External and Intrinsic Plagiarism Detection using a Cross-Lingual Retrieval and Segmentation System

Lab Report for PAN at CLEF 2010

Markus Muhr, Roman Kern, Mario Zechner, and Michael Granitzer

Know-Center Graz
{mmuhr,rkern,mzechner,mgrani}@know-center.at

Abstract We present our hybrid system for the PAN challenge at CLEF 2010. Our system performs plagiarism detection for translated and non-translated externally as well as intrinsically plagiarized document passages. Our external plagiarism detection approach is formulated as an information retrieval problem, using heuristic post processing to arrive at the final detection results. For the retrieval step, source documents are split into overlapping blocks which are indexed via a Lucene instance. Suspicious documents are similarly split into consecutive overlapping boolean queries which are performed on the Lucene index to retrieve an initial set of potentially plagiarized passages. For performance reasons queries might get rejected via a heuristic before actually being executed. Candidate hits gathered via the retrieval step are further post-processed by performing sequence analysis on the passages retrieved from the index with respect to the passages used for querying the index. By applying several merge heuristics bigger blocks are formed from matching sequences. German and Spanish source documents are first translated using word alignment on the Europarl corpus before entering the above detection process. For each word in a translated document several translations are produced. Intrinsic plagiarism detection is done by finding major changes in style measured via word suffixes after the documents have been partitioned by an linear text segmentation algorithm. Our approach lead us to the third overall rank with an overall score of 0.6948.

1 Introduction

Plagiarism detection has gained increased interest in research as well as in the industry [8] over the last couple years. The PAN challenge accommodated this fact and provided researchers a basis to compare different approaches.

We refrain from giving an introduction on plagiarism detection as Grozea et. al [2] formulated an excellent overview of the matter in last year's lab report. Instead we will discuss the motivation for our approach for this year's challenge.

The first three ranked participants in the first PAN competition all used a document-centric approach for external plagiarism detection as presented in [6]. Our approach in last year's competition was based on a block-level comparison of source and suspicious documents. Although our approach yielded acceptable results it was clear that the chosen block granularity, non-overlapping sentences, does not perform exceptionally

well. To identify similar suspicious and source blocks we used a simple cluster pruning technique which, while easy to implement, also introduced several problems.

To improve on our last approach we reformulated our problem solution slightly. Instead of comparing non-overlapping blocks of sentences we used overlapping blocks of tokens with fixed sizes. The cluster pruning technique was replaced by an open-source document search engine called Lucene [1]. Source documents are first split into overlapping blocks. Each block is then indexed by a Lucene instance. Suspicious documents are similarly split into overlapping blocks which get transformed to boolean Lucene queries. Each query results in a ranked list of potentially plagiarized source blocks.

We also reworked our post-processing step. We adapted an approach based on sequence analysis similar to dot-plot [7] with further heuristic merging and filtering steps to increase the overall precision of the system.

In last years competition none of the participants tried to solve the cross-lingual plagiarism subtask. We decided to give it a try in this year's challenge, building upon techniques developed in the machine translation community. We performed word alignment with the BerkleyAligner software package [5] using the Europarl corpus [4] to provide us with potential translations for each word in German or Spanish source documents.

Intrinsic plagiarism detection [1] seems to be a much harder task which is supported by the fact that less related work is available. Last year's competition supports this notion as only one competitor (Stamatatos [10]) could beat the baseline. Previous approaches used stylometric features or semi-stylometric features like character n-grams on sliding windows over the text to form a mean vector. A major difference of a certain block to this mean vector is expected to mark a style change which is interpreted as author change and therefore plagiarism. This year we tried to detect intrinsic plagiarism by adapting the text segmentation algorithm from Kern et. al. [3] to segment a document into stylometric coherent segments to identify plagiarism instead of topic coherent segments.

To sum up our system consists of the following basic ideas which is outlined by a flowchart in figure 1

- The external task is interpreted as retrieval task on a sub-document level
- Post-processing based on sequence analysis with merge and filter heuristics
- Translation on a word level by using word alignment of Europarl as translations
- Intrinsic plagiarism detection using text segmentation by detecting ad hoc flows in the text due to style changes
- Merging intrinsic and external plagiarism (intrinsic blocks are only taken, if there are none external ones for a specific document)

2 External Plagiarism Detection

Our external plagiarism detection approach consists of two main steps. In the first step we search for potentially matching suspicious document blocks within an inverted index of overlapping source document blocks. In the second step we apply heuristic post-processing on the potential matches to arrive at the final detection result. For Non-English source documents we have an additional pre-processing step to get translations

[1] http://lucene.apache.org/java/docs/index.html

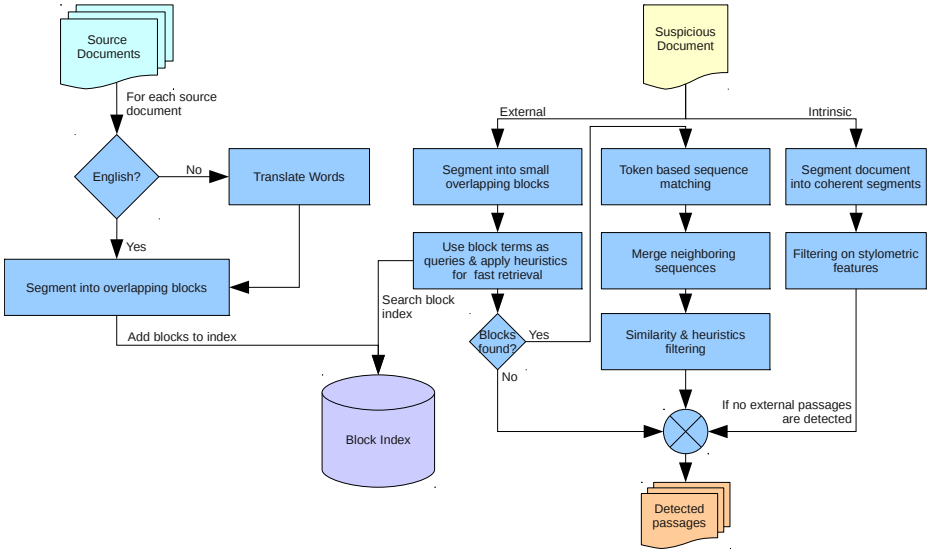

Figure 1. This flowchart gives an overview over the different stages of our plagiarism detection system. The left part shows the indexing process of the source blocks while the right side presents our hybrid system of intrinsic and external plagiarism detection.

for each word in the Spanish or German source documents. Furthermore, the post-processing step has to be slightly modified for translated plagiarism detection.

Translating Non-English Documents using Word Alignment To detect cross-lingual plagiarism we build upon techniques developed in the field of machine translation. Instead of applying a complete machine translation solution to translate whole documents or sentences we took the output of a word alignment algorithm. This kind of algorithm tries to find pairs of words that might be used as translation candidates and are a main component of many state-of-the-art machine translation systems. The base of the word alignment algorithms is a set of documents that are aligned on a sentence level. We used the Europarl aligned corpus (Release 5) [4]. To calculate the aligned words we employed the BerkeleyAligner[2] software package [5]. The output of the word alignment algorithm is a list of English translation candidates for the German and Spanish words present in the Europarl corpus. For each source document that is not written in English we replaced each word with up to 5 translation candidates. If no translation candidate is available, the word is not replaced. After the words have been replaced the documents are treated similar like the English source documents.

Overlapping Source Blocks Indexing The translated as well as non-translated source documents were each transformed into overlapping blocks of 40 tokens. For translated passages 40 refers to words in the original text, so that the actual translated passage may have more words due to the replacement with up to five translations. These blocks are

[2] http://code.google.com/p/berkeleyaligner/

indexed via a Lucene instance. Besides the text of each block we also stored additional information such as the offset and length of each block in the source document as well as the ID of the source document the block originated from.

Heuristics to limit Executed Suspicious Queries Similarly to the source documents the suspicious documents were tokenized and overlapping blocks of tokens were transformed to boolean queries. As this results in a massive amount of potential queries we applied several heuristics to limit the overall complexity of our approach while trying to keep the recall of correct hits reasonable high.

The first heuristic employed was the selection of a window step size of 6 and a window size of 16 tokens for each block. We arrived at this settings after testing various combinations on a small development corpus. Each query is only executed if at least one of the tokens in it has a normalized document frequency below a given threshold (0.004 for the evaluation corpus). Note that a document is really a block of a document.

The terms of the boolean query are first sorted by their corpus frequency in increasing order. The first four terms of this sorted list must be included (AND query), the other 12 terms (OR query) are determining the final rank of a hit in the list of ranked query results. By this technique we can ease the work load of the search engine as it can prune many documents due to the limitation that the four least frequent terms must be included in the query results. We further prune the query result by only using blocks which have a score above a certain threshold. After some testing on the development set we arrived at a threshold of 8.0. By adjusting these mentioned parameters, one can alter the trade-off between the number of potential hits (recall) and the number of query invocations (faster runtime).

Although translations have more words in common due to multiple translations for each word, we did not see major changes for the results when using the same parameter settings. Finally, we store the offset and length in the suspicious document for each executed query as well as the source document ID with offset and length for each block found for a query.

Post-Processing using Word Sequence Analysis with Merge Heuristics and Similarity Filtering The retrieved potential plagiarized blocks must now be refined and filtered. The advantage of our approach is that the locations in the source and suspicious documents are roughly known after the retrieval part, so that we can neglect a detailed post-processing on whole document pairs.

What we did so far was taking a suspicious document, split it into queries and search for potentially matching blocks for each query. As a first step we generate lists of query-block pairs. Note that a single query can have multiple matching blocks and thus generate several query-block pairs.

Given a query-block pair we extend the text around the query in the suspicious document as well as the text around the block in the source document by a number of characters (2000 for the evaluation set). Given the offset of both the query and the source document block we can align these two extended text passages. The alignment is given in form of a token by token matrix on which we apply the sequence analysis. A bigger window around the query-block pair leads to higher run-time, but can detect the passages more accurately. A sequence of tokens in both texts is a match if the sequence is composed of at least 3 consecutive tokens and has a length of at least 10 characters.

As with other settings we arrived at these by playing with the development set. For translated source documents the sequence analysis is a little bit less strict to compensate the incomplete translation we used. In this case the minimum length of sequence must be 6, but gaps are allowed between consecutive words in the suspicious document. Furthermore, the order of consecutive words in a suspicious document must not be the same in the source document, in other words a match must be found in a window of 10 tokens in the source document to count as a match.

The result is a list of sequence matches grouped by the source document they arose from. These matches are potentially small and we thus try to further merge them in a follow-up step. First we inspect all sequence matches originating from the same source document and eliminate those that do not have other sequence matches in their neighbor hood (defined as the surrounding 3000 characters) in said source document and are smaller than 50 characters.

Next we merge all sequence matches in the suspicious document that point to the same source document. Sequences are again matched via a neighbor hood criterion. However, this time the neighborhoods in both the suspicious document as well as the source document are considered. For sequence matches to be merged they have to be within a 500 character neighborhood in the suspicious document and within a 3000 character neighborhood in the source document. This can result in asymmetrically sized spans in the source and suspicious document. After this merging step we remove any matches smaller than 200 characters.

The final list of merged sequence matches are filtered on more time by calculating a Jaccard Similarity between the suspicious document sequence text and the source document sequence text. Sequence matches smaller than 5000 characters are eliminated if their similarity is smaller than 0.55, bigger matches must have a minimum similarity of 0.7 in order to be accepted.

3 Intrinsic Plagiarism Detection

The main idea of the intrinsic plagiarism detection algorithm is to detect changes in the style within a document. Base of our approach is a function that transforms a sequence of tokens (words, punctuations, ...) into a set of features that should represent the style of the author. Many different stylometric features have been proposed in the past. Among them are feature transformation functions based on parts of the word, for example character n-grams. Other stylometric features are constructed by using just a subset of words or tokens, for example pronouns. The usage and frequency of punctuation marks have also been investigated as a proxy for specific writing styles.

For our intrinsic plagiarism detection system we experimented with two different stylometric feature functions:

- Stop-words - This feature transformation is motivated by the intuition that different authors tend to resort to different stop words to construct grammatically correct sentences. For this feature transformation function to work all words need to be annotated as either function word or content word. This is accomplished by looking up all words in a manually crafted stop word list. All words that have been identified as stopword are added to the feature set and their frequency is additionally

recorded. The remaining function words are ignored by this feature transformation function. As stop word list we took the stop word lists from the Snowball stemmer project. These list are available in a number of languages and also contain the set of pronouns.

- Stem-Suffix - The last characters of words have already been used to identify specific author styles. The motivation for this feature transformation is the assumption that different authors may differ in their use of flections. One possible approach for this kind of function is to pick the last n characters of each words, where n is usually set to 3. For our system we used a flexible number of suffix characters. The suffix was determined by the number of characters a stemming algorithm would cut off or replace (we utilized the Snowball stemmer for this task). Finally this function produces a set of word suffixes.

To detect areas within a document that are written in a different style we first created a feature set out of the complete document, which can be seen as centroid of the whole document. This procedure assumes that the document is mostly written by a single author and the plagiarized sections do not cover the majority of the document. Next the document is split into blocks. This step is needed because the set of generated features by the stylometric transformation function tend to be sparse if applied on the sentence level. Partitioning the document into blocks of equal size (number of sentences) would be the most straightforward way to achieve this. For our system we have chosen to make use of a linear text segmentation algorithm [3]. Instead of producing blocks of equal size, the output of this algorithm is a list topically coherent blocks of multiple sentences. To identify changes in topics all words are first filtered out for stop-words and then stemmed. For each of the identified blocks a stylometric feature representation is generated. This set of features is then compared with the document-wide feature set, by calculating the cosing similarity. If the difference of the two feature sets exceeds a certain threshold, the block is considered to be written by another author than the majority of the document.

4 Results

This section starts with a short summary of the multiple parameters of our approach. Afterwards, we present and discuss detailed results on the development and on the evaluation corpus. We evaluate the quality of the retrieval step as well as provide performance measures for different obfuscation levels (none, low, high), for translated and non-translated plagiarism. Furthermore, we will compare our performance on external and intrinsic plagiarism.

4.1 Parameter Settings

We have multiple parameters used on each step of our approach. Since they have all been mentioned in the detailed description of our method, we just want to summarize them in table 4.1. We have to admit that our approach has many parameters, but as a matter of fact we did not really optimize them with the exception of the merge and

filter parameters. The parameters for the index and search step affect only the trade-off between precision and runtime, so a setting was used that gives a good precision with a reasonable runtime.

Table 1. Parameter settings for the external task.

parameter	*value*	workflow step
sliding block length (source)	40	indexing
sliding block step (source)	20	indexing
sliding query length (suspicious)	16	searching
sliding query step (suspicious)	6	searching
minimum document frequency (suspicious)	0.004	searching
minimum ranking score (suspicious)	8.0	searching
must occurence of least prominent tokens (suspicious)	4	searching
window extension	2000	sequence matching
minimum matching tokens (non-translated)	3	non-translated sequence matching
minimum length of match (non-translated)	10	non-translated sequence matching
minimum matching tokens (translated)	3	translated sequence matching
minimum length of match (ranslated)	10	translated sequence matching
maximum gaps in matching tokens (translated)	2	translated sequence matching
window for matches (translated)	10	translated sequence matching
island threshold	50	sequence filtering
suspicious merge offset	500	sequence merging
source merge offset	3000	sequence merging
final valid source difference	1500	sequence filtering
minimum block length	200	sequence filtering
similarity treshold small (<5000)	0.55	sequence filtering
similarity treshold large (>5000)	0.70	sequence filtering

As processing unit we used a desktop machine with a Intel Core 2 quad-core CPU (2.66 GHz) with 8 GB DDR2 RAM and a 1 Terra-byte hard disk.

4.2 Development Corpus

The development corpus was the same one as in the first PAN competition from 2009 (a detailed description can be found in [9]). Since our approach is not very fast, we optimized our approach on a sub set of 500 suspicious documents, so the following performance measures are computed only for this sub set. However, note that as basis for the retrieval step still the complete set of source documents was used.

In table 4.2 the results of the retrieval step are shown. Basically the table shows the number of all real plagiarism blocks in the data set and the ones which are hit at least by one query - block pair. In other words, these are the blocks our post-processing step can extract at best (upper bound on the recall). These results show that we only loose a considerable amount of the high obfuscated blocks, but a very high percentage (above 90 %) of low or none obfuscated as well as translated plagiarism have been hit after the retrieval step. The difference for high obfuscated plagiarism can be explained by the fact that we do not handle for example synonyms which represent a big part of the deterioration of high-obfuscated plagiarism. The retrieval step delivered a total of

6461076 query - block pairs, from which 4614485 are correct (are partially overlapping with a real plagiarism block), so 71.42 % of the pairs are correct ones.

Table 2. Performance evaluation of retrieval step for a sub set of 500 suspicious documents of the development corpus. The table shows values for the number of blocks that have been hit at least by one query - block pair in the retrieval step, the number of real plagiarism blocks in the suspicious documents and a ratio between these two.

task	*hit*	*all*	*ratio*
high	2543	3676	0.6918
low	6614	6988	0.9465
none	9381	9592	0.9780
translated	2349	2543	0.9237

Furthermore, in table 4.2 a detailed evaluation of our detected plagiarism after the post-processing steps (sequence analysis, merge heuristics, similarity thresholding) is shown. Surprisingly our cross-lingual approach performs better than the plain English one in terms of recall and precision with a worse granularity. However, if we take a closer look this is mostly because of our quite bad performance on highly obfuscated plagiarism. High obfuscation means a high level of paraphrasing, exchanging words by synonyms, etc. which we do not deal with specifically. On the other hand for low and none obfuscation our results are better than the cross-lingual approach.

Table 3. Performance results of detected plagiarism separated by different sub-tasks for a sub set of 500 suspicious documents of the external development corpus and for the whole set of the intrinsic development corpus. Performance measures include precision, recall, granularity and the overall score.

task	*Precision*	*Recall*	*Granularity*	*Score*
non-translated all	0.5623	0.8105	1.071	0.6321
non-translated none	-	0.9619	1	-
non-translated low	-	0.8338	1.1709	-
non-translated high	-	0.4706	1.0	-
translated	0.8441	0.732	2.2785	0.4576
external	0.6211	0.7977	1.2618	0.5931
intrinsic	0.4709	0.245	1.0006	0.3221

4.3 Evaluation Corpus

In contrast to the development corpus the evaluation corpus does not explicitly separate external and intrinsic plagiarism, since in this years competition the winner should develop a hybrid system that can handle both types of plagiarism. Nevertheless, the following evaluation will distinguish between the different kinds of plagiarism considered as sub-tasks of the global problem. A detailed description of this corpus can be found in the overview paper.

In table 4.3 the results of the retrieval step for the evaluation corpus are shown. In contrast to the results on the development corpus even for high obfuscated plagiarism we hit most of the blocks and the difference to low and none obfuscated passages

is much less significant. However, for low and none obfuscation the results are also higher compared to the development corpus. Surprisingly, translated blocks are a little bit worse, but since we only used a split of 500 of the development corpus the amount of translated passages will most likely be smaller, so that the difference lies in statistical limits of variance. The retrieval step delivered a total of 1.7642292E7 query - block pairs, from which 1.4431375E7 are correct (are partially overlapping with a real plagiarism), so the ratio increased as well from 71.42 % to 81.8 %.This shows that heuristics like ranking score seem to be reasonable good to achieve very high recall values with a very good precision in this initial step.

Table 4. Performance evaluation of retrieval step on the evaluation corpus. The table shows values for the number of blocks that have been hit at least by one query - block pair in the retrieval step, the number of real plagiarism blocks in the suspicious documents and a ratio between these two.

task	*correct*	*maximum*	*ratio*
high	13348	14756	0.9046
low	14832	14883	0.9966
none	16784	16784	1.0
translated	5462	6314	0.8651

Furthermore, in table 4.3 a detailed evaluation of the final detected plagiarism blocks are shown. In contrast to the results on the development corpus our cross-lingual approach performs worse than the plain English plagiarism detection. This can be explained by the fact that on the evaluation corpus the recall on high-obfuscated plagiarism detection increased to 0.8122 compared to 0.4706 on the development corpus. For low and none obfuscation the recall values are in similar ranges, so that the overall evaluation on non-translated plagiarism detection could be increased considerable. Again it can be recognized that our performance on the translated blocks are especially bad concerning granularity, so it seems to be the case that there are several holes in the detected passages. As expected our performance on the intrinsic plagiarism detection sub-task is very poor and might have deteriorated our overall result more than expected.

Table 5. Performance results of detected plagiarism separated by different sub-tasks for the hybrid evaluation corpus. Performance measures include precision, recall, granularity and the overall score.

task	*Precision*	*Recall*	*Granularity*	*Score*
non-translated all	0.9299	0.8967	1.0553	0.8785
non-translated none	-	0.9497	1.0025	-
non-translated low	-	0.9207	1.0968	-
non-translated high	-	0.8122	1.0771	-
translated	0.8036	0.61616	2.1655	0.4195
external	0.9053	0.8631	1.1611	0.7949
intrinsic	0.212	0.1566	1.0	0.1802
Overall	0.8417	0.7057	1.1508	0.6948

Despite the good results we have to admit that our current implementation is not very fast. The whole process takes about a week. However, we did not optimize our approach in any kind. There are many possible ways to improve our approach. For example, we can utilize some prefiltering on document basis or try to distribute our approach.

5 Conclusion

We scored the 3rd place in the overall ranking of all systems, with our recall values being very close to the winning system. The precision of our system is still an area to be improved in future iterations. We attribute the performance of your system with respect to precision mainly to the poor results from the intrinsic plagiarism detection system which lowers the overall precision considerably. The post-processing step also needs some more tuning as evidenced by the poor granularity achieved.

We plan on transforming our approach into a web-service, seeded by the articles of Wikipedia as a source corpus. Handling other types of plagiarism such as stealth approaches or missing citations are also on our agenda. We'd also like to increase the scalability and performance of our system by employing distributed indices along with document-level cluster pruning for large datasets.

References

1. zu Eissen, S.M., Stein, B.: Intrinsic plagiarism detection. In: ECIR. Lecture Notes in Computer Science, vol. 3936, pp. 565–569. Springer (2006)
2. Grozea, C., Gehl, C., Popescu, M.: ENCOPLOT: Pairwise Sequence Matching in Linear Time Applied to Plagiarism Detection. In: 3rd PAN Workshop. Uncovering Plagiarism, Authorship and Social Software Misuse. p. 10 (2009)
3. Kern, R., Granitzer, M.: Efficient linear text segmentation based on information retrieval techniques. In: MEDES '09. pp. 167–171. ACM (2009)
4. Koehn, P.: Europarl: A parallel corpus for statistical machine translation. MT summit 5, 12–16 (2005)
5. Liang, P., Taskar, B., Klein, D.: Alignment by agreement. In: Proceedings of the Human Language Technology Conference of the NAACL. pp. 104–111 (June 2006)
6. Lyon, C., Barrett, R., Malcolm, J.A.: A theoretical basis to the automated detection of copying between texts, and its practical implementation in the Ferret plagiarism and collusion detector. In: JISC (UK) Conference on Plagiarism: Prevention, Practice and Policies Conference. pp. 10–18 (2004)
7. Maizel, J., Lenk, R.: Enhanced graphic matrix analysis of nucleic acid and protein sequences. In: Proceedings of the National Academy of Sciences. pp. 7665–7669 (1981)
8. Maurer, H.A., Kappe, F., Zaka, B.: Plagiarism - a survey. J. UCS 12(8), 1050–1084 (2006)
9. Potthast, M., Stein, B., Eiselt, A., Barron-Cedeno, A., Rosso, P.: Overview of the 1st International Competition on Plagiarism Detection. In: 3rd PAN Workshop. Uncovering Plagiarism, Authorship and Social Software Misuse. pp. 1–9 (2009)
10. Stamatatos, E.: Intrinsic Plagiarism Detection Using Character n-gram Profiles. In: 3rd PAN Workshop. Uncovering Plagiarism, Authorship and Social Software Misuse. pp. 38–46 (2009)