Tackling Covid-19 Conspiracies on Twitter using BERT Ensembles, GPT-3 Augmentation, and Graph NNs

Damir Korenčić[1,2], Ivan Grubišić[2], Gretel Liz De La Peña Sarracén[1], Alejandro Hector Toselli[1], Berta Chulvi[1] and Paolo Rosso[1]

[1]Universitat Politècnica de València, Spain
[2]Ruđer Bošković Institute, Zagreb, Croatia

Abstract

We describe several approaches to text- and graph-based classification for detecting COVID-19 conspiracies on Twitter. We tackle the tasks of text classification with and without graph data, and classification of Twitter users based on user graph. To this end, we experiment with large transformer ensembles, GPT-3-based techniques, and a variety of graph neural networks. Our results demonstrate that transformer ensembling and GPT-3 text augmentation can improve performance and stability, and that richer graph data does not necessarily lead to improved performance.

1. Motivation and Background

The COVID-19 pandemic has increased the amount of people's exposure to digital content and communications [1]. One of the effects of the rapid digitalization of everyday activities is the increase of online disinformation, including conspiracy theories [2]. A conspiracy theory can be defined as a belief that two or more actors have coordinated in secret to achieve a malevolent outcome [3]. The automatic detection of conspiracy messages on social networks has a huge potential for combating disinformation.

MediaEval 2022 FakeNews challenge consists of three subtasks focused on detecting COVID-19 conspiracy theories in tweets. Nine different conspiracy categories are defined. Subtask 1 consists of determining, for each conspiracy theory, whether a tweet supports it, mentions it, or ignores it. In Subtask 2 the goal is to determine, based on a social network graph, whether a Twitter user is a misinformation poster. The goal of Subtask 3 is the same as that of Subtask 1, but the model can use the tweet's user ID and all the graph data. Both the text and the user datasets contain 1,913 labeled development tweets/users and 830 test tweets/users. In addition, a large user graph, with 1,679,011 vertices and 268,694,698 edges, is provided. The metric of the models' performance assessment is Matthews correlation coefficient (MCC) [4]. A detailed description of the subtasks and the datasets can be found in Pogorelov et al. [5].

Our approach to Subtask 1 was to improve on the best-performing transformer-based system of MediaEval 2021 [6] by adding additional features, creating model ensembles, and augmenting the training data via GPT-3 model. [7]. For subtasks 2 and 3 we designed and evaluated a number of GNN models [8], each consisting of two convolutional layers, three fully connected layers, and a softmax classification layer.

Few works have addressed the detection of conspiracy theories. These include the classification models proposed in the MediaEval 2021 challenge [9] and in the work of Giachanou et al. [10]. There exist numerous approaches to text classification [11] and in recent years deep learning models and especially transformers have led to state-of-art results [12]. Graph neural

MediaEval'22: Multimedia Evaluation Workshop, January 13–15, 2023, Bergen, Norway and Online
damir.korencic@irb.hr (D. Korenčić); ivan.grubisic@irb.hr (I. Grubišić); gredela@posgrado.upv.es (G. L. D. L. P. Sarracén); ahector@prhlt.upv.es (A. H. Toselli); berta.chulvi@upv.es (B. Chulvi); prosso@dsic.upv.es (P. Rosso)

networks (GNNs) have been widely used in recent years for different classification tasks [13, 14], including text classification [15, 16, 17] where they can lead to top results [15, 16]. The strength of GNNs is their ability to model relationships between the data points and integrate them with other features.

2. Approach to Text-based Classification

Subtask 1 of MedEval FakeNews 2022 is identical to Subtask 3 of MedEval 2021, but with a larger dataset. Our approach is therefore to improve on the best system from the 2021 challenge [6] (*hard baseline*), based on domain-specific COVID-Twitter-BERT (CT-BERT) [18] fine-tuned in a multi-task manner. We attempted to append additional features (TF-IDF, topic models, and hand-crafted features) before applying the classification layer, but this led to no improvement. We also experimented with different architectures for the multi-task classifier layer (attention, additional layers), but the basic logistic regression gave the best results.

We improved upon the hard baseline model by combining a large number of transformers and examining different options for ensemble creation. This approach is motivated by the observation that model performances vary significantly, in relation to both the `eval`/`dev`/`train` split and random initialization. Individual models all use the same preprocessing and training parameters and differ only in random seeds and train/dev data splits.

Preprocessing and learning settings were tweaked in the initial development phases of the experiment. We tried the text preprocessing options from [6] but opted to use no preprocessing for the final models. We used the Adam optimizer with a weight decay of 10^{-3} and an initial learning rate (LR) of $7 * 10^{-5}$. We reduced the LR by a factor of 0.3 after 3 epochs of no improvement and stop the training after 6 such epochs. LR used for fine-tuning the transformer parameters is $10\times$ smaller. We used a nested evaluation scheme based on stratified cross-validation. On the first level, 7-fold splitting is performed, followed by the second level 6-fold splitting. This way the development data is split into `eval` (14.3%), `dev` (14.3%), and `train` (71.4%) subsets.

We also attempted to use the GPT-3 (davinci) model, by way of in-context learning [7, 19]. This approach proved ineffective, but a simple data augmentation method – the addition of GPT-rephrased tweets with original labels – yielded competitive results. Additionally, we experimented with the Xgboost classifier [20] applied to topic model features.

3. Approach to Graph-based Classification

Graph-based User Classification (Subtask 2) Subtask 2 consists of classifying Twitter users as (not) being misinformation spreaders, and we tackle it using graph neural networks (GNNs). The graph nodes are described by a matrix A where each row is a vector containing a single user's attributes (location, number of followers, ...). Note that our models do not take into account the user's textual information, in order to examine the viability of using graph-only data. Edges are represented with a square matrix R – each element corresponds to the weight of the relation between two users (the weight being 0 if there is no relation).

GNN classification models consist of two convolutional layers, followed by three fully connected layers of 150 neurons (each followed by a normalization layer), followed by the standard softmax classification layer. We have tested 16 combinations of different convolutional layers – GCNConv [21], GATv2Conv [22], TAGConv [23] and RGGConv [24]. We trained the model in 200 epochs with a learning rate of 0.01 and a batch size of 64.

Graph-based Text Classification (Subtask 3) Subtask 3 has the same goal as Subtask 1 – classifying texts for each of the 9 conspiracy categories. However, in Subtask 3 the information

about user connectivity can be added to the classifiers. We used the same approach as for Subtask 2, but with a modified graph-construction method.

In this subtask, the nodes are texts (instead of users) and each node is represented with a vector of features extracted from a text using pretrained transformer models. Optionally, we added the vector of user attributes describing the text's author. An edge exists between two text nodes if the authors of the corresponding texts are related, and the edge weight is derived as in Subtask 2. Another difference with Subtask 2 is that here we solve a multi-label and multi-class classification problem (instead of binary classification). We trained a classifier per conspiracy category and combined the results to obtain the final output.

Table 1
Subtask 1 text classification MCCs for single basic model (0), "backup" ensemble (1), best-performing ensembles (2–4), ensembled (5), and single (6) models with GPT-3-augmented data, and the Xgboost classifier (7). All the MCCs except the "test" column are calculated on the dev set.

model	score	aggreg.	#models	mean	std	min	max	test
0				0.724	0.027	0.592	0.797	–
1	avg	mult	1	0.749	0.022	0.723	0.785	0.734
2	avg	mult	11	**0.757**	0.025	0.722	0.793	0.733
3	avg	mult	8	0.757	0.029	0.717	0.805	0.735
4	cat	vote	5	0.755	0.030	0.706	0.796	0.738
5	avg	vote	1	0.750	0.028	0.704	0.783	**0.738**
6				0.745	0.021	0.701	0.788	–
7				0.446	0.046	0.394	0.526	0.376

4. Results and Analysis of the Text-based Classification

Best-performing models were selected by varying several options for scoring and ensemble creation. The combinations were evaluated on `eval`/`dev`/`train` splits – each outer `eval` split was used to calculate the performance of models and ensembles created from inner splits. For each inner split, we trained 40 models (1680 models in total), changing only the random seed for initializing the classification layer parameters.

For each inner `dev`/`train` split, the best model from all the epochs was selected according to a scoring method (score option) – either by selecting a model with top average MCC, or the top model per conspiracy category. Optionally, a number of top models (up to 15) were selected (#models option). All the models of inner splits were aggregated, either by majority voting (`vote`) or by multiplication of the models' class probabilities (`mult`). For each combination of options, all the models from inner splits are aggregated and the performances of these ensembles on `eval` splits are averaged to obtain the performance assessment for model selection. For the final submission, we selected the best ensemble configurations (models 2–4) and used the average of `eval`, `dev`, and `train` split scores for inner fold model selection. We also created and evaluated a "backup" model (model 1) created with the default options and using only `eval` splits for inner model scoring. The ensemble of models trained with the GPT3-augmented data (model 5) was created in the same manner as the backup model. Augmentation was performed by rephrasing the tweets of the entire development set and using them (labeled with original labels) to duplicate the size of each `train` split. The best version of the Xgboost classifier (model 7) uses topic models and SVD features.

The results are presented in Table 1. It can be seen that ensembling improves the single-model solution (model 0), by increasing the expected mean performance and the worst-case performance. Interestingly, GPT-3 augmentation without ensembling (model 6) has a similar effect. All the deep ensemble variants perform very similarly and have good generalization in terms of dev/test score differences.

5. Results and Analysis of the Graph-based Classification

Graph-based User Classification (Subtask 2) In the construction of the model we consider 4 different convolutional layers described in Section 3 (GCN, GATv2, TAG, and RGGConv). We evaluated all 16 two-layer combinations to discover the best design for the task. We also considered two types of graphs, one with all the users in the dataset (big) and another with only the users from dev and train sets of Subtask 2 (small).

Table 2
Results for Subtask 2 (top half) and Subtask 3 (bottom half), for different GNN models.

Combination of layers	Type of graph	ACC	MCC on dev	MCC on test
GCN-GCN	big	0.7209	0.2780	0.2411
GCN-GCN	small	0.7006	**0.3111**	0.2319
GATv2-GATv2	small	0.6890	0.1605	-
TAG-TAG	small	0.6831	0.2927	0.1301
RGGConv-RGGConv	small	0.7180	0.2731	-
TAG-GCN	small	0.6686	0.2109	**0.2831**
RGGConv+GCN	small	0.6977	0.2997	0.2480
Combination of layers	User encoding	ACC	MCC on dev	MCC on test
GCN-GCN	user+miniLM	0.9139	0.0220	-
GCN-GCN	miniLM	0.9148	0.4338	-
GCN-GCN	covid-bert	0.9177	0.4833	-
RGGConv-RGGConv	covid-bert	0.9319	0.5303	**0.5036**
TAG-GATv2	covid-bert	0.9235	0.5224	0.4906
TAG-GATv2	user+covid-bert	0.9153	0.0657	-

Table 2 shows the results of some model variants for Subtask 2. We can see that using the big graph does not seem to improve the results and that the GCN layer seems to be a good choice. The best model on the dev set does not match the best model on test.

Graph-based Text Classification (Subtask 3) For Subtask 3 we evaluated combinations of two node representation methods: text embeddings and user attributes. We evaluated two models to obtain the embeddings: COVID-Twitter-BERT-v2 (covid-bert) and all-MiniLM-L6-v2 (miniLM). The results in Table 2 indicate the importance of the node representation – the MCC values dropped steeply after user attributes were added to text embeddings, and the embeddings obtained with the COVID-Twitter-BERT-v2 model seem to perform better.

Additionally, we submitted text-only solutions described in Section 4 – two best transformer ensembles and the Xgboost classifier. The two ensembles have test MCCs of 0.698 and 0.697, while the Xgboost classifier has a test MCC of 0.395. This indicates that the graph information might be detrimental to text classification.

6. Discussion

The results for Subtask 1 show that both large transformer ensembles and simple GPT-3 text augmentation can improve the performance of individual models and lower the expected performance variance. However, the performance of the best models is similar and it seems to have reached a limit of cca. 0.74 MCC. This could be due to the inherent task complexity, or to the noise present in existing labels. It is interesting that the "in-context learning" with GPT-3, which tends to produce decent results [7, 19], failed for this use case (with 0.172 MCC). We hypothesize that this is due to inherent task complexity: GPT-3 cannot distinguish, based on complex prompts with category descriptions, between fine-grained conspiracy categories.

Acknowledgments: This work was carried out in the framework of the following projects: XAI-Disinfodemics (PLEC2021-007681), IBERIFIER (INEA/CEF/ICT/A202072381931, n. 2020-EU-IA-0252), and MARTINI CHIST-ERA (PCI2022-134990-2). It was also supported by the Maria Zambrano grant (Spanish Ministerio de Universidades, NextGenerationEU/PRTR).

References

[1] European Commission, Digital economy and society index (desi) 2022, 2022. URL: https://digital-strategy.ec.europa.eu/en/policies/desi.

[2] B. European Commission, Flash eurobarometer 464 (fake news and disinformation online), GESIS Data Archive, Cologne. ZA6934 Data file Version 1.0.0, 2018. doi:10.4232/1.13019.

[3] K. M. Douglas, M. S. Robbie, What are conspiracy theories? a definitional approach to their correlates, consequences, and communication, Annual review of psychology (2022).

[4] J. Gorodkin, Comparing two k-category assignments by a k-category correlation coefficient, Computational biology and chemistry 28 (2004) 367–374.

[5] K. Pogorelov, D. T. Schroeder, S. Brenner, A. Maulana, J. Langguth, Combining tweets and connections graph for fakenews detection at mediaeval 2022, in: Proceedings of the MediaEval 2022 Workshop, Bergen, Norway and Online, 12-13 January 2023., 2023.

[6] Y. Peskine, G. Alfarano, I. Harrando, P. Papotti, R. Troncy, Detecting covid-19-related conspiracy theories in tweets, in: MediaEval 2021, MediaEval Benchmarking Initiative for Multimedia Evaluation Workshop, 13-15 December 2021, 2021.

[7] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, Advances in neural information processing systems 33 (2020) 1877–1901.

[8] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, S. Y. Philip, A comprehensive survey on graph neural networks, IEEE transactions on neural networks and learning systems 32 (2020) 4–24.

[9] K. Pogorelov, D. T. Schroeder13, S. Brenner, J. Langguth, Fakenews: Corona virus and conspiracies multimedia analysis task at mediaeval 2021, in: Multimedia Benchmark Workshop, 2021, p. 67.

[10] A. Giachanou, B. Ghanem, P. Rosso, Detection of conspiracy propagators using psycho-linguistic characteristics, Journal of Information Science (2021). doi:0165551520985486.

[11] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, D. Brown, Text Classification Algorithms: A Survey, Information 10 (2019). doi:10.3390/info10040150.

[12] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, J. Gao, Deep Learning-based Text Classification: A Comprehensive Review, ACM Computing Surveys 54 (2022) 1–40. URL: https://dl.acm.org/doi/10.1145/3439726. doi:10.1145/3439726.

[13] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, M. Sun, Graph neural networks: A review of methods and applications, AI Open 1 (2020) 57–81.

[14] R. Sato, A survey on the expressive power of graph neural networks, arXiv:2003.04078 (2020).

[15] K. Wang, S. C. Han, S. Long, J. Poon, Me-gcn: Multi-dimensional edge-embedded graph convolutional networks for semi-supervised text classification, arXiv preprint arXiv:2204.04618 (2022).

[16] G. L. De la Peña Sarracén, P. Rosso, Unsupervised embeddings with graph auto-encoders for multi-domain and multilingual hate speech detection, in: Proceedings of the Thirteenth Language Resources and Evaluation Conference, 2022, pp. 2196–2204.

[17] G. L. De la Peña Sarracén, P. Rosso, Convolutional graph neural networks for hate speech detection in data-poor settings, in: International Conference on Applications of Natural Language to Information Systems, Springer, 2022, pp. 16–24.

[18] M. Müller, M. Salathé, P. E. Kummervold, Covid-twitter-bert: A natural language processing model to analyse covid-19 content on twitter, arXiv preprint arXiv:2005.07503 (2020).

[19] J. T. Ornstein, E. N. Blasingame, J. S. Truscott, How To Train Your Stochastic Parrot: Deep Language Models for Political Texts (2022) 30.

[20] T. Chen, T. He, M. Benesty, V. Khotilovich, Y. Tang, H. Cho, K. Chen, et al., Xgboost: extreme gradient boosting, R package version 0.4-2 1 (2015) 1–4.

[21] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, CoRR abs/1609.02907 (2016). arXiv:1609.02907.

[22] S. Brody, U. Alon, E. Yahav, How attentive are graph attention networks?, CoRR abs/2105.14491 (2021). arXiv:2105.14491.

[23] J. Du, S. Zhang, G. Wu, J. M. F. Moura, S. Kar, Topology adaptive graph convolutional networks, CoRR abs/1710.10370 (2017). arXiv:1710.10370.

[24] X. Bresson, T. Laurent, Residual gated graph convnets, CoRR abs/1711.07553 (2017). arXiv:1711.07553.