Using sentence similarity measure for plagiarism source retrieval

Notebook for PAN at CLEF 2014

Denis Zubarev[1] and Ilya Sochenkov[2]

[1] Institute for Systems Analysis of Russian Academy of Sciences, Moscow, Russia
[2] Peoples' Friendship University of Russia, Moscow, Russia
zubarev@isa.ru, isochenkov@sci.pfu.edu.ru

Abstract This paper describes a method that was implemented in the software submitted to PAN 2014 competition for the source retrieval task. For generating queries we use the most important noun phrases and words of sentences selected from a given suspicious document. To download documents that are likely to be sources of plagiarism we employ a sentence similarity measure.

1 Introduction

The plagiarism detection track on PAN [3] is divided in two subtasks: source retrieval and text alignment. Detailed information about these tasks is provided in [7]. In this paper we present a method that is supposed to solve the former task. The search engines (Indri, ChatNoir) are used to retrieve a candidate source for a suspicious document. We formulate a set of queries for search engines based on sentences of a suspicious document. The process of formulating queries is very important because it determines the maximum possible recall that can be achieved by a source retrieval method. One of goals of this paper is to explore an influence of using a phrasal search on recall. We need also to pay attention to filtering of incorrect source candidates to keep precision high enough. It allows to save computational resources during second task solution. We employ a sentence similarity measure for filtering source candidates. If a candidate contains a sentence that is quite similar to some suspicious sentence we consider such a candidate as a source, otherwise the candidate is filtered. An another goal is to minimize amount of queries to maximum possible extent. To achieve this goal we use a small amount of suspicious document sentences for generating a set of queries and we actively filter the queries based on downloaded sources.

The rest of this paper is organized as follows: Section 2 provides the details of the source retrieval method. Section 3 describes used sentence similarity measure. Section 4 provides information about conducted experiments. Section 5 presents the performance of the software in PAN 2014 competition. Section 6 concludes the paper.

2 Source retrieval method

The source retrieval method consists of several steps:

1. suspicious document chunking
2. query formulation
3. download filtering
4. search control

These steps are almost identical to those ones described in [7]. The algorithm 1 shows a pseudocode of our source retrieval algorithm, which we describe in details in the sections below.

In our method we use linguistic information such as forms of a word and syntactic dependencies of words. We use the dependency tree of a sentence to construct two-word noun phrases. If a noun is linked to another word and there are no words between them, then we consider such a structure as a phrase. Forms of words are used for measuring sentence similarity. To obtain linguistic information we use Freeling [1]. It performs document splitting, part-of-speech tagging, and dependency parsing.

We use TF-IDF weighting in our method, hence we formed a collection for calculating IDF weights of words. This collection consisted of English Wikipedia's articles (about 400,000 documents) that were chosen randomly. We suppose that this collection is sufficient for our tasks, since it allows us to distinguish words that are regularly used in any writing from some important ones.

Algorithm 1 General overview of source retrieval method

```
function SourceRetrieval($doc$)
    $misuses \leftarrow \varnothing$
    $sentences \leftarrow$ Split($doc$)
    for all $s \in sentences$ do
        $DownloadedDocs \leftarrow \varnothing$
        if $s \notin misuses$ then
            $query \leftarrow$ FormQuery($s$)
            $results \leftarrow$ SubmitQuery($query$)              ▷ snippets and urls are returned
            for all $r \in results$ do
                if FindSimilar($sentences, r$) $\neq \varnothing$ then
                    $DownloadedDocs \leftarrow$ Download($r$)
                end if
            end for
            for all $d \in DownloadedDocs$ do
                $pairs \leftarrow$ FindSimilar($sentences, d$)       ▷ pairs of similar sentences
                for all $(suspSent, downSent) \in pairs$ do
                    $misuses \leftarrow suspSent$
                end for
            end for
        end if
    end for
end function
```

2.1 Suspicious document chunking

Firstly, the suspicious document is split into sentences. To calculate a weight of a sentence we sum weights of all words and phrases in the sentence. The score of a word is obtained using TF-IDF weighting. The TF weight is calculated using the equation (4) from section 3. The IDF weight is obtained by using the equation (2) from section 3. Words that comprise a two-word noun phrases do not contribute in an overall sentence weight. A phrase weight is calculated as follows: $W_{phr} = 2(W_h + W_d)$, where W_h is the weight of a head word and W_d is the weight of a dependent one.

After calculating the weight of all sentences we select some sentences using different filters. Firstly, a sentence weight must exceed 0.45 value, since the low weight points to insignificance of information. Other parameters are taken into consideration, such as the maximum and minimum amount of words (excluding prepositions, conjunctions, articles) per sentence. Rationale for this is to exclude short sentences that may have rather high similarity with some sentence only because of sharing the most weighted words. Very large sentences are typically either errors of splitting or large lists. It is not likely to fetch a snippet that contains the whole large sentence, so we omit them. N sentences with the highest scores, which satisfied these criteria, are selected for further analysis. We call them suspicious sentences.

2.2 Query formulation

Before formulating queries we delete articles, pronouns, prepositions as well as duplicate words or phrases from each selected sentence. We use two available search engines Indri [10] and ChatNoir [8] for submitting queries. Sentences which contain phrases are submitted to Indri. For that we take 6 the most important (the most weighted) entities (phrases or words) from the sentence. The phrases are wrapped up by the special operator to leverage the phrasal search supported by Indri. If the sentence does not contain any phrases, 6 of the most weighted words are submitted to ChatNoir as a query. The formed queries are sequentially submitted to the search engines. This scheme is similar to approach that was used by Elizalde [2] in PAN 2013.

2.3 Download filtering

Seven first snippets returned by a search engine are downloaded and preprocessed by means of Freeling. Then we calculate the similarity between suspicious sentences and sentences extracted from the snippets. A method described in section 3 is used for measuring the similarity. If the similarity between any suspicious sentence and a snippet sentence exceeds $MinSim$, then a document to which a snippet belongs is scheduled for downloading. Such document is considered to be a source document.

2.4 Search control

To filter the rest of the queries we use downloaded documents. After downloading sources are subjected to preprocessing by Freeling. Then again we calculate the similarity between suspicious sentences and sentences extracted from the downloaded documents. If there is a sentence similar to a suspicious one, the latter is marked as a misuse.

The misuses are not used in the query formulation process, since their sources have already been found. This approach is similar to one used by Haggag [4].

3 Sentence similarity measure

Let us introduce the method for measuring sentence similarity. Given two arbitrary sentences s_e and s_t from documents d_e and d_t respectively, denote as $N(s_e, s_t)$ a set of word pairs with the same lemma, where the first element is taken from s_e and the second one from s_t. We compare two sentences by considering word pairs from the set $N(s_e, s_t)$. For calculating an overall similarity measure of two sentences we compute multiple similarities measures and then combine its values. The employed similarities are described below.

3.1 IDF overlap measure

Similar to [6] we define IDF overlap as follows:

$$I_1(s_e, s_t) = \sum_{(w_e, w_t) \in N(s_e, s_t)} IDF(w_e) \tag{1}$$

$$IDF(w_e) = \log_{|D|}\left(\frac{|D|}{m(w_e, D)}\right), \tag{2}$$

where D is a set of documents and $m(w_e, D)$ is an amount of documents that contain the word w_e. For correctness sake we assume that if $m(w_e, D) = 0$, then $IDF(w_e) = 1$.

3.2 TF-IDF-based similarity measure

Let us define TF-IDF-based measure in the following way:

$$I_2(s_e, s_t) = \sum_{(w_e, w_t) \in N(s_e, s_t)} f(w_e, w_t) IDF(w_e) TF(w_t, d_t) \tag{3}$$

$$TF(w_t, d_t) = \log_{|d_t|}\left(k(w_t, d_t)\right) \tag{4}$$

where $|d_t|$ is an amount of words in a document d_t and $k(w_t, d_t)$ is an amount of the word w_t in a document d_t. $f(w_e, w_t)$ is a kind of a penalty for mismatch of w_e, w_t forms :

$$f(w_e, w_t) = \begin{cases} 1.0, \text{if forms of the words are the same} \\ 0.8, \text{otherwise} \end{cases} \tag{5}$$

3.3 Sentence syntactic similarity measure

To be able to measure syntactic similarity we need to generate syntactic dependency tree from each sentence. We define $Syn(s_e)$ as a set that contains triplets (w_h, σ, w_d), where w_h, w_d are normalized head and dependent word respectively, σ is type of syntactic relation. We define syntactic similarity in the following way:

$$I_3(s_e, s_t) = \frac{\sum\limits_{(w_h,\sigma,w_d)\in(Syn(s_e)\cap Syn(s_t))} IDF(w_h)}{\sum\limits_{(w_h,\sigma,w_d)\in Syn(s_e)} IDF(w_h)} \quad (6)$$

Rationale for using syntactic information is to treat sentences not as a bag-of-words but as syntactically linked text. The syntactic similarity will be low for sentences in which the same words are used but they are linked in a different way. This measure is quite similar to one, described in [5].

3.4 Overall sentence similarity

The overall sentence similarity we define as a linear combination of described measures.

$$Sim(s_e, s_t) = WIdf * I_1(s_e, s_t) + WTfIdf * I_2(s_e, s_t) + WSynt * I_3(s_e, s_t), \quad (7)$$

where $WIdf$, $WTfIdf$, $WSynt$ determine relative contributions of each similarity. If $Sim(s_e, s_t) > MinSim$, then the suspicious sentence is considered as the plagiarised sentence. The process of tuning these four parameters is described in section 4.

4 Experiments

Experiments were run on the training corpus provided by the PAN organizers (about 100 documents of Webis-TRC-12 [9]). The source retrieval method described in section 2 is highly tunable and has multiple parameters. Some parameters were tuned separately, namely parameters that are involved in the process of the query generation (the amount of words and phrases in query, document chunking parameters). One of these parameters was varied while the others were fixed. The values that performed well in terms of the F-measure were chosen for further experiments. The ChatNoir oracle was used for the source retrieval evaluation. The found parameters are shown in table 1. The experiments showed that using only words for the candidates retrieval decreased recall by over 30%.

There are many tunable parameters in the method for measuring sentence similarity. For tuning these parameters we fixed parameters that are involved in the query generation process. All possible snippets and full document texts were fetched for queries that were formulated by using the fixed parameters. The downloaded data were preprocessed by Freeling and prepared for loading by our software. Using such preprocessed data, multiple combinations of $MinSim$, $WIdf$, $WTfIdf$, $WSynt$ parameters were tried. In the end we chose a combination that gave the best F-measure. The obtained parameters are shown in table 2.

Table 1. The chosen parameters that are involved in the query generation

Parameters	Values
minimum weight of a sentence	0.45
minimum amount of words in a sentence	11
maximum amount words in a sentence	33
maximum amount of suspicious sentences	83
words and phrases per query	6

Table 2. The tuned parameters for the measuring sentence similarity

sentences for measuring similarity	MinSim	WIdf	WTfIdf	WSynt
snippet sentences	0.5	0.4	0.4	0.2
sentences from a downloaded document	0.4	0.5	0.0	0.5

5 Results

Table 3 shows results of our software on the test data (about 100 documents of Webis-TRC-12 corpus [9]). Results were obtained by means of the evaluation platform TIRA [3].

Table 3. PAN 2014 Source retrieval final results

Submission	**Retrieval Performance**			**Workload**		**Time to 1st Detection**		**No Detection**
	F_1	Precision	Recall	Queries	Downloads	Queries	Downloads	
zubarev14	0.45	0.54	0.45	37.03	18.62	5.4	**2.25**	3
suchomel14	0.11	0.08	0.40	**19.05**	237.3	**3.1**	38.6	**2**
williams14	**0.47**	**0.57**	0.48	117.13	**14.41**	18.82	2.34	4
kong14	0.12	0.08	0.48	83.5	207.01	85.7	24.9	6
prakash14	0.39	0.38	**0.51**	59.96	38.77	8.09	3.76	7
elizalde14	0.34	0.40	0.39	54.5	33.2	16.4	3.9	7

As can be seen from Table 3, our software achieved F1 score of 0.45. It was the second highest achieved the F1 score by all participants.

An average of 37.03 queries were submitted per suspicious document and 18.62 results downloaded. The amount of both queries and downloaded results were relatively low in comparison with the other softwares particapated in PAN. Only 37 sentences were transformed into queries from 83 selected sentences on average. Such heavy filtering was crucial for achieving relatively high precision. It was experimentally found that the query filtering decreased recall but on the other hand significantly increased precision. According to results our software downloaded $18.62 * 0.54 = 10.05$ true positives for one suspicious document on average. This means that at least 3.7 queries were required for retrieving true positives results. This result proved that using phrases for candidate retrieval is quite reasonable and effective. However, the amount of queries to first detection was 5.4. It seems that ranking of sentences according to its weight was not the best strategy. Nevertheless, indicators that are measuring time to first detection were also low in comparison with the other participants results. On average, 2.25 full texts were downloaded until the first correct source document. It suggests that the snippets filtering based on employing sentence similarity measure worked relatively well.

No plagiarism sources were detected for 3 suspicious documents, which was about 3% of the suspicious documents in the test corpus. This result shows that the software is able to retrieve sources of plagiarism for the majority of documents.

6 Conclusions

This paper describes the source retrieval method that can achieve relatively high retrieval performance while minimizing the workload. It is possible due to employing two approaches, namely the phrasal search for the retrieving of candidates and using sentence similarity measure for the filtering of the candidates.

Many search engines support phrasal search to some extent. Some of them (e.g. Yandex, Yahoo) provide a proximity search which makes it possible to search phrases, and the other search engines (e.g. Google) provide the exact phrase search.

The filtering that is based on the sentence similarity measure works well when a snippet is an original sentence (or some part of it). If snippets contain heavily overlapped fragments of one sentence divided by a delimiter, then it is hardly useful to employ sentence comparison. We occasionally experienced this issue during experiments with ChatNoir snippets. But we believe that snippets that are provided by the popular search engines (e.g. Google, Yandex) contain the original sentences. Therefore results of our method are supposed to be reproducible in real-world environment.

However, there is some room for further improvements. It is probably worth reducing the set of phrases only to collocations. The rationale for this is based on an assumption that it is very easy to change a phrase using synonym of a head or a dependent word. But one cannot simply change any word in collocation because the phrase will lose its meaning. So one needs to synonymize the whole phrase or to leave it as it is.

Acknowledgments

We gratefully acknowledge support by Russian Foundation for Basic Research under grant No. 14-07-31149 and the PAN Lab organizers for their effort in organizing PAN 2014.

References

1. Atserias, J., Casas, B., Comelles, E., González, M., Padró, L., Padró, M.: FreeLing 1.3: Syntactic and semantic services in an open-source NLP library. In: Proceedings of LREC. vol. 6, pp. 48–55 (2006)
2. Elizalde, V.: Using statistic and semantic analysis to detect plagiarism. In: CLEF (Online Working Notes/Labs/Workshop) (2013)

3. Gollub, T., Potthast, M., Beyer, A., Busse, M., Rangel, F., Rosso, P., Stamatatos, E., Stein, B.: Recent trends in digital text forensics and its evaluation. In: Information Access Evaluation. Multilinguality, Multimodality, and Visualization, pp. 282–302. Springer (2013)
4. Haggag, O., El-Beltagy, S.: Plagiarism candidate retrieval using selective query formulation and discriminative query scoring. In: CLEF (Online Working Notes/Labs/Workshop) (2013)
5. Liu, D., Liu, Z., Dong, Q.: A dependency grammar and WordNet based sentence similarity measure. Journal of Computational Information Systems 8(3), 1027–1035 (2012)
6. Metzler, D., Bernstein, Y., Croft, W.B., Moffat, A., Zobel, J.: Similarity measures for tracking information flow. In: Proceedings of the 14th ACM international conference on Information and knowledge management. pp. 517–524. ACM (2005)
7. Potthast, M., Gollub, T., Hagen, M., Tippmann, M., Kiesel, J., Rosso, P., Stamatatos, E., Stein, B.: Overview of the 5th International Competition on Plagiarism Detection. In: Forner, P., Navigli, R., Tufis, D. (eds.) Working Notes Papers of the CLEF 2013 Evaluation Labs (Sep 2013), http://www.clef-initiative.eu/publication/working-notes
8. Potthast, M., Hagen, M., Stein, B., Graßegger, J., Michel, M., Tippmann, M., Welsch, C.: ChatNoir: a search engine for the ClueWeb09 corpus. In: Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval. pp. 1004–1004. ACM (2012)
9. Potthast, M., Hagen, M., Völske, M., Stein, B.: Crowdsourcing interaction logs to understand text reuse from the web. In: Fung, P., Poesio, M. (eds.) Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 13). pp. 1212–1221. ACL (Aug 2013), http://www.aclweb.org/anthology/P13-1119
10. Strohman, T., Metzler, D., Turtle, H., Croft, W.B.: Indri: A language model-based search engine for complex queries. In: Proceedings of the International Conference on Intelligent Analysis. vol. 2, pp. 2–6. Citeseer (2005)