Bots and Gender Profiling with Convolutional Hierarchical Recurrent Neural Network

Notebook for PAN at CLEF 2019

Juraj Petrik, Daniela Chuda

Slovak University of Technology in Bratislava, Slovakia
{juraj.petrik, daniela.chuda}@stuba.sk

Abstract. Paper describes approach leveraging deep learning principles in bots and gender profiling task at CLEF 2019 conference. Our approach is using hierarchical network for classification of tweets sequences. We achieved 90% accuracy of type profiling for English and 86.9% for Spanish language and 77.6% and 77.2% respectively accuracy in gender profiling.

1 Introduction

This paper describes our approach to bots and gender profiling task for PAN at CLEF 2019 [1] [3]. Our approach is based on our previous work dealing with source code authorship attribution [2]. This approach is based on newest natural language processing principles used in text classification and stylometry. Our solution was evaluated using TIRA evaluation service [4].

Bots are commonly used in bank or insurance sector as chat bots. These chat bots act as first level support for customers. They are able to help people with simple problems. Another positive example are weather forecasting bots or stock exchange information bots. They are effective way for sending information to multiple users (via Twitter, Facebook and Instagram for example).

However, another type of bots is used to spread misleading information, fake news for example. And we need to filter out this kind of information, because people tend to believe in such information as this information is spread across the whole internet and looks like valid fact.

1.1 Task Description

Aim of this task[1] is to determine if given Twitter feed is written by human or bot. In case given Twitter feed is written by human, our next task is determine if it is written by male or female. Also, this task is multilingual, it consists of two sub

[1] https://pan.webis.de/clef19/pan19-web/author-profiling.html

datasets – English and Spanish. Despite of language separation, creators of dataset do not guarantee language consistency for all tweets in feed.

Our performance is evaluated by average accuracy of each subtask (human vs bot and male vs female) of each language.

2 Related Work

In terms of stylometry, authorship attribution is application of linguistic style to written language, but also to music [9], which is defining writer's style as unique property for specific author - his fingerprint. Authorship profiling is part of stylometry, which is specifically focusing on determining author traits, such as age, gender or occupation.

In context of this paper we will focus on linguistic stylometry due to natural language character of Twitter feeds.

Problem of duplicate accounts on internet discussion forums was discussed in [6]. Duplicate accounts are created because of account ban, group accounts, and reputation boosting (sales). Authors of this publication trained one classifier for each account (discussion forum user) – this means that for N user accounts there were N trained classifiers. Advantage of this solution is that we can run these classifiers independently – parallelization is trivial. It is also clear from the paper, that accounts with small number of messages with short length are problematic. Another problem is intentional modification of writer's style by Anonymouth tool for example [7], but fortunately we do not have any suspicions that such a tool was used in task datasets.

Other paper [8] was trying to find out if the writer's style was intentionally modified. They used character, numerical and special characters, words and word function properties. Samples classification was done by support vector machines (SVM) in cooperation with sequence minimal optimization (SMO). Also, other classification methods were tested, such as k-NN, naive Bayes classifier, decision trees and logistic regression. However, SVM with SMO achieved significantly better results than other methods. Next, they evaluated information gain of properties for distinguishing imitated and obfuscated documents from original ones, what is similar problem as type profiling in this task (human vs bot).

3 Our Method

Our method is based on our previous work, which achieved state of the art results in source code authorship attribution [2]. We made several changes to the method, to be able to leverage nature of twitter messages, most importantly to deal with natural language opposed to source code. Important improvement in our approach is hierarchical layers arrangement to take advantage of sequence character of tweets in feed.

We have done experiments also with TF-IDF based approaches in combination with different classifiers. This approach was superior to our method. However, we used it as a good baseline for experiments [5].

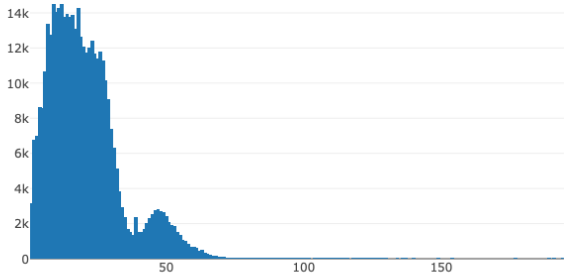

Figure 1. Histogram of tweets length (number of words)

3.1 Preprocessing

Training dataset consists of 4120 English Twitter feeds and 3000 Spanish Twitter feeds. Each feed consists of exactly 100 tweets. Maximum tweet length is 140 or 280 characters[2]. Samples are provided in XML files, one sample per file.

As stated above, data for this task consists of unprocessed twitter feeds. Our brief data analysis shown that majority of tweets contain relatively large number of emojis, quiet large number of typos, double, triple chars, punctuation or mixed language.

Twitter users are standardly utilizing Unicode set of emojis[3]. Working with special Unicode characters is not convenient and is easier working with their word description. In theory this should not be necessary, because we are using word embeddings. But our training corpus is relatively small, so this step is helping us to better train these embeddings - we are de facto extending tweets and got more tokens in dataset. You can see example of such a transformation in Table I.

Table 1. Example transformation table of Unicode emojis

Emoji	Code	Transformed emoji to text
😀	U+1F600	:grinning_face:
👍	U+1F44D	:thumbs_up:
👎	U+1F44E	:thumbs_down:
✊	U+270A	:raised_fist:
👊	U+1F44A	:oncoming_fist:

[2] https://developer.twitter.com/en/docs/basics/counting-characters.html
[3] http://www.unicode.org/emoji/charts/full-emoji-list.html

Table 2. Example tweet transformation of emojis to text and to lemmas

Before
@0rangelic @Roslinnovation You're doing way better than everyone here. 😉
After
@0rangelic @Roslinnovation -PRON- be do way well than everyone here . : winking_face :

Next step of our preprocessing pipeline is lemmatization. Lemmatization is process of extracting word lemma (word root). We can think of this as dimensionality reduction which will potentially easier generalization of our model (Table 2). Usually lemmatization is not needed when word embeddings are used, however our as stated, our corpus is small, so we are using all available method to make embeddings more stable and more domain specific.

Next step is tokenization and token encoding. We used standard Keras framework function for these two steps. Tokens were split by space characters and we filter out special characters such as braces, hash key, punctuation characters, etc. with combination of converting tokens to their lowercase representation. Tokens were encoded to integers based on their index in corpus dictionary.

Last step is zero-padding of inputs to fixed length (our model doesn't support variable length input sequence). We empirically chose sequence length of 60 tokens (words) based on histogram in Figure 1.

3.2 Classification

Our classifier is based on our previous work [2] – convolutional recurrent neural network. It consists of multiple layers – embedding, convolutional, recurrent and dense layers.

Convolutional neural networks are often used in image processing, where they achieved state of the art results in image recognition. They are gaining popularity also in natural language processing, because they act as feature extractors in texts too.

Embeddings are heavily used throughout variety of text processing problems. They are effectively encoding words into vector representation. Our embedding layer is randomly initialized (not pretrained) and trained on the fly - it should learn more specific domain specific embedding vectors this way.

Recurrent networks, especially Long-Short Term Memory units are used in state of the art models for text classification, emotions detection or speech recognition. They are good at sequence learning – and tweets are word sequences.

We were struggling with overfitting of the network, which we solve by adding dropouts. Dropout is randomly dropping connections between layers and therefore helping generalization of the model.

As stated above, tweets are word (character) sequences. We can say, that feeds are tweet sequences, therefore it could be beneficial to work with them as they are

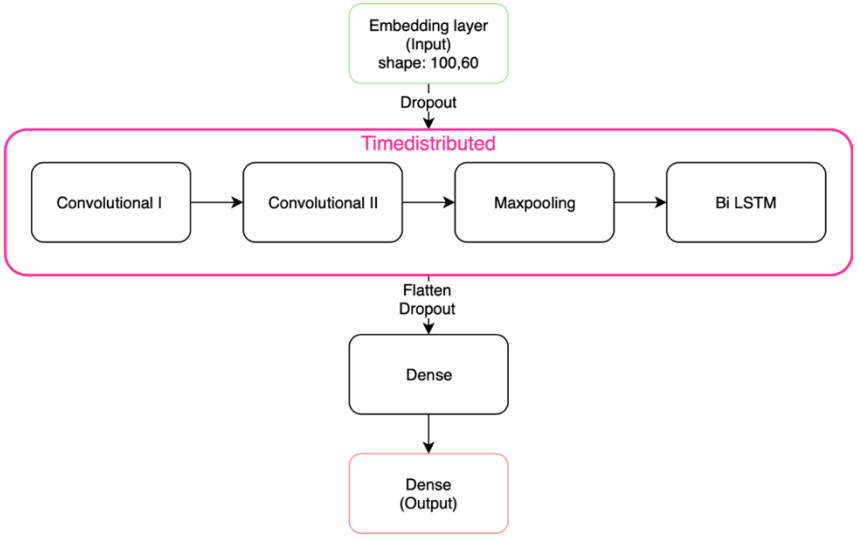

Figure 2. Our convolutional hierarchical recurrent network architecture

sequences. That's why we propose hierarchical network on top of tweets and dealing with them as with sequences, with all 100 tweets from sample "at once".

In text below are stated specific parameters of proposed and implemented neural network:

Layers parameters:
Embedding: vector length 30
Convolutional I: kernel size 2, number of filters 16, ReLU activation
Convolutional II kernel size 2, number of filters 16, ReLU activation
Max pooling: pooling size 2
Bidirectional LSTM: 16 units
Dense: 24 units
Dense (Output): 2 units, SoftMax activation
Dropout rate 0.5

Hyperparameters:
Batch size: 8
Epochs: 100
Early stopping: patience 5, validation loss monitoring
Loss: categorical crossentropy
Adam optimizer: learning rate 0.001, $beta_1$ 0.9, $beta_2$ 0.99

Table 3. Training dataset accuracy results (testing split)

	Type (acc)	**Gender* (acc)**
English	0.9831	0.9412
Spanish	0.9767	0.9221

4 Results

This chapter describes our testing results and task testing dataset results. Our testing results are average of multiple runs (10) of different dataset splits.

4.1 Our Testing Results

For our testing purposes we used 50/25/25 split for training, validation and testing fractions of data respectively. We are using accuracy metric, because classes are perfectly balanced, which means every class has exactly same number of samples.

Table 3 demonstrates our results on testing split (25% of training dataset). Our results were quite encouraging, although were significantly worse than results on organizers testing dataset (Table 4).

Also, our testing score (accuracy) was calculated differently than on task testing dataset. We were training exclusively on tweets posted by humans, so our gender testing accuracy is just from human samples. Gender task testing dataset was calculated from all samples (human and bot), this is main cause of big accuracy difference.

4.2 Task Testing Dataset Results

Our approach achieved roughly 90% accuracy in type recognition (human or bot) and 77.5% accuracy in gender recognition (male or female) for English Twitter feeds for task testing dataset. Spanish samples results were slightly worse – 86.9% accuracy for type recognition and 72.5% for gender recognition (Table 4).

It is evident that results on task testing dataset are significantly worse than results on our testing split. This is probably caused by insufficient generalization of our model, we suspect collected “vocabulary” is not large enough. Specifically speaking, vocabulary of our embedding layer is not large enough, which results in a lot out of vocabulary words and therefore next layers in our model don’t have enough information to make reliable decision. Testing dataset wasn’t published to the date of paper submission, so we are unable to make proper analysis in context of task testing dataset.

Table 4. Testing dataset results provided by task organizers (accuracy)

	Type (acc)	**Gender (acc)**
English	0.9008	0.7758
Spanish	0.8689	0.7250

5 Conclusions and Possible Upgrades

Our final task ranking is 21 from total of 55 contestants. Unfortunately, even two baseline methods (word and character n-grams outperformed our solution). Despite of final ranking, we must say that this our first appearance in such a competition was not total disaster – we rank in better half of solutions.

Unfortunately, because of time and computational constrains we were not able to realize and test all our ideas. Task results give us also some ideas, what we could done better.

First of all, tokenization step could be improved, for example there is lot of URLs in tweets, and we could get links from them and use information from sites such as topic or language of site. Additionally, we could use for example hypernyms in preprocessing to normalize texts.

Discussed above, our vocabulary was probably very limited (due to small training dataset). We could overcome this problem using pretrained English and Spanish embedding vectors or enrich dataset using Twitter real time API.

We used simple word level embedding layer, however other papers show, that using more sophisticated methods such as ELMo or character-based embedding have better results in topic modeling for example. Therefore, we can deduct usage of these methods could improve our results.

Acknowledgments

This work was partially supported by Human Information Behavior in the Digital Space, the Slovak Research and Development Agency under the contract No. APVV-15-0508, by the Slovak Research and Development Agency under the contract No. APVV-17-0267 - Automated Recognition of Antisocial Behaviour in Online Communities and by data space based on machine learning, the Scientific Grant Agency of the Slovak Republic, grant No. VG 1/0725/19.

Bibliography

1. Francisco Rangel, Paolo Rosso. Overview of the 7th Author Profiling Task at PAN 2019: Bots and Gender Profiling. In: Cappellato L., Ferro N., Müller H, Losada D. (Eds.) CLEF 2019 Labs and Workshops, Notebook Papers. CEUR Workshop Proceedings. CEUR-WS.org
2. Juraj Petrík and Daniela Chudá. 2018. Source code authorship approaches natural language processing. In Proceedings of the 19th International Conference on Computer Systems and

Technologies (CompSysTech'18), Boris Rachev and Angel Smrikarov (Eds.). ACM, New York, NY, USA, 58-61.
3. Daelemans, W., Kestemont, M., Manjavancas, E., Potthast, M., Rangel, F., Rosso, P., Specht, G., Stamatatos, E., Stein, B., Tschuggnall, M., Wiegmann, M., Zangerle, E.: Overview of PAN 2019: Author Profiling, Celebrity Profiling, Cross-domain Authorship Attribution and Style Change Detection. In: Crestani, F., Braschler, M., Savoy, J., Rauber, A., Müller, H., Losada, D., Heinatz, G., Cappellato, L., Ferro, N. (eds.) Proceedings of the Tenth International Conference of the CLEF Association (CLEF 2019). Springer (Sep 2019)
4. 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 - Lessons Learned from 20 Years of CLEF. Springer (2019)
5. Rangel, F., Rosso, P., Franco, M. A Low Dimensionality Representation for Language Variety Identification. In: Proceedings of the 17th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing'16), Springer-Verlag, LNCS (9624), pp. 156-169, 2018
6. S. Afroz, A. Caliskan-Islam, A. Stolerman, R. Greenstadt, and D. McCoy, "Doppelganger finder: Taking stylometry to the underground," Proc. - IEEE Symp. Secur. Priv., pp. 212–226, 2014.7.
7. A. W. E. Mcdonald, J. Ulman, M. Barrowclift, and R. Greenstadt, "Anonymouth Revamped: Getting Closer to Stylometric Anonymity," pp. 2–4, 2012.
8. S. Afroz, M. Brennan, and R. Greenstadt, "Detecting hoaxes, frauds, and deception in writing style online," Proc. - IEEE Symp. Secur. Priv., pp. 461–475, 2012.
9. E. Backer and P. Van Kranenburg, "On musical stylometry-a pattern recognition approach," Pattern Recognit. Lett., vol. 26, no. 3, pp. 299–309, 2005.