Representational learning for the detection of COVID related conspiracy spreaders in online platforms

Adrián Girón Jiménez[2,*,†], Ángel Panizo-LLedot[1,†], Javier Torregrosa[1,†] and David Camacho[1,†]

[1]ETSI de Sistemas Informáticos, Universidad Politécnica de Madrid
[2]Dept. Computer Sciences, Universidad Rey Juan Carlos

Abstract

The approach known as representational learning is a set of techniques that allows the automatic discovery of features required for a machine learning task from raw data. In recent years, the application of these techniques to graphs has shown promising results in node classification tasks. This work applies representational learning to identify users that share COVID-related conspiracy theories, using their interactions with peers as the main features for the classification algorithms. To do so, Node2vec and FastRP were used to learn numeric representations, i.e. embeddings, of the users. Then, Random Forest and XGBoost were used for the downstream classification task. In addition, a pseudo-labeling procedure was applied. The experimentation shows that using interaction data for the classification task achieves better performance compared to classifying using only node attributes. Moreover, FastRP achieve better performance compared to Node2vec. However, pseudo-labeling does not improve the performance of the models at all. Finally, we reject the inclusion of "cannot determine" labels in our model, as they prove to be detrimental.

1. Introduction

This work introduces a social network analysis approach to detect nodes spreading conspiracy theories related to COVID[1]. The overview paper [1] explains the task in depth. The paper focuses on the actors, rather than the messages, and their interactions within a network as features for classification. In particular, it focuses on the use of representational learning techniques [2] to generate user embeddings in a semi-supervised manner, i.e. using unlabeled nodes related to the original training sample, to be used in a downstream classification task [3].

2. Approach

Random Forest [4] and *XGBoost* [5] were selected as classifiers heads due to their good general performance in different tasks [6]. Additionally, given the unbalanced nature of the dataset, we have opted for the use of weights, assigning greater importance to the spreaders class. Concerning the graph, due to its size, many of the techniques to be applied were not feasible. Therefore, the most superfluous connections, i.e. those edges with a weight of less than a threshold, were incrementally removed until a graph with a feasible size was reached. This was achieved with a threshold of five. However, as this generated several connected components, all the superfluous edges that touched any of the nodes under study, i.e. those with a label or those that need to be labeled, were added. Finally, all nodes outside the biggest component were discarded. The final graph had $1,574,681$ nodes and $39,946,463$ edges.

MediaEval'22: Multimedia Evaluation Workshop, January 13–15, 2023, Bergen, Norway and Online
*Corresponding author.
†All the authors contributed equally.
adrian.giron@urjc.es (A. G. Jiménez); angel.panizo@upm.es (: Panizo-LLedot); franciscojavier.torregrosa@upm.es (J. Torregrosa); david.camacho@upm.es (D. Camacho)

[1]https://multimediaeval.github.io/editions/2022/

2.1. Node attributes only

As a baseline standard, a classification model using only the node attributes was created. The information from each node (*Twitter* account) available for the classifier is the following: creation date (number of days after Twitter's creation), description length, number of favorites, number of statuses, number of friends, and country (as one hot encoding + "unknown_country"). All the data was normalized between 0 and 1.

2.2. Representational learning

Representational learning techniques generate vectors (also known as embeddings) so that nodes that are similar in the graph are closer together in the embedding space [2]. Once the embeddings for each node were calculated, they were used in a downstream classification task. For this work, two representation learning techniques were used: node2vec [7] and FastRP [8]. The former is a popular method that has proven good results in node classification tasks [9]. The latter is a random projection algorithm that is capable of generating embeddings that take into account node attributes, which node2vec cannot do.

2.3. Pseudo-labeling

Pseudo-labelling is a semi-supervised technique that selects unlabeled samples that a model has classified with high confidence and adds them to the training set. Rizve et al. [10] argue that pseudo-labeling performance is usually low due to erroneous high-confidence predictions from poorly calibrated models; these predictions generate many incorrect pseudo-labels, resulting in noisy training. To correct this problem they propose an uncertainty-aware pseudo-label selection framework. Originally, the authors propose their framework to be used with neural networks. Therefore, in this work, we adapted that framework to work with tree ensembles. In particular, we changed the uncertainty estimation method MC-Dropout [11] to the method proposed by Polimis et al. [12].

2.4. "Cannot Determine" labels

The ability of the model to identify when a sample cannot be determined was assessed using two approaches. The first uses the output probabilities generated by the model. When the probability is lower than a threshold, the sample will be labeled as "Cannot Determine". The second uses the confidence of the model's predictions instead of the output probabilities. Finally, to calculate the confidence of a model's prediction the method proposed by Polimis et al. [12] was used.

3. Results

3.1. Validation and hyperparameter tuning

To obtain robust metrics we follow the Stratified KFolds cross-validation method with 10 folds. The *Matthews correlation coefficient* (MCC) [13] was used as the evaluation metric. To evaluate each model, the mean and standard deviation of the scores obtained in each fold was computed. In addition, *Optuna* [14] framework was used for hyperparameter tuning. Table 1[2] and 2[3] shows

[2]For the rest of the values of the hyperparameters refer to the default in https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html and https://xgboost.readthedocs.io/en/stable/python/python_api.html#xgboost.XGBClassifier

[3]For the rest of the hyperparameters refer to the default values in https://neo4j.com/docs/graph-data-science/current/machine-learning/node-embeddings/fastrp/ and https://neo4j.com/docs/graph-data-science/current/machine-learning/node-embeddings/node2vec/

Random Forest	Node attributes	Node2Vec	FastRP	FastRP optimized
n_estimators	132	128	120	183
min_samples_leaf	3	2	4	0.22
min_samples_split	2	3	2	0.22
max_depth	14	15	15	7
class_weight(1/2)	1.0/2.001	1.0/2.547	1.0/2.109	0.615/1.161

XGBoost	Node attributes	Node2Vec	FastRP
learning_rate	0.084	0.020	0.037
min_child_weight	3	3	5
gamma	4.65	4.66	3.80
subsample	0.92	0.97	0.97
colsample_bytree	0.62	0.62	0.59
max_depth	12	16	14
scale_pos_weight	3.94	3.35	3.39

Table 1
Hyperparameters selected for ensemble models. Rows are hyperparameters and columns are different approaches.

FastRP	default	optimized
embedding dimensions	64	416
iteration weights	[0.0, 0.0, 0.5, 1.0]	[0.0, 0.0, 0.7, 0.1]
normalization strength	-0.5	0.78
property ratio	0.11	0.20

Node2vec	default
embedding dimension	64
walk length	100
walks per node	17
in out factor	0.88
return factor	0.57

Table 2
Hyperparameters selected for representation techniques. Rows are hyperparameters and columns are different approaches.

the values selected for the hyperparameters.

3.2. Ensemble results

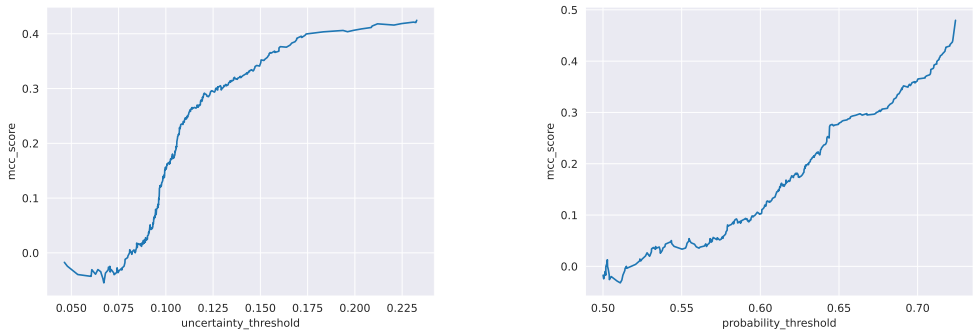
Approach	**Random Forest**	**XGBoost**
Node attributes	0.130 (0.054)	0.156 (0.055)
Node2Vec	0.129 (0.061)	0.115 (0.088)
FastRP	0.259 (0.063)	0.301 (0.030)
FastRP optimized	**0.434 (0.071)**	

Table 3
MCC scores obtained by the different approaches and tree ensembles. Mean and standard deviation obtained in the 10 folds

Table 3 contains the 10-fold MCC means and their standard deviation. The row "node attributes" refers to the baseline approach; "Node2vec" to the representational learning approach using the default hyperparameters; "FastRP" is the same as "Node2vec" but using the FastRP method instead; and, "FastRP optimized" is the representational learning approach where both the hyperparameters of FastRP and random forest were optimized at the same time.

3.3. "Cannot Determine" labels

Figure 1 shows the variation of the MCC score when different thresholds are selected for the *FastRP optimized* model. The graph on the right shows the results of the model's confidence in the prediction, while the graph on the left shows the results of the output probability. As we can see, labeling samples as "Cannot Determine" did not improve the model performance. Please note that the maximum value is always obtained at the maximum possible value of the threshold. Hence, no sample is labeled as "Cannot Determine".



(a) (b)

Figure 1: Variation of MCC according to the chosen threshold. On the left, the results for the model confidence in the prediction. On the right, the results for the output probability.

3.4. Pseudo-labeling

To evaluate the effectiveness of the pseudo-labeling procedure, the MCC performance obtained by the *FastRP optimized* model trained only with the labeled data was compared to the MCC performance obtained by the same model, but trained with the pseudo-labeling procedure. For this procedure, $10,000$ extra unlabeled nodes were randomly selected. A θ_{proba} of 0.7, and a $\theta_{uncertain}$ of 0.15 were selected after manual experimentation. This process has been repeated for each fold of a stratified KFold validating procedure with 31 iterations.

At the end of the pseudo-labeling procedure carried during each fold, at least 95% of unlabeled samples were used to train the model. However, a p-value of 0.75 in a Kruskal-Wallis H-test showed that applying the pseudo-labeling procedure, for this particular setup, was not beneficial for the task.

4. Discussion and outlook

This work presents a model to detect COVID-related conspiracy theory spreaders on online platforms. Four approaches were proposed: (i) baseline model using only the node's attributes and no graph features; (ii) representation learning model using node2vec and FastRP to calculate node embeddings; (iii) pseudo-labeling procedure to take advantage of the big amount of unlabeled data; (iv) labeling nodes as "cannot determine" when the model's confidence in a prediction is low.

From our experimentation, it can be concluded that for our particular setup: (i) using the topology of the network is beneficial for this problem as the models that used node embeddings were superior to one using only node attributes; (ii) embeddings generated using FastRP performed better than the ones generated using node2vec, as FastRP can take into account node attributes and topology features, while node2vec can only use topology features; (iii) models trained following our approaches do not benefit from using "Cannot determine" labels, as the experiments show that they have the same distributions of confidence in their predictions when they are right or wrong; (iv) applying a pseudo-labeling procedure does not further improve the performance of the model. (v) finally, while the results are promising for a preliminary attempt, they are not good enough for a real-world system and need to be refined.

Acknowledgements

This work has been supported by the research project DisTrack: Tracking disinformation in Online Social Networks through Deep Natural Language Processing, granted by Barcelona Mobile World Capital Foundation; by the Spanish Ministry of Science and Innovation under FightDIS (PID2020-117263GB-I00); by MCIN/AEI/10.13039/501100011033/ and European Union NextGenerationEU/PRTR for XAI-Disinfodemics (PLEC2021-007681) grant, by Comunidad Autónoma de Madrid under S2018/TCS-4566 grant, by European Comission under IBERIFIER - Iberian Digital Media Research and Fact-Checking Hub (2020-EU-IA-0252); by Comunidad Autonoma de Madrid under grant S2018/TCS-4566 (CYNAMON: Cybersecurity, Network Analysis and Monitoring for the Next Generation Internet); This work is part of the project PCI2022-134990-2 (MARTINI) of the CHISTERA IV Cofund 2021 program, funded by MCIN/AEI/10.13039/501100011033 and by the "European Union NextGenerationEU/PRTR"; and by "Convenio Plurianual with the Universidad Politécnica de Madrid in the actuation line of Programa de Excelencia para el Profesorado Universitario"

References

[1] K. Pogorelov, D. T. Schroeder, S. Brenner, A. Maulana, J. Langguth, Combining tweets and connections graph for fakenews detection at mediaeval 2022, 2023.

[2] Y. Bengio, A. Courville, P. Vincent, Representation learning: A review and new perspectives, IEEE transactions on pattern analysis and machine intelligence 35 (2013) 1798–1828.

[3] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, Advances in neural information processing systems 30 (2017).

[4] L. Breiman, Random forests, Mach. Learn. 45 (2001) 5–32. URL: https://doi.org/10.1023/A:1010933404324. doi:10.1023/A:1010933404324.

[5] T. Chen, C. Guestrin, XGBoost, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016. URL: https://doi.org/10.1145%2F2939672.2939785. doi:10.1145/2939672.2939785.

[6] O. Sagi, L. Rokach, Ensemble learning: A survey, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 8 (2018) e1249.

[7] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 2016, pp. 855–864.

[8] H. Chen, S. F. Sultan, Y. Tian, M. Chen, S. Skiena, Fast and accurate network embeddings via very sparse random projection, in: Proceedings of the 28th ACM international conference on information and knowledge management, 2019, pp. 399–408.

[9] P. Goyal, E. Ferrara, Graph embedding techniques, applications, and performance: A survey, Knowledge-Based Systems 151 (2018) 78–94.

[10] M. N. Rizve, K. Duarte, Y. S. Rawat, M. Shah, In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning, arXiv preprint arXiv:2101.06329 (2021).

[11] A. Gammerman, V. Vovk, V. Vapnik, Learning by transduction, vol uai'98, 1998.

[12] K. Polimis, A. Rokem, B. Hazelton, Confidence intervals for random forests in python, Journal of Open Source Software 2 (2017).

[13] P. Baldi, S. Brunak, Y. Chauvin, C. A. F. Andersen, H. Nielsen, Assessing the accuracy of prediction algorithms for classification: an overview , Bioinformatics 16 (2000) 412–424. URL: https://doi.org/10.1093/bioinformatics/16.5.412. doi:10.1093/bioinformatics/16.5.412. arXiv:https://academic.oup.com/bioinformatics/article-pdf/16/5/412/476945/160412.pdf.

[14] T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama, Optuna: A next-generation hyperparameter optimization framework, in: Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2019.