Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 November 2020 doi:10.20944/preprints202011.0511.v1

Article

Rumor detection based on SAGNN: Simplified Aggregation Graph Neural Networks

Liang Zhang [1]*, Jingqun Li [2], Bin Zhou [1], Yan Jia [1]

[1] College of Computer, National University of Defense Technology, Changsha 410073, China; gfkdzliang@163.com
[2] Shenzhen LiCi electronic company, Guangdong, China

Abstract: Identifying fake news on the media has been an important issue. This is especially true considering the wide spread of rumors on the popular social networks such as Twitter. Various kinds of techniques have been proposed to detect rumors. In this work, we study the application of graph neural networks for the task of rumor detection, and present a simplified new architecture to classify rumors. Numerical experiments show that the proposed simple network has comparable to or even better performance than state-of-the art graph convolutional networks, while having significantly reduced the computational complexity.

Keywords: Rumor detection; Graph neural network; Artificial intelligence

0. Introduction

Rumors are messages which have not been verified. False rumors are those which communicate fabricated news. Identifying false rumors has now become a major concern for effective use of social media like Twitter and Instagram because of the wide availability of these social media platforms. Rumors can propagate very fast and might have big negative impacts on the society. However, it is a complicated matter to identify rumours from massive amounts of online information. Therefore, it is necessary and highly desirable to develop automatic approaches to detect rumors at an early stage in order to mitigate their damages.

Early studies on automatically detecting rumors mainly focused on designing effective features from various information sources, including text content, publisher's profiles, and propagation patterns [1],[2]. However, these feature-based methods are extremely time-consuming, biased, and labor-intensive. Furthermore, if one or several types of hand-crafted features are unavailable, inadequate or manipulated, the effectiveness of these approaches will be affected.

Motivated by the success of deep learning, many recent studies apply various neural networks for rumor detection. For example, recurrent neural network [3]] is applied to learn a representation of tweet text over post time.

Latest efforts focus on applying graph learning techniques for rumor detection [4], [5], [6], due to the development of graph neural networks and the fact that posts on social media are naturally structured as graphs.

The rest of the paper is organized as follows. First, we review the related rumor detection algorithms and graph convolutional networks in Section 1. Then the proposed simple aggregation

*The work described in this paper is partly supported by the National Key Research and Development Program of China (No. 2018YFC0831703)

© 2020 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 November 2020 doi:10.20944/preprints202011.0511.v1

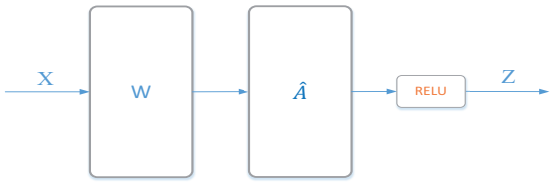

Figure 1. Classical GCN convolution layer

network architecture is presented, along with the learning procedure. Next, we give experiment results of applying the SAGNN architecture to two publicly available datasets. Finally, some conclusions are drawn about the simple aggregation network and its possible applications

1. Related work

1.1. Existing rumor detection methods

As suggested in the introduction, rumor detection related work can be divided into following categories [5]: (1) feature-based methods; (2) propagation tree related methods; (3) deep learning methods.

A: Feature-based methods

Early rumor detection studies were based on hand-crafted features extracted from text content and users' profile information. Features based on the text contents, users, topics and propagation patterns of messages were examined in [7] to measure the credibility of news on Twitter. Temporal characteristics of the features were explored in [1] to incorporate various social context information, based on the time series of rumor's life cycle.

B: Propagation tree related methods Different from the previous methods that focus on the text information, the propagation of trees related methods focuses on the differences in the characteristics of real and false information transmission. These include a kernel-based method called propagation tree kernel was proposed in [2] to capture high-order patterns differentiating different types of rumors by evaluating the similarities between their propagation tree structures.

See [2], [5] for a more detailed review of other feature-based and propagation tree related methods.

C: Deep learning methods

To address the difficulties with hand-crafted features, deep learning models have been applied to automatically learn efficient features for rumor detection in recent years. A recurrent neural networks (RNN) based model was proposed in [3] to learn text representations of relevant posts over time. It is the first study to introduce the deep learning methods into rumor detection. Later on, Ma et al. proposed a recursive network architecture for rumor detection[8]. Another development along this line was [9], in which a GAN based architecture was proposed.

Yu et al. proposed a convolutional method for misinformation identification based on Convolutional Neural Network (CNN), which can capture high-level interactions among significant features [10].

In [11], a multi-module deep learning model was proposed to capture text and user characteristics of messages. Liu et al. modeled the propagation path as multivariate time series, and applied both recurrent and convolutional networks to capture the variations of user characteristics along the propagation path [12].

A global-local attention network (GLAN) was proposed in [5] for rumor detection, which jointly encodes the local semantic and global structural information. A bidirectional graph convolutional network architecture was proposed in [6] to further enhance the rumor detection performance.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 November 2020 doi:10.20944/preprints202011.0511.v1

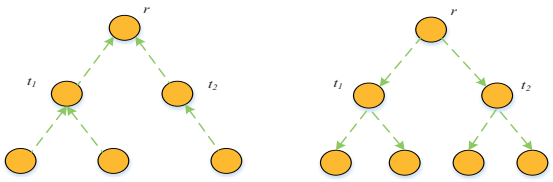

Figure 2. Interaction between parents and children

1.2. Review of graph convolutional network

Consider a network defined by a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$. A classical graph convolutional layer is given by

$$Z = \sigma(\hat{A} \cdot X \cdot W) \tag{1}$$

as shown in Fig.1 [13], where

$$\hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} \tag{2}$$

with

$$\tilde{A} = A + I, \tag{3}$$

where $A \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ is the adjacent matrix of the graph, and

$$\tilde{D} = \begin{bmatrix} \tilde{d}_1 & & \\ & \ddots & \\ & & \tilde{d}_{|\mathcal{V}|} \end{bmatrix} \tag{4}$$

with

$$\tilde{d}_i = \sum_{j=1}^{|\mathcal{V}|} \tilde{A}_{ij} \tag{5}$$

2. Simplified aggregation graph neural networks

Given a source twitter r, let us call the responses or retweets $(t_i)_{i=1}^{M_0}$ associated with r the children of r, where M_0 is the total number of the responses or retweets to r. Then the set of twitters/responses associated with r and its children and grand children and so on form a tree rooted at r, see Fig. 2, with each post represented by a node. Let $\mathcal{V}$ be the set of vertices of the tree.

The responses to true rumors and false rumors display different characteristics: When a post denies the false rumor, it tends to spark supportive or affirmative replies confirming the denial; in contrast, denial to a true rumor tends to trigger question or denial in its replies [8]. This observation suggests that it might be possible to distinguish true rumors and false rumors by considering the interaction between the twitters and their responses or retweets.

Recently, network representation learning has recently aroused a lot of research interest [14], [15], [16]. Graph neural networks (GNN) are promising architectures for this task. In particular for our rumor detection task, GNN provides an efficient solution, since a tree representing a twitter is a particular case of graphs [4], [5], and the recent [6].

To further improve rumor detection efficiency, we propose a simplified graph neural network architecture in this work, using the inherent aggregation mechanism of the graph neural network to calculate the interaction between a twitter and its children. The overall architecture is shown in Fig. 3.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 November 2020 doi:10.20944/preprints202011.0511.v1

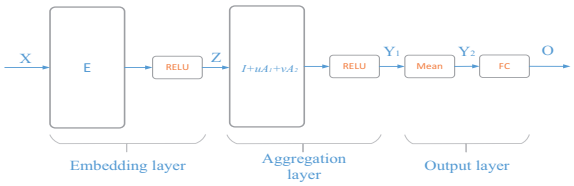

Figure 3. Simplified aggregation graph convolutional network

2.1. Simplified graph network architecture

2.1.1. Embedding layer

Let N_0 be the size of the vocabulary containing all words in the considered twitters. $E \in \mathbb{R}^{N_0 \times N}$ be a matrix. In our experiments, $N_0 = 5000$. The embedding layer can then be represented as

$$Z = \sigma(X \cdot E) \tag{6}$$

where $X = (x_{ij}) \in \mathbb{R}^{|\mathcal{V}| \times N_0}$ is the input features, σ is a nonlinear function, which is taken to be the *ReLU* function as is usually done in neural networks. N is a super-parameter, and will become the number of output features from the embedding layer.

2.1.2. Aggregation layers

The heart of the network is the aggregation layers, which are to implement the aggregation operations so as to capture the node information and characteristics of the interactions between nodes. Recall that each node represents a post.

A aggregation layer can be represented as

$$Y_1 = \sigma((I + uA_1 + vA_2) \cdot Z) \tag{7}$$

where Z is the output of the embedding layer of previous aggregation layer.

Compared to classical graph convolutional layer as shown in Fig.1 [13], our simplified GNN does not contain the weight matrix W. This can also be interpreted as fixing W to be an identity matrix, to put things in the framework of graph convolutional networks. Moreover, in classical GNNs, $\hat{A}$ is defined by Eq. 2, while in our model, the matrix is given by

$$\check{A} = I + uA_1 + vA_2 \tag{8}$$

where $A_1 = (p_{ij}) \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ and $A_2 = (c_{ij}) \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ are parent adjacent matrix and children adjacent matrix, respectively. More specifically,

$$p_{ij} = \begin{cases} 1, \text{ if node } i \text{ is the parent of node } j \\ 0, \text{ else;} \end{cases} \tag{9}$$

and

$$c_{ij} = \begin{cases} 1, \text{ if node } i \text{ is a child of node } j \\ 0, \text{ else.} \end{cases} \tag{10}$$

While $\hat{A}$ in traditional GCN is fixed, the matrix $\check{A}$ in our SAGNN contains two learnable parameters u and v, to distinguish the interactions from a node to its children and those from a node to its parent.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 November 2020 doi:10.20944/preprints202011.0511.v1

2.1.3. Output layer

$$Y_2 = Mean(Y_1) \tag{11}$$
$$O = FC(Y_2) \tag{12}$$

where $Y_1 \in \mathbb{R}^{|\mathcal{V}| \times N}$ is the output of the last aggregation layer, *Mean* represents an mean operation over the rows of Y_1, and FC is a fully connected linear layer of neural network.

2.2. Learning algorithm

As mentioned above, the SAGNN contains two learnable parameters u and v for each aggregation layer. Furthermore, the embedding layer contains a matrix E, and the output layer is fully connected layer, which is essentially a matrix. These constitute the learnable parameters of the SAGNN network. To determine these parameters, an optimization procedure is applied on a properly chosen loss function.

2.2.1. Cross entropy loss function

For classification problem, the cross entropy loss function is usually taken as the optimization objective function. In other words, we try to minimize

$$L(\hat{y}, y) = -\sum_{i}^{C} y_i \log(\hat{y}_i) \tag{13}$$

where C is the number of classes, $y = (y_1, \ldots, y_C)$ is the label vector with the only nonzero element being 1 at the i'th position if the sample is drawn from class i, $\hat{y}$ is the estimation given by

$$\hat{y} = Softmax(O) \tag{14}$$

where O is the output of the network shown in Fig. 3.

3. Experiments

3.1. Datasets

The datasets used in the experiment are the publicly available datasets *Twitter15* and *Twitter16* [3]. Each of these two datasets is divided into five groups of subsets. More specifically, *Twitter15* is divided into five subsets denoted by it Twitter150,..., *Twitter154* respectively, with each subset further divided into a train dataset and a test dataset.*Twitter16* is divided in a similar manner.

3.2. Network setup

We apply SAGNN to the ten subsets of *Twitter15* and *Twitter16* to evaluate its performance. The GCNII network [17] serves as the baseline to assess the proposed algorithm. In this experiment, SAGNN has two aggregation layers, unlike the one shown in Fig. 3, in which only a single aggregation layer is displayed for clarity . Correspondingly, GCNII has two convolutional layers.

For the loss function, a square regularization term is applied for both the SAGNN network and the GCNII network. Meanwhile, dropout layers are applied to both networks to overcome possible overfitting.

3.3. Results

The results are shown in Table 1 and Table 2.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 November 2020 doi:10.20944/preprints202011.0511.v1

Dataset	Method	Acc	F1			
			NR	FR	TR	UR
T150	SAGNN	0.857	0.851	0.892	0.867	0.826
	GCNII	0.823	0.796	0.85	0.864	0.786
T151	SAGNN	0.845	0.844	0.857	0.895	0.784
	GCNII	0.813	0.810	0.829	0.896	0.725
T152	SAGNN	0.796	0.846	0.817	0.810	0.706
	GCNII	0.773	0.775	0.790	0.834	0.698
T153	SAGNN	0.792	0.75	0.790	0.907	0.718
	GCNII	0.768	0.703	0.763	0.884	0.723
T154	SAGNN	0.802	0.8	0.771	0.824	0.813
	GCNII	0.769	0.761	0.719	0.861	0.742

Table 1. Results for dataset Twitter 15

Dataset	Method	Acc	F1			
			NR	FR	TR	UR
T160	SAGNN	0.764	0.526	0.783	0.877	0.791
	GCNII	0.802	0.718	0.849	0.873	0.731
T161	SAGNN	0.869	0.769	0.881	0.974	0.846
	GCNII	0.841	0.732	0.875	0.974	0.778
T162	SAGNN	0.816	0.817	0.836	0.919	0.698
	GCNII	0.847	0.824	0.853	0.947	0.769
T163	SAGNN	0.726	0.636	0.737	0.867	0.7
	GCNII	0.790	0.776	0.794	0.879	0.762
T164	SAGNN	0.802	0.769	0.771	0.9	0.8
	GCNII	0.753	0.625	0.788	0.872	0.735

Table 2. Results for dataset Twitter 16

Comparing these results, we see that the proposed SAGNN architecture and the more complicated GCNII give comparable results. For datasets like T150, T154, SAGNN even gives better results. Of course, for datasets like T160, the Acc score is lower for SAGNN than GCNII. However, the overall performance of SAGNN is better than GCNII.

It is also interesting to note the variations of the weights u and v during the training process, see Fig. WeightCurveT153 and Fig. WeightCurveT160. It can been seen, these curves converge in the training process. The variations of these weights show the same pattern for the two distinct datasets, though having different values. This demonstrates that it is indeed necessary to learn the matrix $hat A$ for better rumor detection performance, instead of keeping them fixed. This might explain why the simple SAGNN networks can compete the more complicated GCNII network.

4. Conclusion

Observing that different types of rumors trigger different interactions between source twitters and their responses, we suggest it is possible to classify rumors by aggregating the information around nodes representing twitters through simplified aggregation operations. Based on this observation, we claim that it is possible to simplify traditional graph convolutional neural networks for rumor detection applications. The proposed simplified aggregation neural network gives comparable to or even better results than the complicated GCNII architecture. This suggests that it is the aggregation operation which can capture different characteristics of distinct rumors.

The proposed simple aggregation layers can be further applied in more complicated architecture as in [4],[6]. What is essential is that it is possible to improve the rumor detection performance through capturing the interactions between nodes in a twitter tree by learning the combination coefficients of node feature vectors.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 November 2020 doi:10.20944/preprints202011.0511.v1

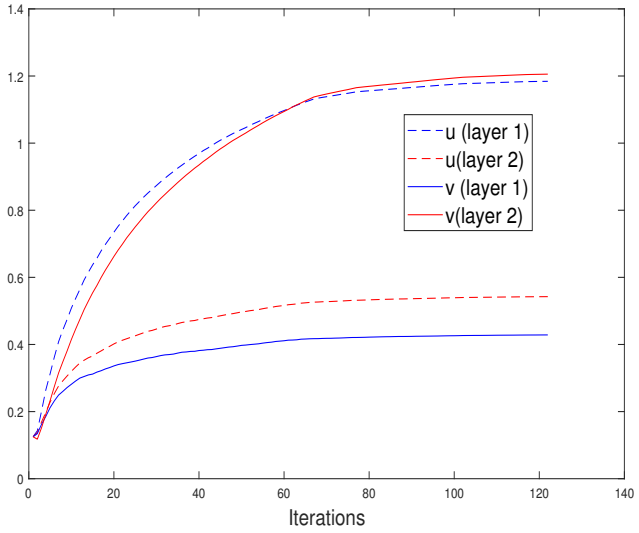

Figure 4. Weights u and v variations during the training process for the dataset T153

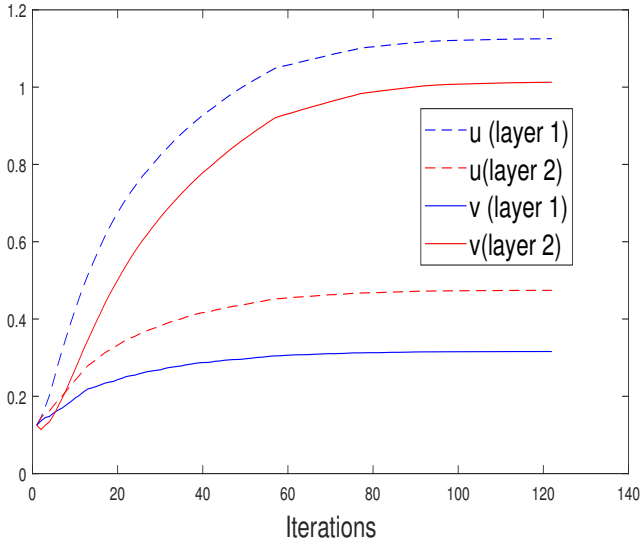

Figure 5. Weights u and v variations during the training process for the dataset T160

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 November 2020 doi:10.20944/preprints202011.0511.v1

References

1. Ma, J.; Gao, W.; Wei, Z.; Lu, Y.; Wong, K.F. Detect rumors using time series of social context information on microblogging websites. Proceedings of the 24th ACM International on Conference on Information and Knowledge Management. ACM, 2015, number 3, pp. 1751–1754.
2. Ma, J.; Gao, W.; Wong, K.F. Detect rumors in microblog posts using propagation structure via kernel learning. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, 2017, Vol. 1, p. 708–717.
3. Ma, J.; Gao, W.; Mitra, P.; Kwon, S.; Jansen, B.J.; Wong, K.F.; Cha, M. Detecting rumors from microblogs with recurrent neural networks. Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, 2016, p. 3818–3824.
4. Qi, H.; Chuan, Z.; Wu, J.; Wang, M.; Wang, B. Deep Structure Learning for Rumor Detection on Twitter. 2019, pp. 1–8. doi:10.1109/IJCNN.2019.8852468.
5. Yuan, C.; Ma, Q.; Zhou, W.; Han, J.; Hu, S. Jointly embedding the local and global relations of heterogeneous graph for rumor detection. The 19th IEEE International Conference on Data Mining. IEEE, 2019.
6. Bian, T.; Xiao, X.; Xu, T.; Zhao, P.; Huang, W.; Rong, Y.; Huang, J. Rumor Detection on Social Media with Bi-Directional Graph Convolutional Networks. The Thirty-Fourth AAAI Conference on Artificial Intelligence, 2020, pp. 549–556.
7. Castillo, C.; Mendoza, M.; Poblete, B. Information credibility on twitter. Proceedings of the 20th international conference on World wide web. ACM, 2011, p. 675–684.
8. Ma, J.; Gao, W.; Wong, K.F. Rumor Detection on Twitter with Tree-structured Recursive Neural Networks. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, 2018, p. 1980–1989.
9. Ma, J.; Gao, W.; Wong, K.F. Detect rumors on twitter by promoting information campaigns with generative adversarial learning. The World Wide Web Conference. ACM, 2019, p. 3049–3055.
10. Yu, F.; Liu, Q.; Wu, S.; Wang, L.; Tan, T. A convolutional approach for misinformation identification. Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, 2017, p. 3901–3907.
11. Ruchansky, N.; Seo, S.; Liu, Y. Csi: A hybrid deep model for fake news detection. Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, p. 797–806.
12. Liu, Y.; Wu, Y.F.B. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. Thirty-Second AAAI Conference on Artificial Intelligence, 2018, p. 354–361.
13. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. Proc. Int. Conf. Learn. Represent, 2017.
14. Perozzi, B.; Al-Rfou, R.; Skiena, S. Deepwalk: Online learning of social representations. Proceedings of SIGKDD, 2014, p. 701–710.
15. Yang, C.; Liu, Z.; Zhao, D.; Sun, M.; Chang, E.Y. Network Representation Learning with Rich Text Information. Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, 2015, p. 2111–2117.
16. Zhang, D.; Yin, J.; Zhu, X.; Zhang, C. Network Representation Learning: A Survey. *IEEE Transactions on Big Data* **2020**, *6*, 3–28.
17. Ming Chen, Z.W.; Zengfeng Huang, B.D.; Li, Y. Simple and Deep Graph Convolutional Networks. Proceedings of the 37th International Conference on Machine Learning, 2020.