Multi-feature Plagiarism Detection

Jan Kasprzak, Šimon Suchomel, and Michal Brandejs

Faculty of Informatics, Masaryk University
{kas,suchomel,brandejs}@fi.muni.cz

1 General Approach

Our approach in PAN 2012 Plagiarism detection—Detailed comparison sub-task is loosely based on the approach we have used in PAN 2010 [1].

2 Cross-lingual Plagiarism Detection

In the final form of the detailed comparison sub-task, the results of machine translation of the source documents were provided to the detector programs by the surrounding environment, so we have discarded the language detection and machine translation from our submission altogether, and used only line-by-line alignment of the source and translated document for calculating the offsets of text features in the source document. We have then treated the translated documents the same way as the source documents in English.

3 Multi-feature Plagiarism Detection

Our pair-wise plagiarism detection is based on finding common passages of text, present both in the source and in the suspicious document. We call them *common features*. In PAN 2010, we have used sorted word 5-grams, formed from words of three or more characters, as features to compare. Recently, other means of plagiarism detection have been explored: stopword n-gram detection is one of them [4].

We propose the plagiarism detection system based on detecting common features of various types, for example word n-grams, stopword n-grams, translated single words, translated word bigrams, exact common longer words from document pairs having each document in a different language, etc. The system has to be to the great extent independent of the specialities of various feature types. It cannot, for example, use the order of given features as a measure of distance between the features, as for example, several word 5-grams can be fully contained inside one stopword 8-gram.

We therefore propose to describe the *common feature* of two documents (susp and src) with the following tuple: $\langle \text{offset}_{\text{susp}}, \text{length}_{\text{susp}}, \text{offset}_{\text{src}}, \text{length}_{\text{src}} \rangle$. This way, the common feature is described purely in terms of character offsets, belonging to the feature in both documents. In our final submission, we have used the following two types of common features:

- word 5-grams, from words of three or more characters, sorted, lowercased

– stopword 8-grams, from 50 most-frequent English words (including the possessive suffix 's), unsorted, lowercased, with 8-grams formed only from the seven most-frequent words (*the, of, a, in, to, 's*) removed

We have gathered all the common features of both types for a given document pair, and formed *valid intervals* from them, as described in [2]. A similar approach is used also in [4]. The algorithm is modified for multi-feature detection to use character offsets only instead of feature order numbers. We have used valid intervals consisting of at least 5 common features, with the maximum allowed gap inside the interval (characters not belonging to any common feature of a given valid interval) set to 3,500 characters.

4 Postprocessing

In the postprocessing phase, we took the resulting valid intervals, and made attempt to further improve the results. We have firstly removed overlaps: if both overlapping intervals were shorter than 300 characters, we have removed both of them. Otherwise, we kept the longer detection (longer in terms of length in the suspicious document).

We have then joined the adjacent valid intervals into one detection, if at least one of the following criteria has been met:

– the gap between the intervals contained at least 4 common features, and it contained at least one feature per 10,000 characters[1], or
– the gap was smaller than 30,000 characters and the size of the adjacent valid intervals was at least twice as big as the gap between them, or
– the gap was smaller than 30,000 characters and the number of common features per character in the adjacent interval was not more than three times bigger than number of features per character in the possible joined interval.

These parameters were fine-tuned to achieve the best results on the training corpus. With these parameters, our algorithm got the total plagdet score of 0.73 on the training corpus.

5 Further discussion

As in our PAN 2010 submission, we tried to make use of the intrinsic plagiarism detection, but despite making further improvements to the intrinsic plagiarism detector, we have again failed to reach any significant improvement when using it as a hint for the external plagiarism detection.

In the full paper, we will also discuss the following topics:

– language detection and cross-language common features
– intrinsic plagiarism detection
– suitability of plagdet score[3] for performance measurement
– feasibility of our approach in large-scale systems
– discussion of parameter settings

[1] we have computed the length of the gap as the number of characters between the detections in the source document, plus the number of charaters between the detections in the suspicious document.

References

1. Kasprzak, J., Brandejs, M.: Improving the reliability of the plagiarism detection system. In: Notebook Papers of CLEF 2010 LABs and Workshops. Citeseer (2010)
2. Kasprzak, J., Brandejs, M., Kipa, M.: Finding plagiarism by evaluating document similarities. In: SEPLN'09: The 25th edition of the Annual Conference of the Spanish Society for Natural Language Processing (2009)
3. Potthast, M., Stein, B., Barrón-Cedeño, A., Rosso, P.: An Evaluation Framework for Plagiarism Detection. In: Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010) (to appear). Association for Computational Linguistics, Beijing, China (Aug 2010)
4. Stamatatos, E.: Plagiarism detection using stopword n-grams. Journal of the American Society for Information Science and Technology (2011)
5. Stamatatos, E.: Intrinsic plagiarism detection using character n-gram profiles. In: Proceedings of the SEPLN'09 Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse. pp. 38–46 (2009)