Precision greatly by filtering out many noise documents. As a result, F-score is improved vastly. Under the circumstances of using the same download filtering algorithm, QGML outperforms the baselines on F-score at statistically significant levels ($p<0.05$). In Table 5, F-score for QGML is better than that for the Williams method by 7.12%, and better than that for TF–IDF by 6.55%, while in Table 7, F-score for QGML is better than that for the Williams method by 8.39%, and better than that for TF–IDF by

6.24%. From these results, we can see that the proposed methods with download filtering can always achieve a better performance than the baselines. This also demonstrates that incorporating our query generation method can bring benefits in boosting the overall source retrieval performance.

5 Discussions

There are several aspects to consider further for the method we propose in this study.

5.1 Integrating new query generation methods and new features

Our method makes a good use of the differences in source retrieval performances among different candidate queries. For each suspicious document segment, different candidates can be attained by different query generation methods, resulting in different source retrieval performances. The objective of our method is to select the queries that tend to achieve a better source retrieval performance using the learning-to-rank method.

More importantly, our method provides a general framework for query generation for source retrieval for plagiarism detection. With regard to new query generation methods, our method allows them to participate in the generation of candidate query predictions. In our method, the candidate queries can be extracted by many candidate query generation methods, allowing new query generation methods to be simply integrated into our framework. With regard to new feature sets, our method can integrate them to identify better queries. New query generation methods could provide new, and maybe better, candidate queries, and new features could enable the proposed method to identify better queries, all of which can make performance improvements possible.

5.2 Query terms for source retrieval versus keyphrases in a document

Note that the keywords in a document are not always suitable for performing the queries for source retrieval, because the objectives of the two tasks are different. The objective of query generation for source retrieval is to retrieve the plagiarism sources using these terms as queries, while the purpose of keyphrase generation in a document is to find the terms or phrases that can precisely and compactly represent the content of a document. Plagiarism is always obfuscated by paraphrasing and summarizing to change most of its appearance (Alzahrani *et al.*, 2012; Barrón-Cedeño *et al.*, 2013), and the terms in source documents are often replaced in the suspicious document. Thus, the keyphrases that express the subject of a document segment are not necessarily the best queries for source retrieval. For example, the Williams method (Williams *et al.*, 2013; 2014b) (one of our baselines) and the Jayapal method (Jayapal, 2012) (which achieved a precision of 0.6582 in the PAN@CLEF 2012 source retrieval track) apply only the POS information to generate queries. Haggag and El-Beltagy (2013) used the rarest word (whose frequency is at the document level) as the queries and achieved an acceptable F-score of 0.44 in the PAN@CLEF 2013 source retrieval track.

5.3 Local performance improvement versus global performance improvement

Optimization for local source retrieval performance for each segment via a machine learning method is the main strategy of our approach. The final plagiarism sources of the suspicious document are obtained by combining the results retrieved by the queries of each segment. For a suspicious document, taking F-score as an example, the local optimized F-score for each suspicious document segment does not necessarily mean the global optimal F-score for the whole suspicious document. However, not knowing in which segments the plagiarism is present, searching plagiarism sources for each segment is a necessary step in plagiarism detection. Under these circumstances, the problem is transformed into obtaining the best F-score in each segment. In our method, the process described above is performed using learning to rank to generate the best queries for each segment to achieve the optimal F-score.

6 Conclusions

Query generation is the core task of source retrieval for plagiarism detection. Due to the lack of training samples, heuristic-based methods are used mainly in current investigations. Compared with statistical machine learning methods, the basis for the heuristic methods is absent. However, statistical machine learning methods have not been exploited in query generation for source retrieval. To address these issues, we investigated the machine learning approach to query generation for source retrieval based on a new formulation of this issue into the ranking framework. Distinctively, we presented a method to build the training samples for source retrieval. Furthermore, a machine learning approach based on pairwise learning to rank was used to deal with the query generation issue inherited from source retrieval.

The validity of the method was proved by a variety of experiments on the PAN datasets. The experimental results showed the effectiveness of the proposed method, which statistically outperforms the state-of-the-art methods in terms of F-score with similar or even lower retrieval costs.

References

Alzahrani, S.M., Salim, N., Abraham, A., 2012. Understanding plagiarism linguistic patterns, textual features, and detection methods. *IEEE Trans. Syst. Man Cybern. C*, **42**(2):133-149. https://doi.org/10.1109/TSMCC.2011.2134847

Barrón-Cedeño, A., Vila, M., Martí, M.A., *et al.*, 2013. Plagiarism meets paraphrasing: insights for the next generation in automatic plagiarism detection. *Comput. Ling.*, **39**(4):917-947. https://doi.org/10.1162/COLI_a_00153

Cao, Y., Xu, J., Liu, T.Y., *et al.*, 2006. Adapting ranking SVM to document retrieval. Proc. 29th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, p.186-193. https://doi.org/10.1145/1148170.1148205

Cortes, C., Vapnik, V., 1995. Support-vector networks. *Mach. Learn.*, **20**(3):273-297. https://doi.org/10.1023/A:1022627411411

Elizalde, V., 2013. Using statistic and semantic analysis to detect plagiarism—notebook for PAN at CLEF 2013. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Gillam, L., 2013. Guess again and see if they line up: surrey's runs at plagiarism detection—notebook for PAN at CLEF 2013. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Hagen, M., Potthast, M., Stein, B., 2015. Source retrieval for plagiarism detection from large web corpora: recent approaches. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Haggag, O., El-Beltagy, S., 2013. Plagiarism candidate retrieval using selective query formulation and discriminative query scoring—notebook for PAN at CLEF 2013. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Hastie, T., Tibshirani, R., Friedman, J., 2001. The Elements of Statistical Learning: Data Mining, Inference and Prediction. CRC Press, Boca Raton.

Herbrich, R., Graepel, T., Obermayer, K., 2000. Large margin rank boundaries for ordinal regression. *In*: Smola, A.J., Bartlett, P., Schölkopf, B., *et al.* (Eds.), Advances in Large Margin Classifiers. MIT Press, Cambridge, p.115-132.

Höffgen, K.U., Simon, H.U., Vanhorn, K.S., 1995. Robust trainability of single neurons. *J. Comput. Syst. Sci.*, **50**(1):114-125. https://doi.org/10.1006/jcss.1995.1011

Jayapal, A., 2012. Similarity overlap metric and greedy string tiling at PAN 2012: plagiarism detection—notebook for PAN at CLEF 2012. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Joachims, T., 2002. Optimizing search engines using clickthrough data. Proc. 8th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, p.133-142. https://doi.org/10.1145/775047.775067

Kong, L.L., Qi, H.L., Wang, S., *et al.*, 2012. Approaches for candidate document retrieval and detailed comparison of plagiarism detection—notebook for PAN at CLEF 2012. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Lee, T., Chae, J., Park, K., *et al.*, 2013. CopyCaptor: plagiarized source retrieval system using global word frequency and local feedback—notebook for PAN at CLEF 2013. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Nallapati, R., 2004. Discriminative models for information retrieval. Proc. 27th Annual ACM SIGIR Int. Conf. on Research and Development in Information Retrieval, p.64-71. https://doi.org/10.1145/1008992.1009006

Potthast, M., Gollub, T., Hagen, M., *et al.*, 2012a. Overview of the 4th International Competition on Plagiarism Detection. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Potthast, M., Hagen, M., Stein, B., *et al.*, 2012b. ChatNoir: a search engine for the ClueWeb09 corpus. Proc. 35th Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, p.1004. https://doi.org/10.1145/2348283.2348429

Potthast, M., Hagen, M., Gollub, T., *et al.*, 2013a. Overview of the 5th International Competition on Plagiarism Detection. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Potthast, M., Hagen, M., Völske, M., *et al.*, 2013b. Crowdsourcing interaction logs to understand text reuse from the web. Proc. 51st ACM Annual Meeting of the Association of Computational Linguistics, p.1212-1221.

Potthast, M., Hagen, M., Beyer, A., *et al.*, 2014. Overview of the 6th International Competition on Plagiarism

Detection. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Prakash, A., Saha, S., 2014. Experiments on document chunking and query formation for plagiarism source retrieval—notebook for PAN at CLEF 2014. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Rafiei, J., Mohtaj, S., Zarrabi, V., *et al*., 2015. Source retrieval plagiarism detection based on noun phrase and keyword phrase extraction—notebook for PAN at CLEF 2015. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Robertson, S.E., 1997. Overview of the Okapi projects. *J. Docum.*, **53**(1):3-7. https://doi.org/10.1108/EUM0000000007186

Suchomel, Š., Brandejs, M., 2015. Improving synoptic querying for source retrieval—notebook for PAN at CLEF 2015. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Toutanova, K., Klein, D., Manning, C.D., *et al*., 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. Proc. Conf. of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, p.173-180. https://doi.org/10.3115/1073445.1073478

Williams, K., Chen, H.H., Choudhury, S.R., *et al*., 2013. Unsupervised ranking for plagiarism source retrieval—notebook for PAN at CLEF 2013. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Williams, K., Chen, H.H., Giles, C.L., 2014a. Supervised ranking for plagiarism source retrieval—notebook for PAN at CLEF 2014. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.

Williams, K., Chen, H.H., Giles, C.L., 2014b. Classifying and ranking search engine results as potential sources of plagiarism. Proc. ACM Symp. on Document Engineering, p.97-106. https://doi.org/10.1145/2644866.2644879

Zubarev, D., Sochenkov, I., 2014. Using sentence similarity measure for plagiarism source retrieval—notebook for PAN at CLEF 2014. Proc. CLEF Evaluation Labs and Workshop, Working Notes Papers.