Detecting Conspiracy Tweets Using Support Vector Machines

Manfred Moosleitner, Benjamin Murauer, Günther Specht
Universität Innsbruck, Austria
manfred.moosleitner@uibk.ac.at,b.murauer@posteo.de,guenther.specht@uibk.ac.at

ABSTRACT

This paper summarizes the contribution of our team UIBK-DBIS-FAKENEWS to the task "FakeNews: Corona virus and 5G conspiracy" as part of MediaEval 2020. The goal for this task is to classify tweets as "5G corona virus conspiracy", "other conspiracy", or "non conspiracy", based on text analysis and based on the retweet graphs. We achieved our best results using a calibrated linear SVM with word and character n-grams for the text classification task and a non-calibrated linear SVM with graph statistics for the graph classification task.

1 INTRODUCTION

The main objective in the task is to distinguish tweets and classify them as either (1) contributing to a conspiracy suggesting that the 5G network technology caused the SARS-CoV-2 virus epidemic, (2) contributing to a different conspiracy, or (3) not contribute to a conspiracy. For the first subtask, this classification is based on the text content of the tweets. The second subtask focuses on the retweet and follower graph of the tweets. A detailed description and the results of the challenge can be found in [8], the collection of the data is described in [9].

In the remainder of this overview, we present our solutions for the two subtasks in the following Section 2, and discuss the results thereafter in Section 3.

2 METHODOLOGY

In both subtasks, the participants are allowed to submit 5 different solutions, whereas the first 2 solutions of each subtask are restricted to only use part of the information available. In the remaining 3 submissions, also external data points may be used.

2.1 Subtask 1: Twitter Messages

We extract character and word-based *n*-grams from the text of the tweets and use them as features for our classification models. This has been shown to be effective and versatile in different text classification task ranging from stance detection [2] to classifying hacked tweet accounts [4]. We tested different parameters in a grid search, the values of which are listed in Table 1.

Submissions 2 may include additional information, so we added all features that were included in the JSON structure, which correspond to the fields available from Twitter's API[1]. We transformed all textual features to tf/idf normalized frequencies of *n*-grams, as listed in Table 1, left the numeric features were left as-is, and mapped all categorical features to one-hot vectors.

[1]https://developer.twitter.com/en/docs/twitter-api

Copyright 2020 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
MediaEval'20, 14-15 December 2020, Online

We included two additional features that were not in the JSON files directly. Firstly, we crawled all URLs which were included in the messages and extracted the content of the sites `<title>` tag, hoping that it would contain a distinctive vocabulary. Secondly, we used the free OCR software *tesseract*[2] to find any text within the images that are included in the messages.

We tested linear support vector machines and extra random trees as classifiers, and also added the option of calibrating the SVM using Platt's method [7]. These classifiers have been well-studied and perform well in diverse text classification tasks [10], and can compete with neural-network-based approaches in many fields like spam detection [5].

2.2 Subtask 2: Retweet-Follower-Graphs

Standard graph statistics like the number of nodes or the graphs degrees are known to carry characteristics about the retweet graph to help in classification [1]. Also, algorithms like HITS [3] and PageRank [6] could produce discriminating features, as they were used on retweet graphs by Yang et al. in [11] to distinguish between tweets that are interesting only to a small group of people or a broader audience. Thus, we used the statistical networking Python package NetworkX [3] to extract statistical figures describing the retweet-follower-graphs. For the first run of the second subtask, we calculate *order*, *size*, *degree*, *indegree*, *outdegree*, *number of connected components*, *density*, *transitivity*, *pagerank*, *HITS (hubs, authorites)*, *number of partitions*, *planarity*, and *number of cycles*, and combined them into a single feature vector.

Some of the functions in NetworkX to calculate the graph statistics return lists of variable length, as their number depends on the number of nodes and edges. To create fixed-length feature vectors, we computed arithmetic mean, standard deviation, and the five-number summary of the values in the individual lists, and used these as features. For the second run in subtask 2, we additionally used the data from the nodes files, from which we calculated min, max, mean, and standard deviation of the number of friends and followers, and added these to the feature vectors calculated for the first run.

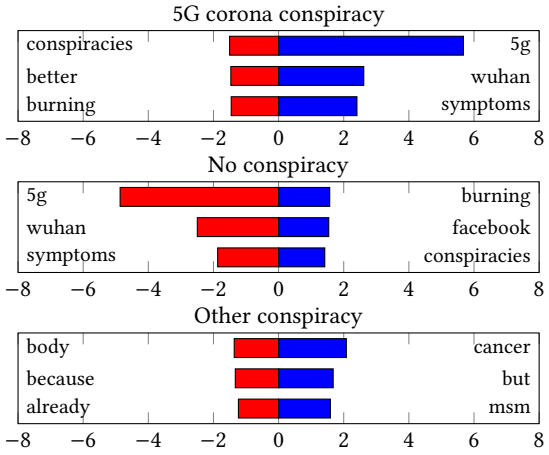
[2]https://tesseract-ocr.github.io/
[3]https://networkx.org/

Table 1: Hyperparameters tested in grid search.

Parameter	Tested values
Word & character *n*-gram size1	[1,2,3,4]
SVM: C	[0.1, 1, 10]
Extra Trees: number of trees	[1, 2, 3, 4] $\times 10^3$
Poly. degree	[2, 3]
Poly. include bias	[True, False]
KNN: number of neighbors	[3, 4, 5, 10, 20, 50]

CEUR-WS.org/Vol-2882/paper10.pdf

Figure 1: Top 3 positive and negative SVM coefficients for each class after fitting the message bodies of the training data.



Since we extracted significantly fewer features in the second subtask, we added polynomial feature generation, and added a gaussian naïve Bayes classifier and a K-nearest neighbor to the models from the first subtask. Both are well-studied algorithms and we were interested in how well they would perform for this task. We tested several parameters in a grid search, which are displayed in Table 1.

3 RESULTS AND DISCUSSION

After preliminary experiments for both subtasks, we selected the setup with the highest MCC score in a 10-fold cross-validation setup as the model that predicts our submission results for each subtask.

3.1 Subtask 1

The scores displayed in Table 2a show that the SVM model clearly outperforms the extra random trees approach in the first subtask. Thereby, calibrating the SVM increased the performance slightly.

Interestingly, the performance of the classifiers dropped when taking more features into account for the second submission. This indicates that either too many features are extracted from the text, or that the additional meta-information was not expressive to the problem. Nevertheless, we submitted the two results in this state, being aware that we could have possibly increased the performance of the second submission by ignoring the meta-features. The evaluation results, on the other hand, don't display a performance decrease between the two submissions, where both runs result in a score of 0.440 and 0.441, respectively. As shown in Table 3, the best results were obtained by combining word unigrams and character-3- and -4-grams and a strict regulation parameter of C=0.1.

Using a linear SVM as a model allows an easy interpretation of the importance of words by looking at the respective coefficients. For each output class, Figure 1 shows the terms with the three highest and lowest coefficients. The high value for the term *5g* suggests that not many topics within the *other conspiracies* are discussing the telecommunication standard. This relationship could be experimented with in more detail using topic modeling.

Table 2: Evaluation results measured with Matthew's correlation coefficient.

Phase	Model	Run 1	Run 2
Training	Linear SVM (calibrated)	**0.432**	**0.412**
	Linear SVM	0.428	0.404
	Extra Random Trees	0.274	0.253
Evaluation	Linear SVM (calibrated)	0.440	0.441

(a) Results of Subtask 1

Phase	Model	Run 1	Run 2
Training	Linear SVM (calibrated)	0.003	0.054
	Linear SVM	**0.127**	**0.197**
	KNN	0.118	0.135
	Extra Random Trees	0.089	0.091
	Gaussian Naive Bayes	0.092	0.101
Evaluation	Linear SVM	0.090	0.092

(b) Results of Subtask 2

Table 3: Best parameters for the four submissions.

Subm.	Parameters
Text 1	word-1-grams + character-3+4-grams, calibrated SVM, C=0.1
Text 2	word-1-grams + character-3+4-grams, calibrated SVM, C=0.1
Graph 1	linear SVM, C=10, Poly. deg=2, Poly. include bias = True
Graph 2	linear SVM, C=10, Poly. deg=3, Poly. include bias = False

3.2 Subtask 2

Similar to subtask 1, we used grid search to find the best performing classifier and parameters. The scores of the classifiers were rather similar, with the linear SVM producing the best score with the parameters C=10. While using polynomial features at all increased the result in both submissions by 0.05, whereas the parameters (degree=[2,3], include bias=[true, false]) did not have a great influence (< 0.01 MCC). as shown in Table 3. The results in training and evaluation approaches for subtask 2 were quite low, as displayed in Table 2b. Interestingly, our MCC validation scores for subtask 2 were lower than the training scores, which is in contrast to the scores of subtask 1, where the validation scores were slightly better than our training scores.

4 CONCLUSION

Our simple text-based approaches were able to classify the tweets reliably, and the coefficients of the model give insights into the most important terms. We suggest that more preprocessing might further improve these results.

The simple graph statistics, on the other hand, were not expressive enough for this task. Here, incorporating more metadata like the time between the retweets might improve the classification results.

REFERENCES

[1] David R Bild, Yue Liu, Robert P Dick, Z Morley Mao, and Dan S Wallach. Aggregate characterization of user behavior in twitter and analysis of the retweet graph. *ACM Transactions on Internet Technology (TOIT)*, 15(1):1–24, 2015.

[2] Peter Bourgonje, Julian Moreno Schneider, and Georg Rehm. From clickbait to fake news detection: an approach based on detecting the stance of headlines to articles. In *Proceedings of the 2017 EMNLP Workshop: Natural Language Processing meets Journalism*, pages 84–89, 2017.

[3] Jon M Kleinberg. Hubs, authorities, and communities. *ACM computing surveys (CSUR)*, 31(4es):5–es, 1999.

[4] Benjamin Murauer, Eva Zangerle, and Günther Specht. A peer-based approach on analyzing hacked twitter accounts. In *Proceedings of the 50th Hawaii International Conference on System Sciences*, 2017.

[5] N. L. Octaviani, E. Hari Rachmawanto, C. A. Sari, and D. Rosal Ignatius Moses Setiadi. Comparison of multinomial naïve bayes classifier, support vector machine, and recurrent neural network to classify email spams. In *2020 International Seminar on Application for Technology of Information and Communication (iSemantic)*, pages 17–21, 2020.

[6] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[7] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advanced Large Margin Classifiers*, 10, June 2000.

[8] Konstantin Pogorelov, Daniel Thilo Schroeder, Luk Burchard, Johannes Moe, Stefan Brenner, Petra Filkukova, and Johannes Langguth. Fakenews: Corona virus and 5g conspiracy task at mediaeval 2020. In *MediaEval 2020 Workshop*, 2020.

[9] Daniel Thilo Schroeder, Konstantin Pogorelov, and Johannes Langguth. Fact: a framework for analysis and capture of twitter graphs. In *2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS)*, pages 134–141. IEEE, 2019.

[10] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *Journal of machine learning research*, 2(Nov):45–66, 2001.

[11] Min-Chul Yang, Jung-Tae Lee, Seung-Wook Lee, and Hae-Chang Rim. Finding interesting posts in twitter based on retweet graph analysis. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 1073–1074, 2012.