Evaluating AdaBoost for Plagiarism Detection

Thiago V. Reginaldo, Magali R. G. Meireles, and Zenilton K. G. Patrocínio Jr.[✉]

Pontifical Catholic University of Minas Gerais, Belo Horizonte, MG, Brazil
thiagopesquisa42@gmail.com, {magali,zenilton}@pucminas.br

Abstract. Plagiarism is a current problem with serious consequences. Recently, many research efforts have addressed plagiarism detection task. This problem is more difficult, when some obfuscation strategy is used. This work proposes and evaluates the adoption of AdaBoost for classifying suspicious text passages as plagiarism or not. We also present a simple post-processing heuristic for improving results granularity. Comparative analysis with other classification methods were conducted and experimental results have shown that Adaboost reached a F-measure of 0.93 and the best granularity score using the proposed post-processing heuristic. The assessment of three distinct obfuscation strategies pointed out that Adaboost is able to detected all of them, but it had problems in differentiating among them, which could to be related to a poor feature selection.

Keywords: Plagiarism detection · Text alignment · AdaBoost

1 Introduction

Plagiarism could be defined as the act to copy and to take as yours the work of someone else [9]. Detecting plagiarism in a huge collection of documents, such as articles, thesis, laws, news and others artifacts is not manually viable. In practice, reviewers need to use automated systems that help them checking similarity in order to detect eventual citation lack during the process of analysis of the submitted works [2]. In this way, the plagiarism detection task naturally comes to a context of automated plagiarism detectors, a specialized software to this task.

In order to contribute to plagiarism detection, this work proposes and evaluates the adoption of AdaBoost as a tool for classifying suspicious text passages

The authors thank the financial support of the Pontifical Catholic University of Minas Gerais (PUC Minas), the National Council for Scientific and Technological Development (CNPq, grant 429144/2016-4) and the Foundation for Research Support of the State of Minas Gerais (FAPEMIG, grant APQ 01454-17).

as plagiarism or not, even when some obfuscation strategy is used. Additionally, a simple post-processing heuristic for improving results granularity is also presented. The contributions of this work are two-fold: (i) impact analysis of a boosting-based approach on plagiarism detection; and (ii) evaluation of a simple post-processing heuristic to reduce granularity of detection results.

The rest of this paper is organized as follows. Section 2 presents basic concepts and works related to plagiarism detection. In Sect. 3, our proposed approach is fully described, while in Sect. 4 experimental results are presented and analyzed. Finally, in Sect. 5, we draw some conclusions and point out possible future research directions.

2 Basic Concepts and Related Works

The task of plagiarism detection in text documents can be divided into two distinct approaches [6]: (i) external; and (ii) intrinsic. In external plagiarism detection, given a set of suspicious documents and a set of potential source documents, the task is to find all plagiarized passages in the suspicious documents and their corresponding source passages in the source documents. However, in an intrinsic approach, given the same set of suspicious documents, the task is to extract all plagiarized passages without comparing them to any potential source documents.

The external detection consists of two main steps: (i) source retrieval that queries the external world and selects source candidates; and (ii) text alignment which compares pairs of documents to find all the plagiarism occurrences between the suspicious and source documents. According to [6], the challenge in text alignment is to identify passages of text that have been obfuscated. At the end, additional steps (such as filtering) could be used to post-processing the results [6].

In this work, we used a boosting algorithm. Boosting is a method to combine a number of weak classifiers (possibly generated by the same algorithm) into a single classifier with higher performance. Thus, it is a way for a low performance classifier become a high performance one [7].

One of its most successful versions is Adaptative Boosting (AdaBoost) [3]. AdaBoost works by using resampling and reweighting of the data set. Its main features are the capacity of using data set reweighting in a way that samples with poor performance receive more attention of the next classifiers, and the use of resampling to improve the diversity of each classifier. At the end of AdaBoost iterations, many classifiers are generated and then, a single classifier can be simulated by a voting among all the classifiers created.

Since 2007, there is a series of scientific events and shared tasks on digital text forensics and stylometry, named Plagiarism analysis, Authorship identification, and Near-duplicate detection (PAN)[1]. In [8], the authors used TF-IDF to identify the words' frequency. And then, using this with some heuristics, they got to the top of PAN competitors ranking in plagiarism detection.

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

In [4], machine learning was used to train some classifiers to cope with plagiarism detection. A series of features were extracted from the words' occurrences on the text to train the classifiers that were inspired by the best methodologies among PAN competitors, such as the 2014 winner [8]. The author of [4] used *Decision Tree* and *Naive Bayes* classifiers; and also explores the adoption of *Random Forest*. Compared to top approaches in PAN, results obtained by learning methods proposed by [4] were very competitive, specially for *Decision Tree* and *Random Forest*.

Multiple types of n-grams were used to ensemble a system for plagiarism detection in [5]. Another work used semantic and syntactic features to feed a fuzzy logic based plagiarism detector [1].

3 Approach for Plagiarism Detection

3.1 Preprocessing and Seeding

Analogously to [4,8], for each document, tokenization, stemming, and removal of stop-words were applied to each sentence. And, after that, a Bag-of-Words (BoW) model was generated for each of those preprocessed sentences.

According to [8], the seeding step is used to construct a large set of plagiarism candidates called *seeds*. In our case, the seeds are generated by the mapping every sentence from a source document to all sentences belonging to a suspicious document. Therefore, after seeding a large number of seeds is obtained for each pair of suspicious and source documents.

3.2 Feature Extraction and Seed Classification

In order to apply a machine learning approach to seed classification, a set of features are needed to describe each seed. We have adopted the same 13 features used by [4], which are based on *cosine* and *dice* dissimilarities computed over the BoWs representing the suspicious and the source sentences for each seed. Table 1 shows the list of those features with a brief description.

Due to the generation process and even to the problem nature, most of obtained seeds does not represent a plagiarism case, causing a significant imbalance that can hinder a machine learning approach. To cope with that, we adopted a under-sampling strategy to rebalance class distribution.

Then, training data were used to generate a classifier which is evaluated using a distinct testing subset with several metrics. In this work, we investigate the adoption of boosting for plagiarism detection; choosing to use Adaboost because it has been successfully applied to several different tasks. We have also selected *decision tree* as its base classifier, since it presented a high performance in [4].

3.3 Post-processing

After seed classification, the detected cases of plagiarism are usually fragmented into sentences. To reduce this fragmentation, improving the granularity of plagiarism detection, several distinct strategies have been proposed [4,8].

Table 1. Features from [4] used to describe *seeds* (or plagiarism candidates).

Feature	Type	Description
Cosine	Float	BoW similarity measurement (frequency dependent)
Dice coefficient	Float	BoW similarity measurement (frequency invariant)
IsMaxCos and IsMaxDice	Boolean	TRUE if it is source fragment with the highest, respectively, Cos or Dice value for that suspicous fragment
MaxDiffCos and MaxDiffDice	Float	Difference between the current source fragment and the source fragment with the highest maximum similarity value of the same type for that suspicious fragment
MeanDiffCos and MeanDiffDice	Float	Difference between the current source fragment and the document mean maximum values of the same type for that suspicious fragment
MaxNeighbourCos and MaxNeighbourDice	Float	The highest value of the given similarity type for immediate suspicious fragment neighbors
VerticalMaxDistCos and VerticalMaxDistDice	Integer	Distance in fragments between the current source fragment and the fragment with the highest similarity value of the given type
SrcSuspLenRatio	Float	Length ratio between source and suspicious passages

In this work, we proposed a simple heuristic to join adjacent seeds. As shown in Fig. 1, seed union occurs in three steps: (i) seeds are sorted by their positions of occurrence in suspicious and source documents (see Fig. 1b); (ii) union of adjacent seeds with respect to the source document for the same suspicious fragments, *i.e.*, horizontal union (see Fig. 1c); and, finally, (iii) union of adjacent seeds with respect to the suspicious fragments in the same interval, *i.e.*, vertical union (see Fig. 1d).

4 Experimental Results

In order to evaluate the adoption of Adaboost for plagiarism detection, we used the PAN 2013 Plagiarism Corpus, named here PAN-PC-13 [6]. The corpus contains in total 3,653 suspicious documents and 4,774 source documents. The training set consists of 5,000 distinct cases: 3,000 of obfuscated plagiarism (i.e., 1,000 for each obfuscation strategy mentioned ahead), 1,000 of non-obfuscated plagiarism, and 1,000 without any plagiarism. The test set presents the same amount of document pairs with the same case distribution. Due to memory and computation power limitations, our experiments used only 20% of training document

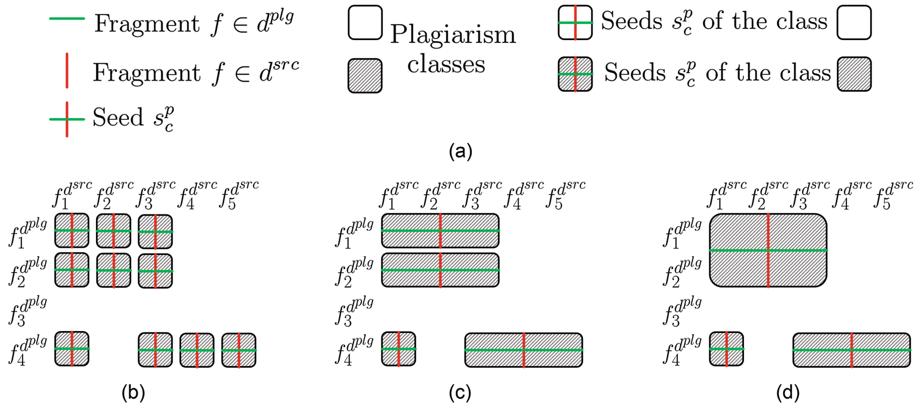
Fragment $f \in d^{plg}$
Fragment $f \in d^{src}$
Seed s_c^p
Plagiarism classes
Seeds s_c^p of the class
Seeds s_c^p of the class
(a)
$f_1^{d^{src}}$ $f_2^{d^{src}}$ $f_3^{d^{src}}$ $f_4^{d^{src}}$ $f_5^{d^{src}}$
$f_1^{d^{plg}}$ $f_2^{d^{plg}}$ $f_3^{d^{plg}}$ $f_4^{d^{plg}}$
(b)
(c)
(d)

Fig. 1. Example of seed union heuristic: (a) legend of elements used; (b) sorting of seeds by their positions; (c) horizontal union; and (d) vertical union.

pairs (1,036 pairs) with 656 suspicious documents and 894 source documents. Each document in the suspicious set is related with a source document. In the collection, there are three strategies of obfuscation: random obfuscation, translation obfuscation, and summary obfuscation [6].

As seed classification metrics, we adopted the traditional precision (P) and recall (R) measures – the first measures the correctness of the classification and the second measures how many of the target classes were classified as it; and also used F-measure (F) – the harmonic mean of precision and recall. All three metrics show the best results at one (1) and the worst at zero (0). To evaluate plagiarism detection, we use the following metrics defined in [6]: detection precision ($Prec$), detection recall (Rec), detection granularity ($Gran$) and overall detection score ($PlagDet$). Those are different from the traditional classification metrics because they check the results between detections generated and the actual cases at corpus, instead of assessing seeds classes prediction. With exception of $Gran$, these metrics show the best results at one (1) and the worst at zero (0); while $Gran$ shows the best results at one (1), but it will get worse if its value is greater than one (1). The results presented are the mean and the standard deviation obtained from 30 iterations of the experiments.

Along with Adaboost, we also trained and tested other two methods: Decision Tree and Random Forest – which presented competitive results in [4]. First, we performed a binary classification experiment: *Non plagiarism* $\times$ *Plagiarism*. Table 2 presents classification results for the binary experiment. One can easily see that AdaBoost and Random Forest are the ones with best performances (with minor differences in their scores). Table 3 presents plagiarism detection results for the same experiment. In this case, Random Forest had the best overall performance ($PlagDet$), while AdaBoost and Random Forest obtained the best $Gran$ value, but this can be explained by the decrease of $Prec$. Before seed union, all classifiers had obtained a great performance in terms of $Prec$ and Rec, despite the high values of *Gran*. But, for all three classifiers, the value of *Prec* decreased after seed union, showing that by joining seeds, the plagiarism cases become larger and errors (*false positives*) also become more meaningful. Another possibility is that small equivocated detections, diffused by the document, are causing errors, once they become more evident after seeds union.

Table 2. Classification results for binary experiment with 20% of PAN-PC-13.

Classifier	Class	*P*	*R*	*F*	Seeds in test set
Decision tree	*Non plagiarism*	.88 ± .00	.94 ± .00	.91 ± .00	25854
	Plagiarism	.93 ± .00	.87 ± .00	.90 ± .00	25855
Random forest	*Non plagiarism*	.89 ± .00	**.96 ± .00**	**.93 ± .00**	25854
	Plagiarism	**.96 ± .00**	.89 ± .00	**.92 ± .00**	25855
AdaBoost	*Non plagiarism*	**.92 ± .01**	.93 ± .00	.92 ± .00	25854
	Plagiarism	.93 ± .00	**.92 ± .01**	**.92 ± .00**	25855

Table 3. Detection results for binary experiment with 20% of PAN-PC-13.

Classifier	Post-processing	*PlagDet*	*Prec*	*Rec*	*Gran*	# Detections
Decision tree	No	.19 ± .00	.93 ± .00	.92 ± .00	29.02 ± .07	24022 ± 55
	Yes	.31 ± .00	.70 ± .00	**.93 ± .00**	4.95 ± .05	5380 ± 44
Random forest	No	.19 ± .00	**.95 ± .00**	.92 ± .00	29.72 ± .10	24029 ± 93
	Yes	**.37 ± .00**	.70 ± .01	**.93 ± .00**	**3.40 ± .08**	3681 ± 60
AdaBoost	No	.18 ± .00	.92 ± .00	**.93 ± .00**	30.68 ± .25	25606 ± 280
	Yes	.33 ± .00	.59 ± .02	**.93 ± .00**	**3.56 ± .12**	4613 ± 122

All tested classifiers presented some loss. So, in order to investigate more deeply this fact, we analyze the impact in results from different obfuscation strategies. Actually, we considered five distinct classes: "Direct" or non-obfuscated, "Random obfuscation", "Translation obfucation", "Summary obfuscation", and "Non plagiarism". The same three classifiers were tested without showing any distinguishable difference among their performance. So, as an analysis reference, we only present the results for Adaboost using 5% of the corpus.

Table 4 presents classification results for the multiclass experiment, while Table 5 shows plagiarism detection results for the same experiment. One should notice that "Direct" and "Non plagiarism" classes are the easiest ones to detect, while the other three obfuscated classes presented poor results with many scores below 0.5. This behavior could be related to the selected features to describe seeds. All selected features are strongly related to statistical information coded in BoW-models. But for obfuscated cases, they usually are neither very equal neither very different, making them fuzzy in some sense. This may hinder the classifier capacity of distinguishing different obfuscation strategies. One possible

Table 4. Classification results for multiclass experiment with 5% of PAN-PC-13.

Class	*P*	*R*	*F*	Seeds in test set
Non plagiarism	.82 ± .04	.83 ± .02	.83 ± .02	692
Direct	**.89 ± .01**	**.91 ± .02**	**.90 ± .01**	692
Random obfuscation	.37 ± .03	.42 ± .04	.39 ± .03	692
Summary obfuscation	.52 ± .05	.41 ± .09	.45 ± .07	692
Translation obfuscation	.46 ± .03	.46 ± .05	.46 ± .03	692

Table 5. Detection results for multiclass experiment with 5% of PAN-PC-13.

Class	Post-processing	*PlagDet*	*Prec*	*Rec*	*Gran*	# Detections
Direct	No	.20 ± .00	.88 ± .01	.68 ± .01	13.55 ± .25	711 ± 17
	Yes	**.29 ± .00**	.78 ± .02	.68 ± .01	4.60 ± .11	269 ± 8
Random obfuscation	No	.16 ± .01	.36 ± .03	.69 ± .05	6.91 ± .58	812 ± 91
	Yes	.19 ± .01	.30 ± .02	.70 ± .04	**3.46 ± .26**	497 ± 42
Summary obfuscation	No	.10 ± .00	.54 ± .04	.76 ± .09	78.96 ± 10.74	582 ± 73
	Yes	.10 ± .01	.41 ± .04	.74 ± .11	34.81 ± 4.86	342 ± 42
Translation obfuscation	No	.18 ± .01	.46 ± .02	.43 ± .04	4.82 ± .49	659 ± 80
	Yes	.22 ± .01	.51 ± .02	.46 ± .04	**3.47 ± .22**	444 ± 35
Plagiarism	No	.23 ± .00	**.95 ± .01**	.83 ± .00	14.10 ± .14	2764 ± 34
	Yes	.28 ± .00	.91 ± .01	**.84 ± .00**	7.64 ± .16	1552 ± 41

solution could be the use of semantic and syntactical attributes to enrich data giving to classifiers. The values for *Gran* were worse than in binary experiment. One possible explanation is that, as data space are more fragmented (by using five classes), the seeds became more diffused in the documents, rarely forming large segments (or forming areas with a great number of gaps). The proposed seed union heuristic is not able to deal with gaps. If they occur, seed union does not work well, keeping the granularity at a high level.

We have also analyzed the detection of any plagiarism case (regardless of obfuscation strategy) by the multiclass classifier. The results are presented under class name entry "Plagiarism" at Table 5; and they present good scores values for *Prec* and *Rec*. One possible reason for this is that many missed cases in multiclass classification would be actually a hit if obfuscation strategy were ignored. In this case, AdaBoost was able to correctly detect most of the plagiarism cases with a high value of *Prec*; nonetheless, it had problems in differentiating among them, which could to be related to a poor feature selection as mentioned before.

Finally, we have conducted an experiment to assess the maximum possible performance that could be achieved by the proposed seed union heuristic. In this test, we assumed that all seeds were correctly classified (by an ideal classifier).

Table 6 shows the detection results with a maximum *Prec* value of 0.98 without the use of seed union and 0.96 when it is used. The value of *Gran* shows the effect of seed union heuristic (reducing from 20.94 to 1.63). The distance from the ideal score (which is 1 for *Gran*) shows that our proposal for reducing fragmentation could also be improved in future.

Table 6. Detection performance for a ideal classifier with 5% of PAN-PC-13.

Post-processing	*PlagDet*	*Prec*	*Rec*	*Gran*	# Detections
No	0.21	**0.98**	0.93	20.94	4104
Yes	**0.68**	0.96	0.93	**1.63**	319

5 Conclusion

Detecting plagiarism in a huge collection of documents is not manually viable. This problem is more difficult, when some obfuscation strategy is used. In this work, we have proposed and evaluated the use of AdaBoost, together with a decision tree as base classifier, for classifying suspicious text passages as plagiarism or not. Additionally, a simple post-processing heuristic for improving results granularity is also presented.

Comparative analysis with other classification methods were conducted and experimental results have shown that Adaboost reached an F-measure of 0.92 and, together with Random Forest, the best granularity score using the proposed post-processing heuristic. The assessment of three distinct obfuscation strategies pointed out that Adaboost is able to detected all of them, but it had problems in differentiating among them, which could to be related to a poor feature selection.

As future research lines, we plan to explore: (i) the use of 5-fold distribution optimally balanced stratified cross-validation (DOB-SCV); (ii) the use of statistical tests to validate the classification results; (iii) the construction of multiple specialized ensembles models for the detection of obfuscated plagiarism cases and a comparison among these multiple models; (iv) the use of other base classifiers; and (v) the use of semantic and syntactic features to obtain a more discriminating representation.

References

1. Alzahrani, S.M., Salim, N., Palade, V.: Uncovering highly obfuscated plagiarism cases using fuzzy semantic-based similarity model. J. King Saud Univ. Comput. Inform. Sci. **27**(3), 248–268 (2015)
2. Blum, S.D.: My Word!: Plagiarism and College Culture. Cornell University Press, London (2009)
3. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. **55**(1), 119–139 (1997)
4. Kalleberg, R.B.: Towards detecting textual plagiarism using machine learning methods. Master's thesis, University of Agder (2015)

5. Palkovskii, Y., Belov, A.: Developing high-resolution universal multi-type N-gram plagiarism detector. In: Cappellato, L., Ferro, N., Halvey, M., Kraaij, W. (eds.) CLEF 2014 Evaluation Labs and Workshop, Sheffield, UK (2014). CEUR-WS.org
6. Potthast, M., et al.: 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, CLEF (2013)
7. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 3rd edn. Prentice Hall Press, Upper Saddle River (2009)
8. Sanchez-Perez, M., Sidorov, G., Gelbukh, A.: A winning approach to text alignment for text reuse detection at PAN 2014. In: CLEF 2014 Working Notes, pp. 1004–1011 (2014)
9. Stevenson, A. (ed.): Oxford Dictionary of English, 3rd edn. Oxford University Press, Hardcover (2010)