Developing High-Resolution Universal Multi-Type N-Gram Plagiarism Detector

Notebook for PAN at CLEF 2014

Yurii Palkovskii, Alexei Belov

Zhytomyr Ivan Franko State University, Institute of Foreign Philology
SkyLine LLC, Plagiarism Detector Project
vb-expert@yandex.ru cargo-base@yandex.ru

Abstract. This paper describes approaches used for the Plagiarism Detection task during PAN 2014 International Competition on Uncovering Plagiarism, Authorship, and Social Software Misuse, that scored 1-st place with plagdet score (0.907) for test corpus no.3 and 3-rd place score (0.868) for test corpus no. 2. In this work we aggregated all the previously researched experience from PAN12 and PAN 13 research works [2] and thus further improved previously developed methods of detecting plagiarism [8], with the help of: contextual n-grams, surrounding context n-grams, named entity based n-grams, odd-even skip n-grams, functional words frame based n-grams, TF-IDF sentence level similarity index and noise sensitive clusterization algorithm, focused summary type detection heuristics, combined into a single model to mark similarity sections and thus effectively detect different types of obfuscation techniques.

1 Introduction

Each year PAN competition pushes forward the baseline of plagiarism detection effectiveness thus making it harder and harder to compete with the top plagdet score becoming close to the absolute values irrespective of even more demanding plagiarism types included into the new corpora. This year we tried to incorporate all the knowledge we could aggregate via already published PAN works [3,2,4,5] and try to figure out how to make most of the algorithms that already proved to achieve best performance during the last years. The introduction of the lately developed TIRA platform [1] has greatly boosted the software deployment and thus made it easier both to participate in the competition, synchronize the local result and the result in the test environment.

2 Methods

The main goal of this year research was to aggregate all the most efficient methods [6,7] that have already been applied to the task of plagiarism detection. The dominating approach that showed best results and greatly improved recall was the

one demonstrated by Efstathios Stamatatos [7] - namely the introductions of stop-words n-gram generation. We combined the above idea with another idea introduced by Jan Kasprzak [3,4] of generating context based n-grams from both the exact sentence context, surrounding sentences n-gram extraction and n-gram generation based on cyclical word deletion. We also introduced "Named Entity injection as n-gram" method: basically it is not an n-gram in any manner, but we considered an option to somehow "inject" the Named Entity uniqueness into the general model and to achieve this we harvest all the Named Entities from the text and treat it as a regular n-gram, irrespective of the fact that it is not actually a typical n-gram that usually consists of multiple words.

We did such analysis and came up with the following methods of n-gram generation:

1. Regular n-grams
2. Variable length stop-word n-grams
3. Named Entity based n-grams
4. Most Frequently used words n-grams

N-grams expansion by:

1. Odd-even generation on a sentence level
2. Resulting n-gram set stemming
3. A single word deletion on a fingerprint level
4. N-grams alphasorting

Text preprocessing included special symbols removal and space trimming. The input text parsing was made on 2 levels: sentence level and word level.

We applied angled ellipse based graphical clustering algorithm [8] to define clusters of shared fingerprints. The main approach was to detect what kind of plagiarism is dominating within the document pair - by applying special kind of analysis that included global noise level detection, single stage analysis for the existence of verbatim plagiarism clusters, shared fingerprints sequence analysis, diagonal density analysis, summary type presence and then finally selecting which analysis preset will be used for final analysis. Our software selects one of 4 possible parameter presets for the final detection strategy:

1. Verbatim Plagiarism
2. Random Plagiarism
3. Summary type Plagiarism
4. Undefined type with high noise

The algorithm in detail:

1. Both d_1 as the 1st document and d_2 as the 2nd document are preloaded with the same preloading sequence.

2. The text is being preprocessed via removing all special characters and text control symbols with the initial offsets kept intact. Diacritics is being neutralized by dot net provided ".Normalize" method. At word delimitation stage, we also apply StringComparison.InvariantCultureIgnoreCase option to get better split quality. We used our own developed sentence split heuristics, based on a set of predefined rules for sentence delimitation. If global parameter for punctuation signs removal is on, the application removes punctuation marks, again with the offsets left intact.

3. Each sentence is being split into words. We use text representation with object structure, having access to each sentence and word by both value and offset, using hash tables and inverted indices with performance as a consideration.

4. We do quality preprocessing for each word if global flag is on again with the resulting word-length and offset being corrected to pertain globally correct offsets.

5. At this stage we generate all required n-grams. N-grams are formed using different strategies: structural n-grams, regular n-grams, stop-word based n-grams, Named Entity based n-grams, near-context based n-grams. We use different input for n-grams input values: words sequences of different lengths and Named Entities. The former being divided into several stages with a separate stage for a specific n-gram generation strategy [4,6,7,8]. All resulting n-grams are then translated to hashes using dot net built in .GetHash method and stored within hash table, with key being a hash and value being the n-gram offset.

6. We do a complete search within hash table of d_1 against hash table of d_2 aggregating the results into a list of "shared" n-grams.

7. At this stage we do a sentence-against-sentence vector space model statistical analysis. We pre-calculate word vector for each sentence and then we ran exhaustive comparison via sentence-against-sentence analysis with cosine similarity being a final comparison value stored as a result. If this similarity value exceeds a predefined global threshold, we add a final result cluster that points to the analyzed sentence pair. The interesting point is, that this stage results do affect the final results only at late stage 9.

8. For each "shared" n-gram detected we run a clustering algorithm to define the exact clusters of n-grams that are the final product of the developed application. Our clusterization algorithm has been already described in our previous publications [8].

9. Clustering algorithm is the most complicated part of the software and most computationally intensive procedure. The exact logic of this stage depends on a global strategy that is automatically selected in an attempt to better tackle a specific type of obfuscation.

10. We form initial clusters from detected "shared" n-grams via skewed eclipse based (last year we used circle-based type) offset based two dimensional plain graphical analysis.

11. We do vector space model analysis routines at this point.

12. We merge n-gram based detected clusters with VSM detected clusters at this point.

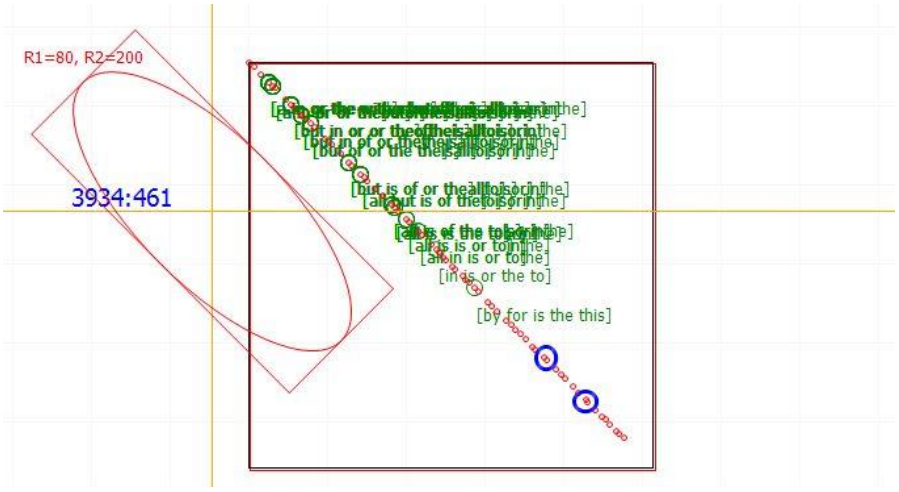
13. We compute n-gram ordering in all detected clusters.

14. We do 1st stage preprocessing for all detected clusters that includes: building n-gram distribution histogram [8], cluster removal from the current result list if cluster absolute length is less than a globally predefined value, cluster removal from the current result list if cluster "is inside" another large cluster.

15. We do final cluster remerge procedure: if clusters are close enough (using a predefined global value) then they are merged into a single cluster, thus adjusting granularity score.

16. We do final cluster list post-processing: removing all clusters that are too small in length (character-based), contain very few n-grams, do not have sufficient percent of n-grams located within the main cluster diagonal - so called low "insufficient n-gram diagonal distribution", "is vertical" filter that is made in order to mitigate PAN corpus caused malformed clusters and duplicating false positives.

17. Final results are formed from the remaining clusters. We use specially developed visualization tool to graphically assess resulting clusters and follow our methodology of "visualize-analyze-correct" strategy. Below there is a screenshot of the produced final analysis cluster graph, with sample skewed eclipse, that shows the exact "is-in-eclipse" n-gram clustering logic output. Structural n-grams are shown in green and are printed, regular n-grams are shown in red circles, the master cluster definition of gold standard xml-descriptor is marked by a red rectangle whereas vector space model detected cluster (marked in grey) fully coincides with finally detected cluster (marked in black), which, in its turn, coincides with the master cluster (marked in red). Shared Named Entities are marked in blue and act as a regular n-grams. The graph is formed using "Zero Obfuscation Strategy" (see 18 for more details):



18. The above mentioned steps from 1 to 17 are strategy dependent. That is, they use a special preset of parameters that best fits specific obfuscation type. At the initial analysis stage, the software analyses several critical input data characteristics: global noise level, cluster ordering, number of verbatim plagiarism clusters via the percent of clusters with absolute diagonal distribution. Then it selects one of the three possible parameter sets - "Zero Obfuscation Strategy", "Low Obfuscation Strategy, Low Noise", "High Noise Strategy", "High Obfuscation Strategy, High Noise Strategy" and recursively runs another analysis. Each of the abovementioned strategies use its

own globally predefined sets of values and switches that define the exact changes in routines in steps 1-17 mentioned above.

19. At this stage software compares the results of preliminary analysis that was used for strategy definition with the results received with strategy applied. Final result is chosen at this stage.

3 Evaluation

The developed software scored 1-st place with plagdet score (0.907) for test corpus no.3 and 3-rd place score (0.868) for test corpus no. 2. At the moment we are not able to assess and explain the difference in results as long as both corpora are not yet publically downloadable and we do not know the difference between the test corpora versions, we are waiting for the test data release to further research the difference in the achieved Recall and Granularity.

Corpus:	Plagdet:	Recall:	Precision	Granularity	Runtime:
Test 3	0.90779	0.88916	0.92757	1.00027	00:57:15
Test 2	0.86806	0.82637	0.92227	1.00580	01:10:04

4 Conclusions and Future Work

The result achieved at this year PAN competition shows really tight competition for every percent. Comparing our previous results to this year results we may conclude that we made really good progress landing between the best competing teams at PAN. Unfortunately, due to many different factors that go beyond the scope of the abovementioned research, we were not able to fully optimize the developed system. The majority of the parameters used were heuristically set to most optimal values we come with initially. This fact shows large space for future improvements. Applying genetic algorithm for global parameter tuning will definitely give much better results and will allow much better tuning for each specific corpus.

Acknowledgments. To the PAN RnD team, for their fantastic job making deploying and running software on TIRA platform possible, for their prompt responses and discrete guidance both on the competition website and Google groups. We would like to thank our competing teams as well, as our research was heavily based on and dramatically boosted by the knowledge and experience they shared in their previously published works and live workshop discussions at previous PANs. We also would like to thank the organizers for providing support and the official letter of invitation to be able to append PAN workshop aiding us with visa acquisition for the conference.

References

1. Martin Potthast, Benno Stein, Alberto Barrón-Cedeño, and Paolo Rosso. An Evaluation Framework for Plagiarism Detection. In 23rd International Conference on Computational Linguistics (COLING 10), August 2010. Association for Computational Linguistics.
2. Martin Potthast, Matthias Hagen, Tim Gollub, Martin Tippmann, Johannes Kiesel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. Overview of the 5th International Competition on Plagiarism Detection. In Pamela Forner, Roberto Navigli, and Dan Tufis, editors, Working Notes Papers of the CLEF 2013 Evaluation Labs, September 2013. ISBN 978-88-904810-3-1.
3. Rodríguez-Torrejón, D.A., Martín-Ramos, J.M.: "Detección de plagio en documentos: sistema externo monolingüe de altas prestaciones basado en n-gramas contextuales" (Plagiarism Detection in Documents: High Performance Monolingual External Plagiarism Detector System Based on Contextual N-grams). Procesamiento del Lenguaje Natural. N. 45 (2010).
4. Rodríguez-Torrejón D.A., Martín-Ramos J.M.: CoReMo System (Contextual Reference Monotony) A Fast, Low Cost and High Performance Plagiarism Analyzer System: Lab Report for PAN at CLEF 2010. In Braschler M., Harman D., Pianta E., editors. Notebook Papers of CLEF 2010 LABs and Workshops, 22-23 September, Padua, Italy, 2010.
5. Tim Gollub, Martin Potthast, Anna Beyer, Matthias Busse, Francisco Rangel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. Recent Trends in Digital Text Forensics and its Evaluation. In Pamela Forner, Henning Müller, Roberto Paredes, Paolo Rosso, and Benno Stein, editors, Information Access Evaluation meets Multilinguality, Multimodality, and Visualization. 4th International Conference of the CLEF Initiative (CLEF 13), September 2013. Springer. ISBN 978-3-642-40801-4.
6. Šimon Suchomel, Jan Kasprzak, and Michal Brandejs. Three Way Search Engine Queries with Multi-feature Document Comparison for Plagiarism Detection—Notebook for PAN at CLEF 2012. In Forner et al. [6]. ISBN 978-88-904810-3-1. URL http://www.clef-initiative.eu/publication/working-notes
7. Efstathios Stamatatos. Plagiarism Detection Using Stopword n-Grams. JASIST, 62(12):2512–2527, 2011. doi: http://dx.doi.org/10.1002/asi.21630.
8. Yurii Palkovskii and Alexei Belov. Applying Specific Clusterization and Fingerprint Density Distribution with Genetic Algorithm Overall Tuning in External Plagiarism Detection—Notebook for PAN at CLEF 2012. In Forner et al. [6]. ISBN 978-88-904810-3-1. URL http://www.clef-initiative.eu/publication/working-notes.