Author obfuscation using WordNet and language models Notebook for PAN at CLEF 2016

Muharram Mansoorizadeh, Taher Rahgooy, Mohammad Aminiyan, Mahdy Eskandari

Department of Computer engineering, Bu-Ali Sina University, Hamedan, Iran
mansoorm@basu.ac.ir, taher.rahgooy@gmail.com,
m.aminiyan@gmail.com, me16eskandari@hotmail.com

Abstract. As almost all the successful author identification approaches are based on the word frequencies, the most obvious way to obfuscate a text is to distort those frequencies. In this paper we chose a subset of the most frequent words for an author and replace each one with one of their synonyms. In order to select the best synonym, we considered two measures: similarity of the original word and the synonym, the difference between the scores (probabilities) that are assigned to the original and distorted sentences by a language model. By using similarity, we aim to select words that are similar to the original word semantically, and by using a language model we try to favor word usages that are common.

Keywords: author obfuscation, Brown corpus, WordNet, language model

1 Introduction

The author masking task of PAN 2016 is to paraphrase a given document so that its writing style does not match that of its original author, anymore. The problem consist of a set of documents written by a specific author and another document of the same author named “original”. The challenge is to use the information in the provided set of documents in order to obfuscate the “original” document. In order to response the challenge we use a five-stage iterative method which is described below.

2 Proposed approach

In the following we describe the steps we took in order to generate an obfuscated text from the original one.

2.1 Author word usage distribution

In the first stage, we used word frequencies of the training text for the author to estimate the word usage probability distribution by that author. For this purpose a maximum likelihood estimate of the word frequencies obtained using NLTK 3.0 toolkit [1]. We use this distribution to emphasis on words that used frequently by the author. We used top 200 most frequent words for this purpose.

2.2 Language Model

The understanding of natural languages is very complex. Words essentially combine in a non-random order but in some complex order. It is intuitively believed that the, language models can be learnt from the word and its neighboring words. Most language modeling tools use N-grams features to learn the probability for sequence of words (e.g. a sentence). We trained a 4-gram language model on Brown corpus [2] by KenLM toolkit [3] with default settings. KenLM uses modified Kneser-Ney smoothing [4] to adjust zero probabilities for non-presented words in the training data. We use this language model to score sentences generated by the obfuscation phase.

Let S = (w_1,w_2,…w_k,…, w_{n-1},w_n) be a sentence where w_k is a word that selected for replacement. Assume there is {v_1, v_2, …, v_m} candidates for replacement. So, in the second stage, we replace w_k with each candidate and score them using the score of generated sentence s_{v_k} in the language model:

$$score_{LM}(s, v_k) = score_{LM}(s_{v_k}) = \sum_{i=4}^{n} log(P(w_i|w_{i-1}w_{i-2}w_{i-3})) \quad (1)$$

By this approach we select words that are common in the corresponding context.

2.3 Synonym generation

In the third stage, for each selected word we generate a subset of word synonyms from WordNet [5] using NLTK toolkit. Then we scored similarity of each synonym with the original word by Wu and Palmer similarity score [6], which is a score denoting how similar two word senses are, based on the depth of the two senses in the taxonomy and that of their most specific ancestor node.

2.4 Post processing

The WordNet synonyms are lemmas of the words, so we need to find the suitable form of the generated words. Hence, in the fourth stage, we used the POS tag of the original word to find the proper form of the synonyms. The Pattern package from CLiPS toolkit [7] used to make this adjustment.

2.5 Word replacement

The state of the art authorship verification algorithms use word frequencies of the training set to determine if the “original” document is written by the corresponding author. Therefore we try to change the word frequencies of the “original” document in a way that they do not match with the training frequencies. The words that only used in the “original” document, but not in the training document, have no effect on the result of authorship verification, so we did not consider them for replacement.

So in the final stage, for each problem we concatenated all train documents and treated them as a single document. We obtained the word usage distribution of training document. Then iteratively a sentence from the "original" document selected for obfuscation. Firstly we tokenized the sentence and then part-of-speech tags of the sentence obtained using NLTK 3.0 toolkit. Then we generated a set of synonyms for each word that used at least once by the author in the training document, using aforementioned method. A subset of synonyms which were have high similarity scores to the original word is selected. From these candidates the word that has better language model score is selected. The final score for each word replacement in the sentence is:

$$score(s, v, k) = score_{LM}(s_{v_k}) * P(s[k]) \quad (2)$$

Where s is the original sentence, s_{v_k} is the original sentence that we replace its k-th word with v, and P(s[k]) is the probability of using k-th word of the sentence by the author. Based on this score we select best replacement. The ties are broken by order of the words in the sentence (the first one is selected). For each sentence we made at most one replacement.

3 Evaluation

In order to evaluate our work, the software run on TIRA platform [8] then a peer review has been performed on the results based on three different evaluation measures including safety, soundness and sensibility [9].
We call an obfuscation software

- Safe, if a forensic analysis does not reveal the original author of its obfuscated texts,
- Sound, if its obfuscated texts are textually entailed with their originals, and
- Sensible, if its obfuscated texts are inconspicuous.

3.1 Safety

To compare proposed algorithms effect on the state of the art author verification methods, we trained GLAD [10] tool on PAN2014 author verification dataset (The essays part). Then we run the trained model on results of each participant. Additionally we obtained the result for the original PAN 2016 author masking dataset. The GLAD accuracy on original dataset was 55.61 percent. Table 1 shows results obtained on each participant output.

Table 1. Results obtained on each participant output.

	Accuracy	Masking Performance
Participant A	50.24	49.76
Participant B	39.51	60.49
Participant C	55.12	44.88
Original Data	55.61	-

Where masking performance is the ratio of instances that classified incorrectly by the authorship verification algorithm and accuracy is the ratio of instances that classified correctly by the algorithm.
Results show that all approaches decrease the accuracy of the authorship verification algorithms, but not as much as expected. The B approach outperforms other two approaches while C approach almost have no effect on the accuracy.
To have a better understanding of the results we obtained four type of output for each approach:

- Positive-Masked: instances that classified correctly by GLAD and masked
- Positive-Unmasked: instances that classified correctly by GLAD but unmasked
- Negative-Masked: instances that classified incorrectly by GLAD and masked
- Negative-Unmasked: instances that classified incorrectly by GLAD and unmasked

The "Negative-Unmasked" type is interesting, because this type of error means that the obfuscation results actually help the authorship verification algorithm to classify that instance correctly.

Table 2. Fout types of results obtained on each participant output.

	Positive-masked	Positive-unmasked	Negative-masked	Negative-unmasked
Participant A	42	72	60	31
Participant B	64	50	60	31
Participant C	42	72	50	41

On the positive instances, the participants A and C performed identically, while participant B performs a lot better than the other tw. In other hand, participants A and B worked the same on negative instances where the participant C performs worse.

3.2 Soundness

Second measure which is evaluated by our human expert (in this case our team of authors) is soundness. For soundness, we take a representative random sample of text fragments and evaluate whether they are paraphrases of the original text. We choose a random number of problems and give them a score between 0-5. Score 5 refers to the

method which is paraphrases the original text most accurately and the score 0 refer to the worst paraphrase.
Soundness results in table 3 illustrates that, our proposed method (participant A) has a much better performance (the score equal to 4.86 in average) compared to the other participants, where the second place belong to participant C which has the score 3.93.

Table 3. Results of soundness evaluation

problem number	A	B	C
7	5	2	3.5
9	5	2	3.5
18	5	1	4
25	5	1.5	5
35	4.5	1	4
42	3.5	3	3.5
58	4	2	4.5
60	5	3	4
63	4	2	4
70	5	4	4
81	5	2	4
89	5	1	4
90	5	2	4
105	5	3	4
106	5	1	5
108	5	2	4
109	5	3	3
113	5	3	4
116	5	3	2
128	5	2	3
141	5	1	5
144	5	1	3
152	5	1	3
157	5	2	3
167	5	2	5
173	5	1	5
182	5	4	4
193	5	2	4
201	5	2	4
203	5	3	5
Average	**4.86**	**2.08**	**3.93**

3.3 Sensibility

The last evaluation measure is sensibility which is also measured by human expert (same as soundness). For sensibility, we take a representative random sample of text fragments and evaluate whether and how obvious it is that the text has been obfuscated. We choose random number of problems and give theme score between 0-5. Score 0 refer to method which is generate the most obvious obfuscated text and the score 5 refer to the best result.

Table 4. Results of properness evaluation

problem number	A	B	C
7	5	1	2
9	5	1.5	1
18	5	1	2
25	4.5	2	4
35	4	2	3
42	5	2	3.5
58	4	1.5	4
60	5	1	3
63	5	2	4
70	4	3	4.5
81	4	1	4
89	5	1	4
90	5	1	3
105	5	3	4
106	5	1	5
108	5	2	4
109	5	2	3
113	5	3	3
116	4	3	3
128	5	2	3
141	5	1	5
144	3	1	5
152	5	1	4
157	5	2	3
167	5	2	5
173	4	1	5
182	5	3	4
193	5	1	3
201	5	3	5
203	3	2	5
Average	**4.65**	**1.76**	**3.7**

Sensibility results in table 5 demonstrates that, our proposed method outperforms the other works (the score equal to 4.86 in average), where the second place belong to participants C which has the score 3.7 in average. It means our method generates phrases which are more similar to human generated texts.
Note that you can find the complete evaluation in the task overview paper [9].

4 Conclusion

Using the approach described above, we follow two major ideas: first, a document obfuscation should not made in a way that any human or non-human reader can easily figure out the changes and second, the method should remove high frequent words and phrases which is used by the author. In case of PAN 2016 task, based on given problem, we proposed simple but efficient method in order to change a document in a way that machine learning techniques cannot simply recognize the original author. Also, we thought smart and few exchanges in the document's words would not deform the document in form machine produced text do, so the human reviewer could be confused with original text. We changed at most one word from each sentence, this parameter can be tuned using parameter selection methods in order to reach to a good balance between text obfuscation and the amount of distortion in the text.
The evaluation results shows that while our approach have a competitive result for safety, it makes very small changes in the original text, which means it keeps the quality of the original text way better than other approaches. Thus it performs very better than the other two approaches from soundness and sensibility perspective.

5 References

1. Bird, S., Klein, E., & Loper, E. (2009). Natural language processing with Python. "O'Reilly Media, Inc.".
2. A Standard Corpus of Present-Day Edited American English, for use with Digital Computers (Brown). 1964, 1971, 1979. Compiled by W. N. Francis and H. Kučera. Brown University. Providence, Rhode Island.
3. Heafield, K. (2011, July). KenLM: Faster and smaller language model queries. In Proceedings of the Sixth Workshop on Statistical Machine Translation (pp. 187-197). Association for Computational Linguistics.
4. James, Frankie. "Modified kneser-ney smoothing of n-gram models." Research Institute for Advanced Computer Science, Tech. Rep. 00.07(2000).
5. Miller, George A. "WordNet: a lexical database for English." Communications of the ACM 38.11 (1995): 39-41.
6. Z. Wu and M. Palmer. "Verb semantics and lexical selection". In Proceedings of the 32nd Annual meeting of the Associations for Computational Linguistics, pp 133-138. 1994.
7. CLiPS Pattern. Computational Linguistics & Psycholinguistics Research Center; Available from: http://www.clips.ua.ac.be/pattern.

8. Potthast, M., Gollub, T., Rangel, F., Rosso, P., Stamatatos, E., Stein, B.: Improving the Reproducibility of PAN's Shared Tasks: Plagiarism Detection, Author Identification, and Author Profiling. In: Kanoulas, E., Lupu, M., Clough, P., Sanderson, M., Hall, M., Hanbury, A., Toms, E. (eds.) Information Access Evaluation meets Multilinguality, Multimodality, and Visualization. 5th International Conference of the CLEF Initiative (CLEF 14). pp. 268–299. Springer, Berlin Heidelberg New York (Sep 2014).
9. Potthast, M., Hagen, M., Stein, B.: Author Obfuscation: Attacking State-of-the-Art Authorship Verification Approaches. In: Working Notes Papers of the CLEF 2016 Evaluation Labs. CEUR Workshop Proceedings, CLEF and CEUR-WS.org (Sep 2016)
10. Hürlimann, Manuela, et al. "GLAD: Groningen Lightweight Authorship Detection." Working Notes Papers of the CLEF (2015).