Detection of COVID-19-Related Conspiracy Theories in Tweets using Transformer-Based Models and Node Embedding Techniques

Youri Peskine[1,*], Paolo Papotti[1] and Raphaël Troncy[1]

[1]*EURECOM, France*

Abstract

With the amount of information shared on the web increasing on a daily basis, we are prone to face more misinformation online. This is especially true on social media websites, where users have good amount of freedom to share their opinion. During the COVID-19 pandemic, numerous conspiracy theories were shared on Twitter. In this "FakeNews Detection" task, the goal is to detect COVID-19-related conspiracy theories using tweet text and user interaction graph. We tackled this challenge using Transformer-based models (CT-BERT) and node embedding techniques (node2vec) with classification objective models. Our best model obtains a MCC score of 0.719 on the test data.

1. Introduction

This year's "FakeNews Detection" task aims at detecting 9 named conspiracy theories in tweets, as well as classifying misinformation spreaders in a user interaction graph. The full task description is detailed in [1].

The first task of this challenge is a multi-label multi-classification problem in the tweet textual content. This type of problem has been studied extensively in the natural language processing (NLP) literature. Some popular baseline approaches to tackle this problem include statistical techniques, such as TF-IDF [2], or transformer-based models, such as BERT [3]. The second task is a node classification problem in a graph. This kind of task can be tackled with graph neural network based approach, such as GraphConv [4] (GCN), or approach that generates node embeddings, such as node2vec [5]. The third task is also a multi-label multi-classification problem, with both textual content from the tweet and information from the graph of user interaction. Our code is available on Github.[1]

2. Related Work

Transformer-based [6] models have achieved state-of-the-art performance for various NLP tasks [7]. These models are pre-trained on large corpus of text, and can be fine-tuned for a specific task on a smaller corpus. An example is COVID-Twitter-BERT (CT-BERT) [8], which is a pre-trained model on large corpus of Twitter data on the topic of COVID-19. This makes it suited for tasks 1 and 3 of this challenge. This year's task 1 is very similar to last year's task

MediaEval'22: Multimedia Evaluation Workshop, January 13–15, 2023, Bergen, Norway and Online

[*]Corresponding author.

youri.peskine@eurecom.fr (Y. Peskine); paolo.papotti@eurecom.fr (P. Papotti); raphael.troncy@eurecom.fr (R. Troncy)

[1]https://github.com/D2KLab/mediaeval-fakenews

3 [9], in which we participated [10]. In our previous experience, the CT-BERT model was the most performing one.

The second task requires methods that leverage graph data. Indeed, the provided data is composed of a graph of interaction between Twitter users, as well as some information about the user itself. More information about the data can be found in the task description paper [1]. This task of node classification can be tackled using node embedding techniques from sequence-based models (e.g. node2vec) or GNNs (e.g. GCN) [11]. Sequence-based models learn the embeddings of a node by using the structure of the graph and the neighbors of a node, without capturing any information about the node features. The node classification can then be done with different models, from the learned node embedding, using traditional classifier approaches such as Multi Layer Perceptron (MLP) or Random Forest (RF). The GNN-based approach optimize both the embedding and the classification task at the same time. It utilizes the structure of the graph, as well as the node features.

3. Approach

In order to tackle this challenge, we studied text-classification transformer-models for tasks 1 and 3, and node-classification models for tasks 2 and 3. Our approach leverages multiple CT-BERT models for text-classification and node2vec in combination with simple classifiers (MLP, RF) for node-classification. We also experimented with GNN without much success and we do not report these results. In all experiments, we split the data into 5 stratified cross-validation sets.

3.1. Text Classification

First, we used some basic pre-processing on the text data. We replaced all emojis with their textual meaning using the *emoji* Python library.[2] We also removed the hashtag ('#' character) from the tweets. Next, we approached task 1 as a multi-label 3-way classification problem. We trained 5 CT-BERT models, one for each cross-validation fold, using a custom loss function. The last layer of our models has 27 dimensions, three for each of the 9 conspiracy theories (discuss, support, not related). We build 9 different Cross-Entropy losses, each measuring the performance of the model at detecting one conspiracy theory. These Cross-Entropy losses are weighted independently, proportionally to the inverse of the frequency of the respective class in the training data. Then, the final loss is the unweighted sum of the 9 different losses.

3.2. Node Classification

We used a node2vec model to generate node embeddings, and then used standard machine-learning classifiers to perform the node classification.

We first build the graph from the user-interaction data, using the *networkx* Python library [12].[3] This graph is composed of around 1.7 Million nodes (representing the Twitter users) and 270 Million directed edges (representing the interaction between the users). We run the *node2vec* algorithm on that graph, using *nodevectors* Python library. [4] We generate 10 random walks per node, of length 40, with the return parameter set to p=1 and the in-out parameter set to q=1/2. The dimension of the embeddings is 32.

[2]https://github.com/carpedm20/emoji/
[3]https://github.com/networkx/networkx
[4]https://github.com/VHRanger/nodevectors

Table 1
MCC results for each run on the test set

	Run	Model	Test MCC
Task 1	001	CT-BERT Ensembling	**0.710**
	002	CT-BERT	0.685
	003	CT-BERT Ensembling+'CD'	0.657
Task 2	101	node2vec+MLP Ensembling	0.327
	102	node2vec+MLP	**0.355**
	103	node2vec+RF Ensembling	0.253
	104	node2vec+MLP Ensembling+'CD'	0.295
	105	node2vec+MLP+RF Ensembling	0.259
	106	node2vec+MLP Ensembling	0.327
Task 3	201	(task 1) CT-BERT Ensembling	**0.719**
	202	(task 1) CT-BERT	0.676
	203	(task 1) CT-BERT Ensembling+'CD'	0.663
	204	MLP Fusion Ensembling	0.690
	205	MLP Fusion	0.676

Next, we train some popular machine-learning classifier algorithms available on the *scikit-learn* Python library [13][5] to perform node classification. Those algorithms take as input the 32-dimension vector from the node2vec model and perform a binary-classification objective. Random Forest (RF) classifier obtained the best results. We also trained a MLP classification head as well.[6]

3.3. Tweet Classification Using Both Text and Graph Data

We used both graph and textual data for the third task of the challenge. We trained a classifier from the concatenation of both text and graph features, without any form of feature space normalization. Those graph features are the 32-dimension vector from the node embeddings, and the textual features are the 27-dimension vector output of the task 1 model. The training loss is similar to the one described in Section 3.1.

4. Results and Analysis

In this section, we first describe each run, and then analyse the main takeaways from the results. We present our results for this challenge in Table 1.

4.1. Runs description

For the first task, we performed an ensembling of the 5 models trained on each fold of the cross validation split (run 001). This ensembling was done with majority voting. In case of a draw between two classes, 1 would have priority over 2 and 3, and 3 over 2. This order follows the proportion of samples we have in the data. Run 002 is the single best model of the 5 models. Run 003 is the same ensembling as run 001 but with the 'cannot determine' labeling if there is less than 4 models out of 5 in agreement. The ensembling (run 001) obtained the best results.

The second task was more challenging and results are lower compared to task 1. We propose multiple ensembling methods from the models trained on each fold of the cross-validation split.

[5]List of tested classifiers: KNeighborsClassifier, GaussianProcessClassifier, DecisionTreeClassifier, RandomForest-Classifier, AdaBoostClassifier, GaussianNB, QuadraticDiscriminantAnalysis and GradientBoostingClassifier
[6]The hyper-parameters for the MLP model are: layers of 32-16-8-1, dropout p=0.1, ReLU activation function

We trained 5 MLP models and 5 RF models. Run 101, 103, and 105 are ensembling of those models. Run 106 is the same as run 101 with the 'cannot determine' labeling logic as in run 003. Run 102 is the best single MLP model from run 101. This run obtained the best results, out-performing the ensembling (run 101). We also submitted our run 101 as a sixth run (106) because we did not use any pre-trained model.

For the last task, we first propose the same models as the first task, without using the graph data. Runs 201, 202 and 203 are the same as runs 001, 002 and 003. Results are slightly different because the test data is different from task 1 and 3. Then, we also propose models using both text and graph data, using the approach described in Section 3.3. Run 204 is an ensembling of the MLP models trained on each fold of the cross validation split and run 205 is the single best MLP model. The best performing model for this third task is the ensembling of CT-BERT models, only on text data (same as run 001).

4.2. Main takeaways

A first takeaway from these results is that the CT-BERT model is suited for text classification tasks and obtains very good results, even if slightly lower than in 2021 [10]. The ensembling of models trained on different cross-validation fold improves the results with textual models (run 001, 201 and 204), but not for the graph-based approach (run 101, 103 and 105). Also, the 'cannot determine' labeling did not improve the results over the normal majority voting (runs 003, 104 and 203). The overall approach for task 2 gives lower results than task 1, but the task is also more challenging. The MLP classifier from the node embeddings is a baseline that can definitely be improved. For example, we could try to reduce the size of the very large graph by removing some nodes in order to remove noise. For the last task, the MLP fusion model, taking both text-based features and graph-based features (run 204), did not improve on the text-only CT-BERT model (run 201). The fusion model could still be improved to correctly use both kinds of data and improve the overall results.

Comparing these results with last year challenge, we did not see a major improvement in the overall score, even though we had access to more data this year. Regarding the results for each conspiracy theory, the best results are obtained for the 'Harmful Radiation/Influence' (0.830), 'New World Order' (0.830), and 'Population reduction' (0.876) conspiracies. The worst result is obtained for the conspiracy theory 'Antivax' (0.563). More data does not seem to correlate to better results for each conspiracy as well, since 'Antivax' has almost four times more data than 'Harmful Radiation/influence' and still perform significantly worse. This is partially due to the weighting of the loss function, which emphasizes the classes with a smaller number of samples.

5. Conclusion

In this paper, we propose a transformer-based method to detect COVID-19-related conspiracy theories in tweets, composed of an ensembling of CT-BERT models. We also propose a node embedding-based techniques to detect misinformation spreader in the user-interaction graph, using node2vec and an MLP classification head. Our best model obtains a MCC score of 0.719 on the test data.

Acknowledgements

This work has been partially supported by CHIST-ERA within the CIMPLE project (CHIST-ERA-19-XAI-003).

References

[1] K. Pogorelov, D. T. Schroeder, S. Brenner, , A. Maulana, J. Langguth, Combining Tweets and Connections Graph for FakeNews Detection at MediaEval 2022, in: Multimedia Benchmark Workshop, 2022.

[2] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, J. Gao, Deep Learning–Based Text Classification: A Comprehensive Review, ACM Computer Survey 54 (2021). URL: https://doi.org/10.1145/3439726.

[3] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, in: Conference of the North American Chapter of the Association for Computational Linguistics (ACL), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. URL: https://aclanthology.org/N19-1423.

[4] T. N. Kipf, M. Welling, Semi-Supervised Classification with Graph Convolutional Networks, in: International Conference on Learning Representations (ICLR), 2017.

[5] A. Grover, J. Leskovec, Node2vec: Scalable feature learning for networks, in: 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery, 2016, pp. 855—-864. URL: https://doi.org/10.1145/2939672.2939754.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, I. Polosukhin, Attention is All you Need, in: Advances in Neural Information Processing Systems (NeurIPS), Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[7] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, A. Rush, Transformers: State-of-the-art natural language processing, in: Conference on Empirical Methods in Natural Language Processing (EMNLP), System Demonstrations, Association for Computational Linguistics, 2020, pp. 38–45. URL: https://aclanthology.org/2020.emnlp-demos.6. doi:10.18653/v1/2020.emnlp-demos.6.

[8] M. Müller, M. Salathé, P. E. Kummervold, COVID-Twitter-BERT: A Natural Language Processing Model to Analyse COVID-19 Content on Twitter, 2020. arXiv:2005.07503.

[9] K. Pogorelov, D. T. Schroeder, S. Brenner, J. Langguth, FakeNews: Corona Virus and Conspiracies Multimedia Analysis Task at MediaEval 2021, in: Multimedia Benchmark Workshop, 2021.

[10] Y. Peskine, G. Alfarano, I. Harrando, P. Papotti, R. Troncy, Detecting COVID-19-related conspiracy theories in tweets, in: CEUR (Ed.), MediaEval Benchmarking Initiative for Multimedia Evaluation Workshop, 2021.

[11] M. Ilya, K. Dmitrii, N. Nikita, S. Lovro, Survey on graph embeddings and their applications to machine learning problems on graphs, PeerJ Computer Science (2021). URL: https://doi.org/10.7717/peerj-cs.357.

[12] A. A. Hagberg, D. A. Schult, P. J. Swart, Exploring network structure, dynamics, and function using networkx, in: 7th Python in Science Conference, Pasadena, CA USA, 2008, pp. 11–15.

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.