Feature Vector Difference based Neural Network and Logistic Regression Models for Authorship Verification

Notebook for PAN at CLEF 2020

Janith Weerasinghe and Rachel Greenstadt

New York University
{janith, greenstadt}@nyu.edu

Abstract. This paper describes the approach we took to create a machine learning model for the PAN 2020 Authorship Verification Task. For each document pair, we extracted stylometric features from the documents and used the absolute difference between the feature vectors as input to our classifier. We created two models: a Logistic Regression Model trained on a small dataset, and a Neural Network based model trained on the large dataset. These models achieved AUCs of 0.939 and 0.953 on the small and large datasets, making them the second-best models on both datasets submitted to the shared task.

1 Introduction

This paper presents our approach for the Authorship Verification Shared Task at PAN 2020 [6]. The objective of this task was to create an approach that would be able to predict if two given documents were written by the same person. The dataset provided for this task was compiled by Bischoff et al. [4] and contains English documents from fanfiction.net. Each record in the dataset consists of two documents which may or may not be written by the same person and the fandom that each document was categorized under. The ground truth specifies the author identifiers for each document and the prediction target indicating if the two documents were written by the same person. The training dataset for the shared task was available in two sizes: a smaller dataset with $52,590$ records and a larger dataset with $275,486$ records, with each document containing about $21,000$ characters and 4800 tokens.

This paper is structures as follows: in Section 2 we will describe our approach and in Section 3 we will present our results of the shared task. Section 4 discusses our conclusions and future work.

2 Approach

This section describes the approach we took to build two models (trained on the smaller and larger datasets) for authorship verification. The pre-processing and feature extraction processes were identical for both models. We used a Linear Regression classifier for the smaller dataset and a Neural Network for the larger dataset. Our approach was implemented on Python with NLTK [3], Scikit Learn [9] and PyTorch [8] libraries and the source code is available at: `https://github.com/janithnw/pan2020_authorship_verification`.

2.1 Problem Statement

The PAN 2020 shared task was to predict if two given documents (D_i and D_j) were written by the same person. We modeled this as a binary classification problem, in which the input to our classifier is a feature vector encoding the two documents (X) and the target variable (Y) indicating whether or not the two documents were written by the same author.

Preprocessing Each document in the dataset was run through a series of pre-processing steps prior to feature extraction. We will use the following sentence as a running example in this section:

> "The Soviets had already been merciless, ruthless as the next army."

Tokenizer: We used the NLTK Treebank Word Tokenizer with the default parameters. The tokenized version of the document is stored to be used in the next pre-processing steps and to be used in feature extraction steps.

Part of Speech (POS) Tagging We trained a Brill Tagger[1] using the script provided by Jacob Perkins[2]. A Brill Tagger uses a combination of simpler taggers provided by NLTK to assign initial tags to a text and then applies a set of transformational rules to fix incorrect tags. We opted to use this method, which is slightly less accurate than NLTK's default Perceptron based POS-tagger, due to the Brill Tagger's significant performance gain. In our preliminary analysis, we realized that a significant amount of time was spent on the POS-tagging phase of our pipeline. The following would be the output of our POS-tagger for the example sentence above:

```
[('The', 'DT'), ('Soviets', 'NNPS'), ('had', 'VBD'),
 ('already', 'RB'), ('been', 'VBN'), ('merciless', 'NN'),
 (',', ','), ('ruthless', 'NN'), ('as', 'IN'), ('the', 'DT'),
 ('next', 'JJ'), ('army', 'NNP'), ('.', '.')]
```

[1] `https://www.nltk.org/_modules/nltk/tag/brill.html`
[2] `https://github.com/japerk/nltk-trainer`

Generating Parse Tree (POS Tag Chunking) We used NLTK's Regex Parser to parse POS-tags and generate a parse tree from documents. We designed regular expression rules that would identify Noun Phrases and Verb Phrases given a sequence of POS-tags. While we could have used a machine-learning-based parser, which would have been slightly more accurate, we opted to use the simpler regular-expression-based parser due to performance concerns. The following would be the output of our parser for the example sentence above:

```
(S
  (NP The/DT Soviets/NNPS)
  (VP had/VBD already/RB been/VBN)
  (NP merciless/NN)
  ,/,
  (NP ruthless/NN)
  as/IN
  (NP the/DT next/JJ army/NNP))
```

2.2 Features

This section lists the features that we extract from the preprocessed data. These features are commonly used in most previous stylometry work [13]. We used some features that are described in Writeprints feature set [1]. We also believed that the syntactic structure of sentences would provide valuable signals to the classifier. Following prior work [5, 7], we included POS-Tag n-grams and partial parses (or POS-Tag chunks) as part of our feature set. The use of parse trees to extract stylometric features, called syntactic dependency-based n-grams of POS tags, was introduced by Sidorov et al. [12]. We used a slightly different approach to encode parse tree features (described below) which captures how different noun and verb phrases are constructed.

Several of our features described below are computed in terms of TFIDF values. We used NLTK's `TFIDFVectorizer` to compute the TF-IDF vectors for the documents. We set the `min_df` parameter to be 0.1 in order to ignore tokens that have a document frequency less than 10%.

- **Character n-grams**: TF-IDF values for character n-grams, where $1 \geq n \geq 6$
- **POS-Tag n-grams**: TF-IDF value of POS-Tag tri grams.
- **Special Characters**: TF-IDF values for 31 pre-defined special characters.
- **Frequency of Function Words**: Frequencies of 179 stopwords defined in the NLTK corpus package.
- **Number of characters**: The total number of characters in the document.
- **Number of words**: The total number of tokens in the document.
- **Average number of characters per word**: The average number of characters per document.
- **Distribution of word-lengths (1-10)**: The fraction of tokens tokens of length l, wehre $1 \leq l \leq 10$
- **Vocab Richness**: The ratio of hapax-legomenon and dis-legomenon. (Divided by the number of tokens in the document to account for documents of varying lengths). Here, hapax-legomenon is the number of words that only occur once in the document and dis-legomenon is the number of words that occur twice in the document.

– **POS-Tag Chunks**: TF-IDF values for Tri-grams of POS-Tag chunks. Here, we consider the tokens at second level of our parse tree. For example, for the sentence above, the input to our vectorizer would be `['NP', 'VP', 'NP', ',', 'NP', 'IN', 'NP', '.']`.
– **NP and VP construction**: TF-IDF values of each noun phrase of verb phrase expansion. For the sentence above, these expansions are `['NP[DT NNPS]', 'VP[VBD RB VBN]', 'NP[NN]', 'NP[NN]', 'NP[DT JJ NNP]']`

The features for each document were scaled. We used the absolute difference of feature vectors of each document as input to our classifier. Specifically, we took the feature vector for the documents D_i and D_j to be X_i and X_j. The feature vector used by our classifier was $X = |X_i - X_j|$

2.3 Classifier

We computed the features for each document pair in the two datasets (smaller and larger) as described in the previous section. Each dataset was randomly divided into three sets: train (70%), validation (15%), and test(15%). The training set was used to train the feature vectorizers and the classifiers, the validation set was used for model selection and parameter tuning and the test set was used to measure performance before submitting our model to PAN 2020 organizers for final evaluation.

We trained a Logistic Regression classifier using the features from the smaller dataset. The validation dataset is used to tune model parameters. We used a Neural Network with hidden layer of size 100 for the larger dataset and the model that achieved the highest AUC on the validation set, over 100 epochs was selected as the final model.

3 Results

Once the final models were trained, we deployed these models to the TIRA evaluation system [11] provided by the PAN 2020 organizers where the models were evaluated on an unseen dataset. They were evaluated on 5 measures: area under the ROC curve (AUC), F1-score, c@1 (a variant of the F1-score, which rewards systems that leave difficult problems unanswered [10]), and F_0.5u (a measure that puts more emphasis on deciding same-author cases correctly [2]). Table 1 shows the results of our two models, released by the PAN 2020 organizers[3]. The runtime of the Logistic Regression model was 3 hours and 43 minutes, and the runtime of the Neural Network model was 2 hours and 19 minutes. Our model was the second-best performing model in the competition for both small and large datasets.

4 Discussion and Conclusion

In this paper we presented the approach we took in designing machine learning models for authorship verification. Our approach involves extracting stylometric features from a given document pair, taking the absolute difference (or the L_1 distance) of the feature

[3] `https://pan.webis.de/clef20/pan20-web/author-identification.html\#results`

Dataset / Model	AUC	C@1	F0.5U	F1-Score
Small, Logistic Regression	0.939	0.833	0.817	0.860
Large, Neural Network	0.953	0.880	0.882	0.891

Table 1. Results of PAN 2020 Shared Task

vectors of the document pair and using the resulting vector as input to a machine learning model. This approach allows us to use features that were used in authorship-attribution problems and use them in an authorship-verification setting. Most machine learning models that solve authorship attribution problems are author-specific, i.e., the models are trained on a known set of authors. Authorship verification problems– and our proposed solution–create a machine learning model that is generic, and thus applicable to any two given documents, even when a specific author is not known. While this particular problem set is closed-world, based on our preliminary results from other open-world datasets, we believe that our approach would generalize well for open-world scenarios.

As future work, we would like to optimize our model for the rest or the evaluation metrics. We believe it is possible to optimize for the c@1 score by taking classifier confidence into account. We would also like to perform a feature analysis of our models to see which features become important in determining if two documents are written by the same person.

5 Acknowledgements

We thank PAN2020 organizers for organizing the shared task and helping us through the submission process. We also thank the reviewers for their helpful comments and feedback. Our work was supported by the National Science Foundation under grant 1931005.

References

1. Abbasi, A., Chen, H.c.: Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. ACM Transactions on Information Systems **26**, 1–29 (01 2008). https://doi.org/10.1145/1344411.1344413
2. Bevendorff, J., Stein, B., Hagen, M., Potthast, M.: Generalizing unmasking for short texts. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 654–659. Association for Computational Linguistics, Minneapolis, Minnesota (Jun 2019). https://doi.org/10.18653/v1/N19-1068, `https://www.aclweb.org/anthology/N19-1068`
3. Bird, S., Klein, E., Loper, E.: Natural language processing with Python: analyzing text with the natural language toolkit. "O'Reilly Media, Inc." (2009)
4. Bischoff, S., Deckers, N., Schliebs, M., Thies, B., Hagen, M., Stamatatos, E., Stein, B., Potthast, M.: The Importance of Suppressing Domain Style in Authorship Analysis. CoRR **abs/2005.14714** (May 2020), `https://arxiv.org/abs/2005.14714`

5. Hirst, G., Feiguina, O.: Bigrams of syntactic labels for authorship discrimination of short texts. Literary and Linguistic Computing **22**(4), 405–417 (2007)
6. Kestemont, M., Manjavacas, E., Markov, I., Bevendorff, J., Wiegmann, M., Stamatatos, E., Potthast, M., Stein, B.: Overview of the Cross-Domain Authorship Verification Task at PAN 2020. In: Cappellato, L., Eickhoff, C., Ferro, N., Névéol, A. (eds.) CLEF 2020 Labs and Workshops, Notebook Papers. CEUR-WS.org (Sep 2020)
7. Luyckx, K., Daelemans, W.: Shallow text analysis and machine learning for authorship attribtion. LOT Occasional Series **4**, 149–160 (2005)
8. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019)
9. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
10. Peñas, A., Álvaro Rodrigo: A simple measure to assess non-response. In: ACL. pp. 1415–1424 (2011), http://www.aclweb.org/anthology/P11-1142
11. Potthast, M., Gollub, T., Wiegmann, M., Stein, B.: TIRA Integrated Research Architecture. In: Ferro, N., Peters, C. (eds.) Information Retrieval Evaluation in a Changing World. Springer (Sep 2019)
12. Sidorov, G., Velasquez, F., Stamatatos, E., Gelbukh, A., Chanona-Hernández, L.: Syntactic n-grams as machine learning features for natural language processing. Expert Systems with Applications **41**(3), 853 – 860 (2014). https://doi.org/https://doi.org/10.1016/j.eswa.2013.08.015, http://www.sciencedirect.com/science/article/pii/S0957417413006271, methods and Applications of Artificial and Computational Intelligence
13. Stamatatos, E.: A survey of modern authorship attribution methods. Journal of the American Society for information Science and Technology **60**(3), 538–556 (2009)