Sparse Coding in Authorship Attribution for Polish Tweets

Piotr Grzybowski
Wrocław University of Science and Technology
Wrocław, Poland
p.grzybowski2@gmail.com

Ewa Juralewicz
Wrocław University of Science and Technology
Wrocław, Poland
evajuralewicz@gmail.com

Maciej Piasecki
Wrocław University of Science and Technology
Wrocław, Poland
maciej.piasecki@pwr.edu.pl

Abstract

The study explores application of a simple Convolutional Neural Network for the problem of authorship attribution of tweets written in Polish. In our solution we use two-step compression of tweets using Byte Pair Encoding algorithm and vectorisation as an input to the distributional model generated for the large corpus of Polish tweets by word2vec algorithm. Our method achieves results comparable to the state-of-the-art approaches for the similar task on English tweets and expresses a very good performance in the classification of Polish tweets. We tested the proposed method in relation to the number of authors and tweets per author. We also juxtaposed results for authors with different topic backgrounds against each other.

1 Introduction

The problem of authorship attribution is one of the major areas of text classification. However, the issue is usually undertaken in the context of longer texts, such as book fragments, journal articles or emails. In recent years, mass media channels started playing a huge role in social life and made it possible to participate in global discussions, especially through social media. Twitter is one of the most influential social media platforms where anyone can share their opinion in form of a short text. Along with the growing influence of such platforms and limited user verification possibilities, importance of verifying the authorship of tweets and other short texts published on social media platforms has grown considerably. Such need is also motivated by moral responsibility of providing reliable media channels and discriminating propaganda messages.

Inspired by the results obtained in (Shrestha et al., 2017) as well as the simplicity of their method, we decided to verify its abilities in terms of classifying tweets written by selected Polish influencers. We treat the problem as a multiclass classification of texts. Our method differs from the original approach in terms of a chosen method for text encoding and next the way of obtaining its distributional vector representation as well as the scope of usage of data prepared in the processing pipeline, i.e. we use different data, unrelated to that used for classification, to train the distributional model.

2 Related Works

A big part of research done in terms of authorship attribution is relevant for big chunks of texts, where the sample of author's writing is relatively big (Gollub et al., 2013), (Frantzeskou et al., 2007), (Koppel et al., 2011). Recently, a lot of work has been done regarding authorship attribution of short texts, especially tweets due to their accessibility. The problem is challenging in comparison to the classification of longer texts as it is harder to maintain the classification accuracy along with the input pruning (Koppel and Winter, 2014). Some methods are based on stylistic features (Macleod and Grant, 2011) or word and character n-grams (Schwartz et al., 2013), (Sapkota et al., 2015).

Considering the fact that tweets may be of highly varying character with usage of multiple special characters, notorious misspellings and mixes of languages, character n-grams seem to be intuitively best-fit for the problem. Moreover, such approach appears to hold both topic-specific and morphology-specific information on the text (Koppel et al., 2011)

Proceedings of Recent Advances in Natural Language Processing, pages 409–417,
Varna, Bulgaria, Sep 2–4, 2019.
https://doi.org/10.26615/978-954-452-056-4_048

(Sapkota et al., 2015). There were a couple of attempts to classify tweets based on their character n-gram representations (Schwartz et al., 2013) (Sari et al., 2017) (Zhang et al., 2015) with different approaches to extracting a subset of meaningful characters (Plakias and Stamatatos, 2008) (Sapkota et al., 2015). In terms of a classifier used for such a task, CNNs have been recently widely explored and proved successful for text analysis (Sierra et al., 2017) (Ruder et al., 2016).

A combination of the two approaches appears in (Shrestha et al., 2017) where a text is classified based on a sequence of input characters. The method uses straightforward sequences of characters and their n-grams which are then embedded and processed in a CNN classifier. Another approach which is using a byte pair encoding algorithm for text encoding together with the application of the same CNN classifier, proves that the method is comparable to the state-of-the-art despite another level of text compression (Wang, 2018). The latter, however, uses a limited range of characters with at least punctuation and white space characters ignored. Additionally, in both works the embedding layer is trained simultaneously with the classifier, which makes it impossible to transfer the acquired knowledge to external data sets.

3 Contribution

In our work we extend the solution presented in (Shrestha et al., 2017) by using *SentencePiece* with *Byte-Pair Encoding* algorithms (Kudo and Richardson, 2018), without exclusion of any characters. Moreover, instead of using a data-specific text embedding trained simultaneously with the classifier, we teach a separate distributional language model on a corpus of Polish tweets. We also test robustness of our solution by alternating topics in the training and testing corpora, as well as using disjunctive time windows for texts in both corpora.

4 Data

For the needs of the development and testing of our solution we collected two separate data sets:

Tweet Corpus – 1 020 234 Polish tweets (including 95 435 564 characters in total) obtained through Twitter API[1] by applying Polish language filter (automatic Twitter filter that can produce some noise) and in addition tracking the top 100 words from the frequency list generated for the Polish language (Kazojć, 2009) in the search query,

Influencer Set – 138 486 tweets written by 28 Polish influencers who were chosen manually.

The *Tweet Corpus* consists mostly of tweets in Polish. We decided not to filter out tweets including fragments in foreign languages as it is a common practice for the Twitter users to write their content with injections of other languages. Data has been gathered using **twint** Python module (Zacharias and Poldi, 2018).

For the *Influencer Set* we selected the authors mainly due to their activity and the number of followers. In addition we pre-selected four categories of authors, namely: *journalists* (8 authors), *politicians* (6 authors), *publicists* (4 authors) and *computer gamers* (2 authors). The topic-related groups are extrated as one of the goals of the experiments was to verify whether the subject on which particular users tweet is a strong distinguishing factor which affects obtained results. As the activity of different authors is diversified, the *Influencer Set* is highly unbalanced with the number of tweets per author deviating from 314 to 18 204 tweets per account. This problem was mediated by subsampling during the experiments.

5 Text Representation

Tweets are very short texts and do not include many repeated occurrences of typical stylometry markers like functional words. We can observe prevalence of the information content over typical stylistic markers. Thus, we need to search for semantic elements reoccurring for a single author, as well as characteristic idiosyncrasies of his language, e.g. words or expressions.

Tweet mostly include many abbreviations, typos, or language errors that result in relatively high variability of the language and complexity of the statistical picture. Thus, we need

[1] https://twitter.com

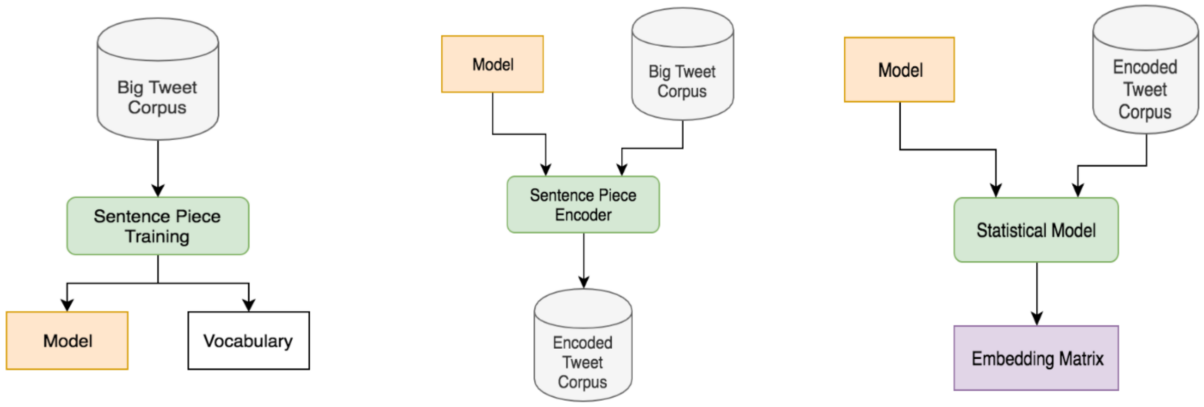

Figure 1: High-level structure of the processing pipeline used for obtaining text representation in our solution. From left to right: encoding model training, data encoding, embedding training.

to transform the original texts into a representation reducing this complexity. Very often, e.g. (Shrestha et al., 2017), tweets are represented by n-grams instead of words. However, the number of different n-grams is still very high. Instead of using n-grams, we processed the tweets with the help of the *Byte Pair Encoding* algorithm from *SentencePiece* library. *Byte Pair Encoding* (henceforth BPE) is a text compression method that constructs a tree-like structure of codes: recursively, two adjacent characters or symbols are encoded with a code represented by a single character that does not occur in the whole text. All punctuation and white characters are treated in the same way as other symbols. So there is no need for tokenisation. The algorithm starts from the most frequent pairs of characters (and next codes) and ends when all sequences are covered or it has reached the vocabulary size. Due to its recursive work the codes mostly represent not only bigrams but also higher-level n-grams, e.g. frequent words or even expressions. In all our experiments we used the BPE vocabulary size of: *4000 codes*. We tested vocabulary sizes from 1000 to 8000 codes and we did not notice improvement in performance for vocabulary size larger than 4000 codes.

BPE is often used for text compression, but our purpose was to make it a basis for a subword distributional semantics model. Texts encoded by BPE (i.e. tweets) were transformed into sequences of BPE codes separated by white spaces and delivered in such a form to the *Skip gram* algorithm from the *word2vec* library (Mikolov et al., 2013). As a result, every code receives its vector representation. The whole process is presented in Figure 1.

As codes represent different character sequences: from bi-grams, through sub-words up to even sub-expressions, we obtain a distributional semantics model which describes text units of varied granularity that reflect to some extent the statistical granularity of a corpus used for building the particular BPE model. A side effect is that vectors are also built for codes representing single letters that are rather meaningless, but this causes no harm to the overall properties of the model, as it will be visible.

The vector size in *Skip gram* was set to 300 elements. As the maximum length of a single tweet is 140 characters, so we made the representation of a single tweet to be a matrix of 140 code vectors[2]. In case BPE encoding for a tweet uses less than 140 codes (that often happens) the rest of the matrix is padded with a special null vector, i.e. not produced by *word2vec*.

In order to visualise the internal character of a BPE-encoded text, we present the histogram of initial tweet lengths in Figure 2, and contrast it with the histogram of the BPE-encoded tweet lengths in Figure 3. It can be noticed that the tweet lengths has become shorter, more evenly distributed with a slight dominance of the shorter representations. So,

[2]In the worst case of the weakest BPE mode codes correspond to single characters (symbols) in text, i.e. no more than 140 codes per a single tweet.

BPE-encoding results in a more packed text representation based sometimes on a few codes (representing sequences of letters).

Figure 4 presents the lengths of letter sequences represented by BPE codes obtained for the domain of politicians. The histograms are based on a subset of *Influencer Set*, containing tweets of 6 authors. When we compare it with the histogram of the whole domain in Figure 5 it can be observed that the code dictionary which is specific for the given domain contains slightly larger number of longer specialised codes. If we take a look into this domain specific dictionary we can find codes representing sometimes whole words that seem to be quite accidentally included into the dictionary. Thus, a dictionary built on the basis of a large corpus can be better suited to analysis of the authors' style that will be visible in Section 7 and especially in the results presented in Tables 4 and 3.

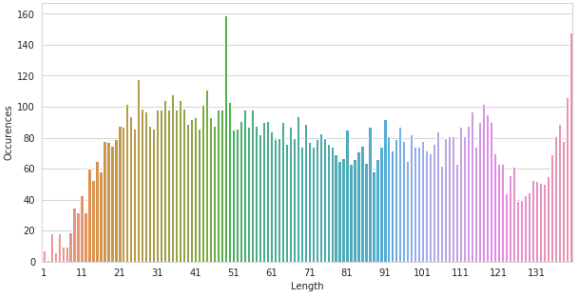

Figure 2: Histogram of tweet lengths for 28 635 tweets of politicians.

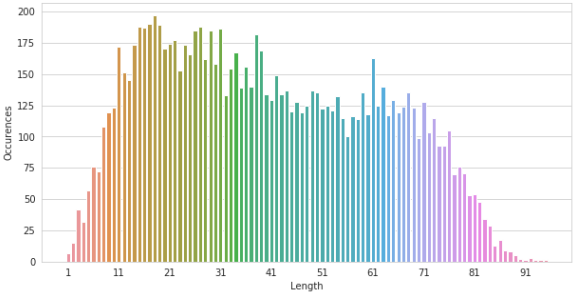

Figure 3: Histogram of BPE-encoded tweet lengths for 28 635 tweets of politicians.

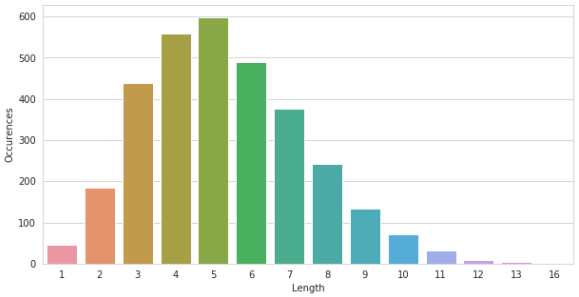

Figure 4: Histogram of BPE code lengths obtained by training the model on tweets of politicians (28 635 tweets).

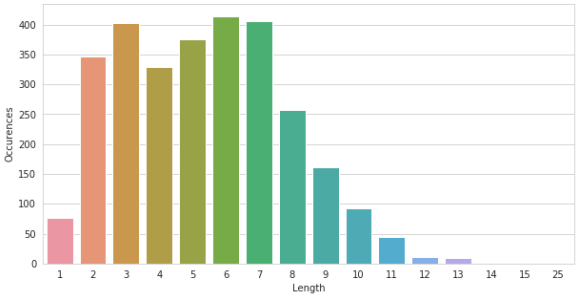

Figure 5: Histogram of BPE code lengths obtained by training the model on *Tweet Corpus* (1 020 234 tweets).

6 Classification Model

Our classification model follows the main lines *N-gram CNN* (Shrestha et al., 2017), but with BPE-based distributional representation in place of the original n-gram based one, i.e. instead of dividing the text into n-grams of characters directly from tweets, we split them into symbols obtained from the SentencePiece model. BPE-based model significantly limits the domain of possible symbols to the most common ones. It saves the vocabulary and the embedding matrix size. The neural network model architecture used in our research is visualised in Figure 6.

The architecture consists of an embedding layer and three parallel convolutional layers. Each convolutional layer contains 500 filters of sizes 3×3, 4×4 and 5×5 and is followed by max-pooling and dropout with 0.2 probability. The convolutional layers are merged and passed to a single dense layer with a 100 units and ReLU activation which is followed

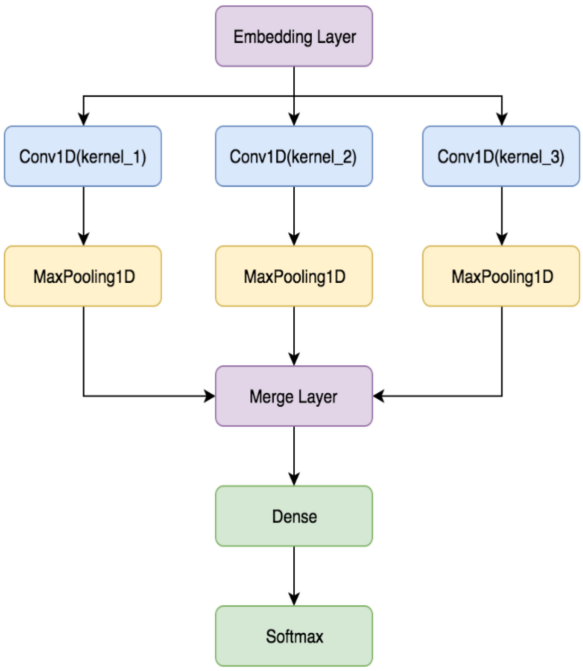

Figure 6: High-level structure of the N-gram CNN model used for author classification.

by dense layer with softmax activation.

Texts on the input to the embedding layer are represented by vectors pre-trained on the BPE encoded *Tweet Corpus*. During training the classifier, this layer was not completely frozen. Instead, it was adapted but with a significantly reduced learning rate as compared to the one used in the remaining layers. In this way, we manage to preserve high level generalisation obtained during training on the tweet corpus while slightly adapting to the training data domain.

The hyperparameters were slightly tuned from their initial state but their values are comparable to the ones used in the model of (Shrestha et al., 2017). We used Adam optimiser with 0.001 learning rate and categorical cross-entropy loss function. We trained the classifier with data batches of 4 samples in 5 epochs with L2 regularisation scaled by a 0.001 factor.

7 Experiments

7.1 Data Preprocessing

Before building the distributional semantics model and training the classifiers, we performed text preprocessing on tweets aimed at removing elements that seemed to be not relevant for the authors' styles and which could bias the classification process towards topic recognition. Thus, we removed all hashtags, mentions and URLs before constructing BPE encoding. The BPE models were generated from the preprocessed *Tweet Corpus* and selected subcorpora. Nonetheless, the very fact of using such extra-linguistic tweet elements, their placement and frequency might be important and relevant to authors' tweeting styles. Thus, we replaced all occurrences of extra-linguistic elements with symbols representing their types:

- *hashtags* were exchanged with '#' sign,
- *mentions* with the '@' sign,
- *URLs* with the 'ň' sign (not present in neither *Tweet Corpus* nor *Influencer Set*).

7.2 Multi-domain Authorship Attribution

In all experiments *Tweet Corpus* was used only to build a distributional semantics model and *Influencer Set* (not overlapping with the former) was used as training and testing data in a way following the k-fold cross validation scheme.

During the first experiment, we wanted to analyse the overall performance of the proposed approach on the whole *Influencer Set*. In order to balance the data set in relation to the number of tweets per author, we generated different subsets of the authors with random sub-sampling of tweets from more active authors. We wanted to investigate how many authors' styles our model can learn and how many tweets of each author are necessary to achieve satisfying results. The results are presented in Table 1.

7.3 Baseline Comparison

We compare our method against the baseline method of (Shrestha et al., 2017), using the same data set, i.e. the set presented in (Schwartz et al., 2013). This data set does not contain external data for language model training, so our approach of using a separate corpus for this task could not be applied. Thus in this experiment our solution differs only in the method for text encoding – the baseline

Tweets	Acc - 2	Acc - 4	Acc - 8	Acc - 12	Acc - 17	Acc - 23	Acc - 26	Acc - 28
16000	97.82 %	-	-	-	-	-	-	-
8000	97.41 %	90.89 %	-	-	-	-	-	-
4000	95.12 %	87.42 %	84.08 %	-	-	-	-	-
2000	93.25 %	84.11 %	81.19 %	72.44 %	71.34 %	-	-	-
1000	92.53 %	82.37 %	73.18 %	66.13 %	62.38 %	57.81 %	-	-
500	87.50 %	81.25 %	71.46 %	57.25 %	49.29 %	49.61 %	51.77 %	-
250	88.03 %	69.52 %	63.00 %	44.67 %	48.47 %	38.43 %	40.23 %	42.14 %
100	82.51 %	63.25 %	58.62 %	42.08 %	39.41 %	31.11 %	32.48 %	30.53 %

Table 1: Classification results for a varying number of authors and number of tweets. First column indicates the number of tweets per user used in experiment and the first row presents how many authors were classified in it. For instance, Acc-8, means the column under this cell presents accuracy results for 8-class classification (8 authors).

No. of tweets	**N-gram CNN**	**Ours**
50	**0.562**	0.528
100	**0.617**	0.609
200	**0.665**	0.657
500	0.724	**0.742**
1000	**0.761**	0.758

Table 2: Accuracy results for (Schwartz et al., 2013) data set, compared with results obtained in (Shrestha et al., 2017) for 50 authors.

method tokenises text into n-grams, while ours is using the BPE algorithm to encode tweets. For this data set we used the vocabulary size of 4000, with embedding length of 300, 128 batch sizes and 1.0 for the learning rate for Stochastic Gradient Descent in embedding training. For the classifier, we used Adam optimizer with 0.0001 learning rate, trained for 50 epochs with batches of 8 samples. Our results are averaged over 10-fold cross-validation. The baseline results are taken from (Shrestha et al., 2017), where we took the best achieved scores out of two studied architectures.

The comparison of classification accuracy for 50 authors and varying number of tweets in presented in Table 2.

As it is visible in Table 2, the baseline method is slightly better in majority of the experiment setups. However, the data set provided by (Schwartz et al., 2013) does not have an explicit train/test split or even a specified pool of 50 authors to perform classification on. Each fold has been randomly subsampled from a large pool of 7 026 authors. Moreover, for each author a specified number of tweets has been also randomly subsampled from a pool of 1 000 tweets.

The results presented in (Shrestha et al., 2017) are also not explicitly said to be averaged over folds. Considering the above mentioned preconditions, the results obtained cannot be used to unambiguously determine a superior method out of the two in a straightforward manner.

7.4 Cross-domain Encoding and Embedding

In the next group of experiments we investigated to what extent the model reflects particular domains of the authors (i.e. reacting first to the domain signal), and to what extent it represents the authors' style by themselves. Thus, we tried to classify authors from a specific domain by using the encoding-embedding model built on some other domain. Three configurations of both data sets were analysed:

1. the same data for building encoding model and tests,
2. data from another domain is used to construct encoding model,
3. the Polish *Tweet Corpus* is used for build-

Table 3: Accuracy score for authors of different domains (rows) with usage of encoding and embedding trained on different data (columns) with embedding layer active during training.

		Encoding and embedding training data					
	Author Domain	Sport	Social	Politics	Publicists	Gamers	Corpus
Classified	Sport	0.89	0.85	0.79	-	-	0.94
	Social	0.94	0.94	-	-	0.98	0.95
	Politics	0.86	-	0.89	0.79	-	0.90
	Publicists	0.87	-	0.86	0.90	-	0.88
	Gamers	-	0.96	-	-	0.97	0.99

ing the encoding mode (i.e. this is the baseline case).

For domains in which the number of authors is not the same, we subsample from the larger pool and average the results over possible combinations (e.g. Sport: 3 authors vs Gamers: 2 authors – we subsampled 3 times).

In addition, we run the experiments in two setups:

- with the embedding layer active during training a classifier
- and with embedding layer remaining frozen.

The results from the first scenario are shown in Table 3, while the effects of the second setting are presented in Table 4. The label "Corpus" means that the encoding model and the embeddings were constructed on the large *Tweet Corpus*, while the classifiers were trained on the given domain.

The lack of a value means that for the given pair we did not have enough data to build a balanced training-testing subset by subsampling to the size of the smaller domain.

8 Results

During the experiments on the whole set of authors, as expected, the results significantly decrease with the increasing number of authors taken into account and consequently the decreasing number of tweets per an author (due to the subsampling). While the results fall behind those presented in (Shrestha et al., 2017), they still surpass initial expectations. It is worth to take into consideration that the Polish language is more complex than English from the statistical point of view due to the rich morphology and a weekly constrained word order. However the latter was of minor importance as the model works mostly on the subword level and the distributional semantics model that does not depend on the word order (i.e. it is not sequential). The subword embeddings seem to be useful in decomposing Polish morphological forms into their natural components.

In addition, Polish tweets often include fragments written in English (of varied correctness) that also adds to the complexity of the problem and models. It is worth mentioning that in our experiments we used a significantly smaller amount of data than it was done in (Shrestha et al., 2017).

We initially suspected that the model would be biased by the authors' domains bias (i.e. topics of tweets or characteristic elements of the domain jargon). However, the result in the cross-domain model scenario, see Table 3 and Table 4, show a different picture. Firstly, in all cases the BPE-based embedding model built on the large corpus appeared to be superior to the domain-based model. Definitely, the difference in size of the corpora mattered in all cases in favour of *Tweet Corpus*. However, the size of the code dictionary was constant and equal to 4 000, i.e. it was quite small, and we can expect that the generalisation in the case of the big corpus was substantial. When the embedding layer got frozen in the second cross-domain experiment, all the results decreased, but only slightly. So, in the case of the limited code dictionary – providing a sparse and generalised picture of texts – the domain-focused tuning of the embedding layer appeared to not be very important.

The comparison with the method of

Table 4: Accuracy score for authors of different domains (rows) with usage of encoding and embedding trained on different data (columns) with embedding layer frozen during training.

		Encoding and embedding training data					
	Author Domain	Sport	Social	Politics	Publicists	Gamers	Corpus
Classified	Sport	0.90	0.84	0.82	-	-	0.95
	Social	0.86	0.94	-	-	0.97	0.96
	Politics	0.85	-	0.86	0.82	-	0.91
	Publicists	-	-	0.85	0.86	-	0.90
	Gamers	-	0.98	-	-	0.99	0.99

(Shrestha et al., 2017) in Table 2 showed that this reference method performs slightly better, however, the top results come from an ensemble of the two different models while our results are achieved by the single method. Moreover, our model is targeted on languages with rich inflection.

9 Conclusions and Further Research

The proposed method for the authorship attribution for Polish tweets expressed good performance for a large group of authors and large data set. It is based on the idea of an application of the Convolutional Neural Network proposed by (Shrestha et al., 2017), but it expands this approach with a distributional semantics model based on the prior application of BPE text encoding (i.e. embedding vectors are built for BPE codes, not words). As a result the proposed methods seems to be better suited for processing short texts in a highly inflectional language. It is worth to emphasise that the proposed approach is language independent, as the BPE model is driven by the statistical patterns in the training corpus.

The idea of an adaptive, coarse-grained distributional text representation for the needs of non-semantic classification seems to be attractive and opens several questions. There are various points in which the whole process could be improved, including but not being limited to:

- collecting a larger tweet corpus,
- improving language-based filtering of obtained tweets,
- investigating more closely the influence of the code dictionary size,
- comparing the proposed approach with other embedding learning methods,
- and optimise the classifier architecture.

While language filtering for using tweets in the Polish language exclusively seems to be a reasonable thing to do, it might not be perfectly adequate for the content such as tweets. We can observe a growing trend for users to post content with both languages present and the usage of such a practice may also be considered as a characteristic feature of their writing style. Language switch recognition in such a short text like tweets might be erroneous, but its tracking can improve the representation.

The question to what extent the method recognises an author's style, and to what extent this is only due to the correlation with some topics can be answered by the analysis of the confusion matrices between authors, distinctive features and stability of the recognition in time. Such an analysis could be combined with a strict filtering of tweets concerning the same events or topics and then verifying the classification performance. This is especially important due to the fact that Twitter data gets quickly irrelevant as topics change very fast.

Acknowledgments

This work is partially financed as part of the investment in the CLARIN-PL research infrastructure funded by the Polish Ministry of Science and Higher Education.

References

Georgia Frantzeskou, Efstathios Stamatatos, Stefanos Gritzalis, Carole E. Chaski, and

Blake Stephen Howald. 2007. Identifying authorship by byte-level n-grams: The source code author profile (scap) method. *IJDE* 6(1).

Tim Gollub, Martin Potthast, Anna Beyer, Matthias Busse, Francisco Rangel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. 2013. 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. Multilinguality, Multimodality, and Visualization*. Springer Berlin Heidelberg, Berlin, Heidelberg, pages 282–302.

Jerzy Kazojć. 2009. Otwarty słownik frekwencyjny leksemów v.06.2009. https://web.archive.org/web/20091116122442/ http://www.open-dictionaries.com/slownikfrleks.pdf.

Moshe Koppel, Jonathan Schler, and Shlomo Argamon. 2011. Authorship attribution in the wild. *Lang. Resour. Eval.* 45(1):83–94. https://doi.org/10.1007/s10579-009-9111-2.

Moshe Koppel and Yaron Winter. 2014. Determining if two documents are written by the same author. *Journal of the Association for Information Science and Technology* 65. https://doi.org/10.1002/asi.22954.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *CoRR* abs/1808.06226. http://arxiv.org/abs/1808.06226.

Nicci Macleod and Tim Grant. 2011. Whose tweet?: authorship analysis of micro-blogs and other short form messages.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *CoRR* abs/1310.4546. http://arxiv.org/abs/1310.4546.

Spyridon Plakias and Efstathios Stamatatos. 2008. Tensor space models for authorship identification. In John Darzentas, George A. Vouros, Spyros Vosinakis, and Argyris Arnellos, editors, *Artificial Intelligence: Theories, Models and Applications*. Springer Berlin Heidelberg, Berlin, Heidelberg, pages 239–249.

Sebastian Ruder, Parsa Ghaffari, and John G. Breslin. 2016. Character-level and multi-channel convolutional neural networks for large-scale authorship attribution. *CoRR* abs/1609.06686. http://arxiv.org/abs/1609.06686.

Upendra Sapkota, Steven Bethard, Manuel Montes, and Thamar Solorio. 2015. Not all character n-grams are created equal: A study in authorship attribution. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Denver, Colorado, pages 93–102. https://doi.org/10.3115/v1/N15-1010.

Yunita Sari, Andreas Vlachos, and Mark Stevenson. 2017. Continuous n-gram representations for authorship attribution. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Association for Computational Linguistics, Valencia, Spain, pages 267–273. https://www.aclweb.org/anthology/E17-2043.

Roy Schwartz, Oren Tsur, Ari Rappoport, and Moshe Koppel. 2013. Authorship attribution of micro-messages. In *EMNLP*. ACL, pages 1880–1891. http://aclweb.org/anthology//D/D13/D13-1193.pdf.

Prasha Shrestha, Sebastian Sierra, Fabio Gonzalez, Manuel Montes, Paolo Rosso, and Thamar Solorio. 2017. Convolutional neural networks for authorship attribution of short texts. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Association for Computational Linguistics, Valencia, Spain, pages 669–674. https://www.aclweb.org/anthology/E17-2106.

Sebastián Sierra, Manuel Montes y Gómez, Thamar Solorio, and Fabio A. González. 2017. Convolutional neural networks for author profiling in pan 2017. In *CLEF*.

Zhenduo Wang. 2018. Text embedding methods on different levels fortweets authorship attribution. http://www.d.umn.edu/ tpederse/Pubs/zhenduo-report.pdf.

Cody Zacharias and Francesco Poldi. 2018. Twint project. `https://github.com/twintproject/twint`.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *CoRR* abs/1509.01626. http://arxiv.org/abs/1509.01626.