A Hybrid Architecture for Plagiarism Detection

Notebook for PAN at CLEF 2014

Demetrios Glinos

Computer Science, University of Central Florida, Orlando, Florida, United States
Advanced Text Analytics, LLC, Orlando, Florida, United States
glinos@eecs.ucf.edu

Abstract We present a hybrid plagiarism detection architecture that operates on the two principal forms of text plagiarism. For order-preserving plagiarism, such as paraphrasing and modified cut-and-paste, it contains a text alignment component that is robust against word choice and phrasing changes that do not alter the basic ordering. And for non-order based plagiarism, such as random phrase reordering and summarization, it contains a two-stage cluster detection component. The first stage identifies a maximal passage in the suspect document that is related to the source document, while the second stage determines whether the suspect passage corresponds to the entire source document or just to a passage within it. Three implementations of this architecture, involving a common text alignment component and three different cluster detection components, participated in the PAN 2014 Text Alignment task and performed very well, achieving very high precision, recall, and overall plagiarism detection scores.

1 Introduction

Plagiarism is the attempt to pass off another person's ideas as one's own. This concept involves both intent, which may be inferred from context, and also the ideas themselves. Moreover, since ideas typically involve shared concepts and their relationships to one another, the originality or uniqueness of ideas is also typically inferred from the language used to express those ideas. Not surprisingly, therefore, current research on plagiarism detection centers on what may be objectively identified in a potential plagiarism context, that is, on the detection of similar expressions of ideas. Put simply, the focus is on the similarity of passages between two documents.

Viewed in this context, there are two fundamentally different types of text plagiarism: order-based and non-order based. Order-based plagiarism involves cutting and pasting a passage from the source, as well as order-preserving modifications such as the use of synonyms, insertions and deletions, and minor differences in phrasing. In each such case, the basic order of the concepts presented is essentially the same in the source and plagiarized documents. Non-order based plagiarism also involves presenting source document concepts in the suspect document, but in this case they appear in a substantially different order. An example of this kind of plagiarism is summarization, in which concepts from throughout the source document are assembled in a passage of the plagiarized document. Another example involves the translation of the source document or

passage into another natural language which, owing to the different grammatical constructions used by different languages, can have the effect of presenting the concepts of the source document in a generally different order in the plagiarized document. Of course, not all summaries or translations are instances of plagiarism, as that is a matter of intent and context, as discussed above.

Given these differences, we propose a plagiarism detection architecture that comprises a text alignment component to detect order-based plagiarism and a separate and independent component, which we call a *clustering component* to detect non-order based plagiarism. We have implemented this architecture in three systems that possess a common text alignment component and differ in their clustering strategies. All three systems participated in the PAN[1] 2014 Text Alignment task and performed well on all test corpora for the task.

Our hybrid approach differs markedly from current practice that applies a common set of algorithms to all plagiarism types in a *seed-extend-filter* approach. For example, Torrejón and Ramos [8] and Suchomel *et al.* [7] employ sets of *n-grams*, while Kong *et al.* [3] uses sentence-level cosine similarity, to identify the seed or anchor points. And while Palkovskii and Belov [4] addressed summarization specifically, they did not employ a text alignment component and did not distinguish plagiarism types as we do. In our proposed architecture the text alignment component does not employ a *seed-extend-filter* approach, but the clustering component does so for both suspect and source passage detection.

The sections that follow describe the architecture in detail and present the experimental results obtained. Section 2 covers the system components, including all three versions of clustering module. Section 3 presents the experiments and results. Finally, our conclusions are presented in Section 4.

2 System Description

The top level architecture of the proposed system is shown in Figure 1. We consider this a hybrid system since the two principal detection components (alignment and clustering) operate independently and could each run in standalone mode. Nevertheless, they are not co-equals within the system. The clustering component is invoked only when the alignment component does not find any aligned passages within a given document pair. The individual components, including variations tested, are discussed separately in the subsections that follow.

2.1 Tokenization

The system operates on document pairs. Both the source document and the suspected plagiarism (hereafter "suspect") document are preprocessed by the Tokenizer component prior to submission to the detection modules. The Tokenizer first reads each file as a stream of UTF-8 bytes and converts non-printable ASCII characters, newlines, carriage returns, and tabs, to spaces. It then converts the text to lower case and splits the

[1] http://pan.webis.de

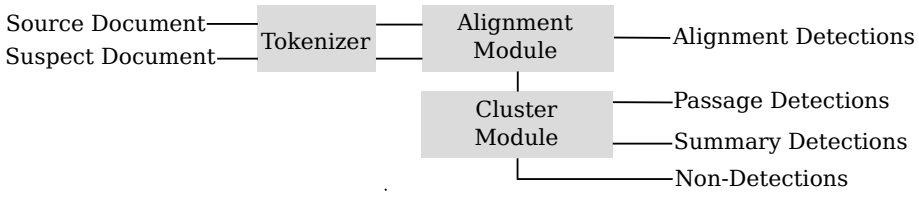

Figure 1. Top-level architecture of the proposed system.

input into tokens using a simple custom tokenization algorithm that separates all words, numbers, special characters, and punctuation, but which preserves punctuation within numerical values, the possessive *" 's"*, and the contraction *"n't"*.

2.2 Text Alignment

The Text Alignment component applies the algorithm described in [1] to find as many maximal-length token sequences as may be contained in the document pair. The algorithm extends the Smith-Waterman [6] dynamic programming algorithm, as modified by [2], by providing: (a) a recursive mechanism for detecting multiple alignments; (b) a method for handling large documents; (c) a method for joing adjacent subsequences; and (d) a similarity measure for comparing document tokens.

For all systems tested, aligned passages that comprised at least 40 tokens in both source and suspect documents were reported as alignment detections. All systems also used these dynamic programming parameters: +2 for token matches, and -1 for insertions, deletions, and misses. The systems were also configured to allow merging up to 2 sets of adjacent subsequences. A simple token similarity measure was also employed in which the articles *"a", "an",* and *"the"* were treated as matches, and the following common prepositions were also treated as equivalent: *"of", "in", "to", "for", "with", "on", "at", "from", "by", "about", "as", "into", "like", "through", "after", "over", "between", "out", "against", "during", "without", "before", "under", "around",* and *"among"*.

2.3 Clustering

A document pair for which no aligned sequences have been found may be an instance of either non-order based plagiarism or of no plagiarism at all. Therefore, to reduce the incidence of false positive detections, the principal idea underlying the proposed clustering approach is that the suspect document must contain a sufficiently dense cluster of terms from the source document that would be greater than the mere coincidental occurrence of common terms in the two documents. The goal is to distinguish, for example, two documents that contain similar expressions of concepts and their relationships, from two documents that merely discuss the same or similar topics. The three variations of the clustering that were tested are discussed separately below.

Basic Clustering The Basic Clustering algorithm seeks to find clusters in the suspect document that contain important concepts from throughout the source document while not performing any deep tagging or parsing of either document.

To approximate the set of important concepts, the algorithm finds the top 30 most frequent tokens in the source document that start with a letter and are of length 5 characters or more. The choice of 5 characters was a deliberate attempt to include most proper nouns, but to exclude determiners, personal pronouns, most prepositions, modals, and simple verbs. Once the most frequent terms are identified, the algorithm finds the set of all word trigrams centered on any of these terms, but excluding intermediate tokens that are punctuation or numeric. As a result, the trigrams in this set very often span more than 3 consecutive tokens.

The trigram set so constructed will be used if the set of trigram occurrences span a sufficient portion of the source document. Since this algorithm searches for summary passages, the concepts from the source are presumed to be drawn from throughout the source document. The algorithm uses 80 percent as the threshold value for this purpose.

If a trigram set exceeds the threshold, the algorithm then finds occurrences of the trigram set in the suspect document. Each trigram will represent an interval in the suspect document. The intervals may possibly overlap. Intervals that are within 20 tokens of each other are merged into clusters. If any clusters exceed 40 tokens in length, the largest such cluster is retained and reported as a summary detection.

It should be noted that this clustering algorithm does not search for a source passage corresponding to the suspect cluster, and so it will report either a summary detection or no detection at all.

Word Clustering The Word Clustering algorithm seeks to improve suspect document cluster detection and also to improve overall detection recall by determining whether the maximal suspect cluster is a summary of the entire source document or whether it corresponds merely to a passage within it.

In this algorithm, suspect cluster detection is improved in several ways. First, the set of source concepts is taken as the set union of the top 20 most frequent words of length 5 or more and also the top 20 most frequent tokens of such length that start with an uppercase letter in the original source document. Second, this algorithm uses bigrams instead of trigrams and dispenses with the concept spread threshold. Instances of the bigrams are then located within the suspect document and clusters are formed from bigram intervals that are within 15 tokens of each other. Finally, the best suspect cluster is selected from among the remaining clusters that are at least 40 tokens in length and which contain at least 8 of the source document terms, if any.

The algorithm selects the remaining suspect cluster that has highest Jaccard coefficient of value at least 0.65 for the source concept words and the content words in the suspect cluster. For this purpose, the content words for a cluster are the tokens in the cluster that begin with an alphabetic character, are at least 5 characters in length, and are not contained in the following list of stop words:*"the", "of", "and", "a", "in", "to", "is", "was", "it", "for", "with", "he", "be", "on", "i", "that", "by", "at", "you", "'s", "are", "not", "his", "this", "from", "but", "had", "which", "she", "they", "or", "an", "were", "we", "their", "been", "has", "have", "will", "would", "her", "n't", "there", "can", "all", "as", "if", "who", "what",* and *"said"*.

If a maximal suspect cluster is found, the algorithm attempts to find a similar passage in the source document based on the content words, determined as above, for the

suspect cluster. Occurrences in the source document are found and merged into clusters if within 15 tokens of each other, and clusters of at least 40 tokens are retained. Similarly to suspect cluster selection, the maximal source cluster is selected as the cluster that contains at least 8 suspect content words and has the greatest Jaccard coefficient computed using its content words and the suspect cluster content words, provided such value is at least 0.50.

Using this algorithm, if a suspect cluster is found and a source cluster is also found, then a source passage detection is reported. Otherwise, if a suspect cluster is found but no source cluster is found, then a summary detection is reported. And if no suspect cluster is found, no detection is reported.

Bigram Clustering The Bigram Clustering algorithm seeks to improve passage detection by extending the Word Clustering algorithm in cases where no source cluster is detected. First, the Word Clustering algorithm is applied as described above to detect suspect and source passages, with the exception that clusters need only contain 4 of the base concept words instead of 8 words.

In the event that a maximal suspect passage is found but no corresponding source passage is found, this algorithm computes all bigrams for the suspect passage. Bigrams and then Jaccard coefficients are then computed for the source clusters that were discovered by the Word Clustering algorithm. The maximal source cluster is selected to be the cluster with the highest Jaccard coefficient, provided such value is at least 0.25.

The outputs of this algorithm are the same as for the Word Clustering algorithm, that is, no detection if no suspect cluster is detected, and either a passage detection or a summary detection depending on whether or not a source passage is found.

3 Experiments and Results

The PAN Workshop series provides a valuable forum and test corpora for the evaluation of plagiarism detection systems. Three test corpora were used for the 2014 Text Alignment task. These corpora contain the numbers and mixes of plagiarism types for document pairs shown in the table below.

Table 1. Plagiarism type composition for PAN 2014 Text Alignment test corpora.

Plagiarism type	***Test Corpus 1***	***Test Corpus 2***	***Test Corpus 3***
No plagiarism	90	1000	1600
No obfuscation	108	1000	1600
Random obfuscation	94	1000	1600
Cyclic translation	105	1000	0
Summary obfuscation	121	1185	0
Total document pairs	518	5185	4800

Full details of the construction of the corpus are contained in [5]. Briefly, however, the plagiarism types are described as follows: The no plagiarism category is self-

explanatory. The no-obfuscation category represents cut-and-paste copying, although some differences in white space and line breaks are introduced. The random obfuscation category includes some amount of random text operations, such as word shuffling, adding or deleting words or phrases, and word replacement using synonyms. Cyclic translation involves translations of a document using automated translation services into two successive languages other than English, and then back into English. Finally, the summary category includes documents obtained from the Document Understanding Conference (DUC) 2006 corpus that have been processed to introduce noisy areas in addition to the summaries in the test documents.

The performance of all three systems against the test corpora is shown in Table 2. Overall, against all corpora, the Word Clustering system performed better than the Basic Clustering system, and the Bigram Clustering system performed best of all.

As shown in the table, precision was uniformly high at approximately 96% for all systems and all corpora. Recall varied, ranging from a low of approximately 76% for the Basic Clustering System running against Test Corpus 2, to values over 84% for the Word and Bigram Clustering systems against Test Corpus 3. Plagiarism Detection (*"PlagDet"*) scores ranged from a low of 84.30% for the Basic Clustering system against Test Corpus 2, to a high of 88.77% for the Bigram Clustering system against Test Corpus 3.

Table 2. Performance against PAN 2014 Text Alignment test corpora.

System	***Measure***	***Test Corpus 1 (518 docs)***	***Test Corpus 2 (5185 docs)***	***Test Corpus 3 (4800 docs)***
Basic Clustering	Recall	0.77088	0.76389	0.83473
	Precision	0.96735	0.96726	0.96243
	Granularity	1.01479	1.01756	1.01783
	PlagDet Score	0.84899	0.84300	0.88274
Word Clustering	Recall	0.79327	0.79105	0.84248
	Precision	0.96524	0.96339	0.96022
	Granularity	1.01441	1.01700	1.01767
	PlagDet Score	0.86192	0.85827	0.88626
Bigram Clustering	Recall	0.79469	0.79331	0.84511
	Precision	0.96599	0.96253	0.96007
	Granularity	1.01437	1.01695	1.01761
	PlagDet Score	0.86309	0.85930	0.88770

From the results, it is apparent that Test Corpus 3 presented an easier mix of test cases. This is not surprising, since this corpus did not include any instances of cyclic translation or summary obfuscation, which present many instances of non-order based plagiarism. Indeed, summary obfuscations are primarily non-order-based. Against this corpus, all three systems performed very well, with recall over 83%, precision over 96%, and PlagDet scores over 88%.

4 Conclusions

From the experimental results above, and from our experience in developing the systems that participated in the PAN 2014 Text Alignment evaluation, several conclusions are in order.

First, detecting non-order based plagiarism is a more difficult problem than order-based plagiarism. The marked improvement for the Basic Clustering system against Test Corpus 3 shows the impact of greater detections by the text alignment component.

Second, clustering does seem to be a beneficial approach for detecting non-order-based plagiarism. When comparing the results obtained here against those presented in [1], which used a text alignment component alone, the results are decidedly better.

And finally, there remains room for improvement. The recall values for the clustering systems tested are much less than their corresponding precision values, indicating to us that improved recall should be focus of future development efforts. Although we feel that this effort should be directed at non-order based plagiarism, we believe it is also worthwhile also to examine what improvements can also be made for order-based plagiarism.

References

1. Glinos, D.: Discovering Similar Passages Within Large Text Documents. (2014), submitted to CLEF 2014
2. Gotoh, O.: An Improved Algorithm for Matching Biological Sequences. In: Journal of Molecular Biology. vol. 162, pp. 705–708 (1981)
3. Kong, L., Qu, H., Du, C., Wang, M., Han, Z.: Approaches for Source Retrieval and Text Alignment of Plagiarism Detection–Notebook for PAN at CLEF 2013. 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
4. Palkovskii, Y., Belov, A.: Using Hybrid Similarity Methods for Plagiarism Detection–Notebook for PAN at CLEF 2013. 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
5. 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
6. Smith, T., Waterman, M.: Identification of common molecular subsequences. Journal of molecular biology 147(1), 195–197 (1981), http://www.bibsonomy.org/bibtex/2b9b41ee6be5e380bb59f67a3249ac8dd/hkayabilisim
7. Suchomel, Š., Kasprzak, J., Brandejs, M.: Diverse Queries and Feature Type Selection for Plagiarism Discovery–Notebook for PAN at CLEF 2013. 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. Torrejón, D., Ramos, J.: Text Alignment Module in CoReMo 2.1 Plagiarism Detector–Notebook for PAN at CLEF 2013. 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