Predicting Future Participants of Information Propagation Trees

Hsing-Huan Chung
josh860826@gmail.com
National Taiwan University
Taipei, Taiwan

Hen-Hsen Huang
hhhuang@nccu.edu.tw
National Chengchi University
MOST Joint Research Center for AI Technology and All Vista Healthcare
Taipei, Taiwan

Hsin-Hsi Chen
hhchen@ntu.edu.tw
National Taiwan University
MOST Joint Research Center for AI Technology and All Vista Healthcare
Taipei, Taiwan

ABSTRACT

Understanding how information propagates among social media users can allow researchers to provide interesting insights into online social networks and lead to applications such as precise advertising and misinformation management. In this work, we focus on information diffusion through post sharing. Given an information propagation tree, our goal is to predict a list of potential users of the tree. A framework based on graph convolutional network (GCN) is proposed to learn the latent representation of a propagation tree and match it with the latent representation of a user. A novel strategy for tree pruning is further investigated to improve the GCN. Experimental results show that our framework outperforms the existing methods for modeling information diffusion.

CCS CONCEPTS

• **Information systems** → *Collaborative filtering*; *Social recommendation*; *Social networks*; • **Human-centered computing** → *Social content sharing*; *Social media*; • **Computing methodologies** → *Neural networks.*

KEYWORDS

social media, information diffusion, graph convolutional network

ACM Reference Format:
Hsing-Huan Chung, Hen-Hsen Huang, and Hsin-Hsi Chen. 2019. Predicting Future Participants of Information Propagation Trees. In *IEEE/WIC/ACM International Conference on Web Intelligence (WI '19), October 14–17, 2019, Thessaloniki, Greece.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3350546.3352540

1 INTRODUCTION

Social media attracts hundreds of million users to publish, share, and consume information. The easiness of producing information leads to a rapid increase in online content. Moreover, social media has shortened the communication distance between people, which results in frequent interactions such as sharing posts, replying and direct messaging. These factors make social media a platform for extensive information diffusion.

In this work, we are interested in modeling how a post is shared and propagates among social media users. Early work models information diffusion with given Independent Cascades (IC) or Linear Threshold (LT) model [7]. Based on IC or LT model, several works introduce machine learning techniques to learn the diffusion model from cascaded data [2, 4]. These methods rely on hand-crafted features, which require extensive domain knowledge and are hard to generalize to all platforms. In order to skip the troublesome feature engineering process, recent works adopt deep learning and representation learning techniques to model information diffusion [1, 3, 6, 9, 10, 12, 13, 16]. The tasks and utilized information in these works vary. In [1, 10], they both regard the diffusing process as a sequence of events where each event indicates a user getting "infected" at a certain time point. In [13, 16], their models not only take a sequence of events as input but also utilize the structure of social graphs to achieve better performance. However, their models do not make use of the sharing relationship among users. That is, they only know a node in a social graph is infected at a certain time point but do not know which node infects it or which node it shares information from. By contrast, the models proposed by [3, 6, 9, 12] take the sharing relationship among users as input instead of social graphs. The setting of our work is close to [3, 6, 9, 12]. Our framework only has access to the sharing relationship among users and does not need the information of social graphs. Having a similar setting with [6, 12], the sharing relationship we consider can be modeled as a tree structure called *propagation tree*. The root node of a propagation tree represents the user who publishes the source post. Each edge between a parent node and a child node represents a sharing action. As the information goes viral, the propagation tree gets larger. Figure 1 illustrates a propagation tree.

Our goal in this work is to predict which users will participate in a propagation tree in the future. This task is similar to a recommendation problem: Given a propagation tree, recommend a list of users who may be interested in participating in it, or in other words, spread the information that the tree carries. According to the setting of traditional collaborative filtering, a tree can be characterized by the set of users who have participated in it. In our case, however, a tree is not just a set of users. There exists a rich sharing relationship among users within a tree. Therefore, we learn user and tree latent representations that take the sharing relationship into account. Then, we match a user representation with a tree representation to estimate the probability of the user participating in the tree in the future.

The main innovation of this work lies in learning tree-level representations. We evaluate our framework on two real-world datasets and compare it with existing methods. Experimental results show

WI '19, October 14–17, 2019, Thessaloniki, Greece
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6934-3/19/10…$15.00
https://doi.org/10.1145/3350546.3352540

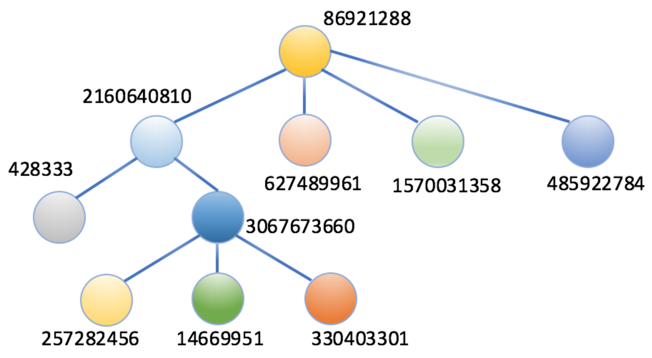

Figure 1: A simple example of an information propagation tree where each node has a user ID. The root node having user ID 86921288 represents the user who publishes the source post of the propagation tree.

that our framework outperforms other methods. The advantages of our framework are threefold:

(1) Our model simultaneously performs top-down and bottom-up representation aggregation which models our intuition that sharing is a form of bidirectional information exchange.
(2) We propose a novel tree pruning strategy that is able to eliminate noise.
(3) The final tree representation is tailored to each user who is matching with the tree, which models the phenomenon that each user sees an information diffusing process differently.

The problem formulation, proposed approach, and experiments will be presented in the rest of this paper.

2 PROBLEM FORMULATION

Let $\mathcal{T} = \{T_1, T_2, T_3, \dots T_{|\mathcal{T}|}\}$ and $\mathcal{U} = \{u_1, u_2, u_3, \dots u_{|\mathcal{U}|}\}$ be the sets of $|\mathcal{T}|$ propagation trees and $|\mathcal{U}|$ users. A propagation tree is composed of nodes and edges where each node represents a user and each edge represents a sharing action. We consider a propagation tree an undirected, acyclic, and connected graph. Formally, $T_i = (U_i, E_i)$ where $U_i \subseteq \mathcal{U}$ and $E_i \subseteq U_i \times U_i$. U_i and E_i each corresponds to the set of nodes and the set of edges in T_i. $(u_p, u_c) \in E_i$, i.e., u_c is the child of u_p in T_i if and only if u_c shares the post published by u_p under T_i. Hereafter, we refer to an arbitrary user or an arbitrary node in a propagation tree interchangeably using the notation u. Likewise, an arbitrary tree is denoted as T.

Our problem is formulated as follows: Given a propagation tree T_i, we aim to generate a ranked list of users $\{u_m, u_n, u_l, \dots\}$ who may participate in the propagation tree in the future.

3 PROPOSED APPROACH

To determine whether a user u_m will participate in a propagation tree T_i, we compute representations of T_i and u_m, $\vec{T}_i$ and $\vec{u}_m$, then match them to obtain a score $\hat{y}_{im}$ which is between 0 and 1. Our framework can be decomposed to several components. First, we pre-train user representations utilizing the global interactions between users. Next, we compute the tree representation $\vec{T}_i$ making use of local interactions between users within T_i. Finally, we match $\vec{T}_i$ with $\vec{u}_m$ to get $\hat{y}_{im}$.

The ground truth score is denoted as y_{im} where $y_{im} = 1$ if u_m will participate in T_i, otherwise 0. Our goal during training is to minimize the cross entropy loss:

$$\mathcal{L} = - \sum_{(i,m) \in \mathcal{S}^+ \cup \mathcal{S}^-} y_{im} \log \hat{y}_{im} + (1 - y_{im}) \log(1 - \hat{y}_{im}) \quad (1)$$

where $\mathcal{S}^+$ and $\mathcal{S}^-$ are the sets of positive training samples and negative training samples, respectively. In the following subsections, we will go through the details of each component.

3.1 Pre-training User Embeddings

We first construct a network of users from all propagation trees. Let the global user network $\mathcal{N} = (\mathcal{U}, \mathcal{E})$ be a weighted and undirected graph. The set of nodes $\mathcal{U}$ represents of all users. $\mathcal{E} \subseteq \mathcal{U} \times \mathcal{U}$ denotes the edges of the user network. There is an edge (u_p, u_c) between u_p and u_c in $\mathcal{N}$ if one of them has shared the other one's post. The weight of (u_p, u_c) is the frequency of u_p and u_c sharing each other's posts.

Afterwards, we use Large-Scale Information Network Embedding (LINE) [14] to take $\mathcal{N}$ as the input graph and pre-train low dimensional user embeddings $\{\vec{u}_1, \vec{u}_2, \dots\}$.

3.2 Graph Convolutional Network

After obtaining the pre-trained user embeddings, we compute the contextual node embeddings within each tree. To gather contextual information for nodes in a propagation tree, we make them exchange information along the tree edges.

Tree edges represent the sharing actions and there are two interpretations of sharing. Firstly, sharing is an action of information diffusion from a post publisher to a sharer. Secondly, sharing is feedback from a sharer, showing her interest in a post and its publisher. Therefore, we claim that sharing is a form of bidirectional information exchange.

The purpose of considering a propagation tree T an undirected graph is to satisfy the above setting. Given the adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ of T and its initial node representation matrix $\mathbf{U}_0 \in \mathbb{R}^{n \times h_0}$ where each row is a pre-trained user embedding $\vec{u}^\top$ of a tree participant of T. A 2-layer graph convolutional network[8] can be written as:

$$\mathbf{U}_2 = \tanh(\mathbf{D}^{-1}\tilde{\mathbf{A}} \tanh(\mathbf{D}^{-1}\tilde{\mathbf{A}}\mathbf{U}_0\mathbf{W}_1)\mathbf{W}_2) \quad (2)$$

where $\mathbf{W}_1 \in \mathbb{R}^{h_0 \times h_1}$ and $\mathbf{W}_2 \in \mathbb{R}^{h_1 \times h_2}$ are two linear transformations containing graph convolution parameters, $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the adjacency matrix of the given graph with each node having an added self loop, $\mathbf{D}$ is a diagonal matrix where $\mathbf{D}_{ii} = \sum_{j=1}^{n} \mathbf{A}_{ij}$. The output of this component, $\mathbf{U}_2$, contains the contexual node embeddings within a tree.

3.3 Tree Pruning and Attention Mechanism

In this subsection, we will describe how we compute the representation of a propagation tree based on the outputs of previous components. The inputs of this component are listed as follows.

(1) $T_i = (U_i, E_i)$, a propagation tree
(2) u_m, the user who is going to match with T_i

(3) $\mathbf{U}_2^i$, the node representations within T_i after 2 graph convolutions
(4) $\vec{u}_m$, the pre-trained embedding of u_m
(5) $\mathcal{N} = (\mathcal{U}, \mathcal{E})$, the global user network.

First of all, we apply a tree pruning strategy on the input propagation tree. The goal of this operation is to select representative nodes and eliminate the rest to obtain better tree representation. Our tree pruning strategy is based on two intuitions. Firstly, we select nodes that have strong influences on the process of information diffusion. Secondly, we select nodes that are close to u_m because they are very likely to be the "entrances" for u_m to participate in the propagation tree.

We define a node as highly influential in the information diffusing process if the user it represents is an opinion leader or an Internet celebrity. Such kind of users will make the information continue diffusing because they would bring more people's attention to it. In other words, an information diffusion path will not end in highly influential nodes. Therefore, we define highly influential nodes as those internal nodes, U_{int}.

Afterwards, we are going to select nodes that are close to u_m. We define two nodes being close if they belong to the same community or even have interacted before. We construct an undirected graph $T_i' = (U_i', E_i')$ where

$$U_i' = U_i \cup u_m \tag{3}$$

$$E_i' = E_i \cup \{(u, u_m) \mid u \in U_i \wedge (u, u_m) \in \mathcal{E}\} \tag{4}$$

u_m, which doesn't belong to T_i, is currently connected to the nodes in T_i that are neighbors of u_m in the global user network $\mathcal{N}$. Afterwards, we apply breadth-first search (BFS) from u_m to search and select its k-hop neighbors. If $k = 1$, only those that have directly interacted with u_m are selected. By keep searching to a larger k, more node information will be considered, but more noise might be included as well. After finishing BFS, the nodes that are close to u_m are selected. We call this set of nodes U_{bfs}. Since U_{bfs} is dependent on u_m, the final tree representation will vary according to the user who is going to match with the tree. This fits the reality because a user does not have a god's-eye view of an entire information diffusing process. Instead, how a user sees a piece of information is influenced by the community she belongs to.

As shown in Figure 2, we can obtain a pruned propagation tree $T_{ip} = (U_{ip}, E_{ip})$ by removing nodes that belong to $U_i - (U_{int} \cup U_{bfs})$ and edges attached to them. Likewise, we can get a pruned node representation matrix $\mathbf{U}_{2p}^i \in \mathbb{R}^{|U_{ip}| \times h_2}$ by removing representations of pruned nodes.

Next, we apply an attention mechanism to compute the final tree representation. The attention function we adopt is scaled dot-product attention [15]:

$$\vec{T}_i = Attention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = Softmax(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{h_2}})\mathbf{V} \tag{5}$$

where $\mathbf{K}$ and $\mathbf{V}$ are both $\mathbf{U}_{2p}^i$, $\mathbf{Q}$ is $\vec{u}_m^\top$. Note that h_2 must be equal to the dimension size of pre-trained user embeddings so that the inner product of a node representation and a user embedding can be performed.

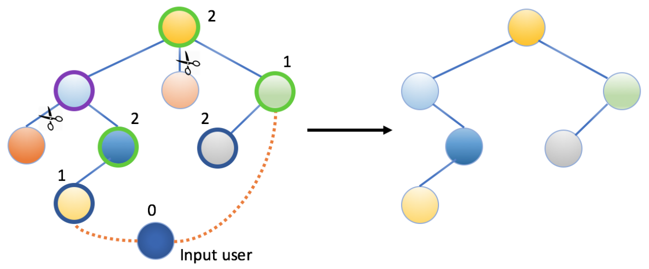

Figure 2: An example of tree pruning with the BFS parameter k set to 2. In the left tree, the node with purple edge belongs to U_{int}, the nodes with dark blue edges belong to U_{bfs}, and the nodes with light green edges belong to both U_{bfs} and U_{int}. The rest of the nodes are removed during the pruning process.

3.4 Matching

After finish computing $\vec{T}_i$, we will finally match it with $\vec{u}_m$ to obtain $\hat{y}_{im}$. We adopt neural matrix factorization (NeuMF) [5], which is a fusion of generalized matrix factorization (GMF) and multiple layer perceptron (MLP), to compute $\hat{y}_{im}$.

To provide a ranked list of users for T_i, our framework matches $\vec{T}_i$ with the representation of each user in a candidate pool. Afterwards, our framework selects the users with the highest probabilities and sorts them to form a ranked list.

4 EXPERIMENTS

4.1 Datasets

In the following experiments, we used the datasets **Twitter15** and **Twitter16**, provided by [11]. These datasets were used for rumor detection in previous works [11, 12]. To the best of our knowledge, we are the first to use these datasets for future participant prediction.

4.2 Comparison Methods

We used different methods to model the information diffusing process and compared their performances. The methods include **random**, **GRU-RNN** [10], **BU-RvNN** [12], **TD-RvNN** [12] and our method, **GCN-Prune-Attention**. On the other hand, we tested our method with several variations, including the model without tree pruning and the model with a different hyperparameter k for BFS.

4.3 Experimental Settings

For each dataset, we selected 85% of trees as the training set and the rest as the test set. For each tree, we ordered the nodes chronologically and selected the last 20% of nodes as future participants to predict.

Aside from random, all methods take the same set of pre-trained user embeddings as input and use NeuMF as the matching function. The difference between these methods lies in diffusion process modeling.

In our experiments, we limited the participant candidate pool of a tree to the set of users who have interacted with the tree participants in other trees, or in other words, the neighbors of all tree participants in the global user network $\mathcal{N}$. Using all users as the candidate pool leads to slow convergence during training and we leave efficient sampling a less limited candidate pool to future work. In most social media platforms, a user's feed only consists of posts published by other users she follows and is optimized to show her posts she may be interested at the top of it. Since we don't have social graph information, we tested with the same candidate pool as the training one to approximate the application scenario.

During training with a tree, we sampled negative instances from the candidate pool uniformly with a negative ratio of one. Although [5] suggests to sample more negative instances, both BU-RvNN and TD-RvNN take too much time to train because they are hard to compute in parallel. Hence, we keep negative instances as many as positive instances in order to save time.

During testing with a tree, we sampled negative instances from the tree's candidate pool along with given positive instances to provide 100 testing users. Performance is evaluated by precision and mean average precision (mAP) at the ranked list.

The hyperparameter settings of our model are described as follows. The dimensions of pre-trained embeddings are 128 for Twitter15 and 64 in Twitter16, respectively. Both h_1 and h_2 are equal to the initial embedding dimensions. The multiple layer perceptron in NeuMF consists of three hidden layers. The number of neurons is 128 in the first layer, 64 in the second layer, and 32 in the third layer. The learning rate was set to 0.001, and the parameters were optimized with the ADAM algorithm.

4.4 Experimental Results

Table 1 shows the experimental results on Twitter15 and Twitter16. In both datasets, Random performed the worst and GCN-Prune-Attention performed the best among all methods. TD-RvNN performed better than BU-RvNN in both datasets. This matches the discovery of a previous work[12].

On the other hand, TD-RvNN and GCN-based methods performed better than GRU-RNN does because, during the preprocessing step of GRU-RNN, the structural information of a propagation tree is lost. The setting discards the sharing relationship and makes GRU-RNN less powerful than TD-RvNN and GCN-based methods, which both fully preserve the propagation tree structure during neural computations.

With respect to the performances of two competitive structure-preserving models, GCN-based methods outperformed TD-RvNN. TD-RvNN has a property that the amount of information a node representation contains depends on its ancestors and is limited by the tree height. One possible reason that TD-RvNN underperformed GCN-based methods is that an information propagation tree is a rather wide and shallow structure. There are more than 400 nodes in a tree in both datasets, while the average height of a tree is only 6.5 in Twitter15 and 6.9 in Twitter16, respectively. The shallow tree structure limits the effectiveness of TD-RvNN. Furthermore, a node in GCN can gather information from its siblings since its representation depends on its first and second-hop neighbors. Information of siblings is important in the context of information diffusion because siblings, who share information from the same tweeter (parent), may share similar interests. Nevertheless, RvNN models, where a node representation cannot take its siblings into account, are inferior to GCN in modeling wide tree structures. Another reason that may contribute to the superiority of GCN over RvNN models is the property of GCN being able to model the bidirectional information exchange between users. On the other side, RvNN models only consider unidirectional information propagation.

Table 1: Results on Twitter15 and Twitter16

	Twitter15		Twitter16	
Method	prec@10	mAP@10	prec@10	mAP@10
Random	0.237	0.359	0.245	0.372
GRU-RNN	0.393	0.556	0.389	0.549
BU-RvNN	0.315	0.449	0.331	0.486
TD-RvNN	0.421	0.606	0.404	0.563
GCN-Attention	0.431	0.562	0.431	0.580
GCN-Prune-Attention	**0.478**	**0.642**	**0.475**	**0.674**

Table 2: GCN-Prune-Attention with different k

	Twitter15		Twitter16	
k	prec@10	mAP@10	prec@10	mAP@10
0	0.425	0.578	0.419	0.579
1	0.425	0.558	0.418	0.592
2	0.424	0.590	0.423	0.548
3	**0.478**	**0.642**	**0.475**	**0.674**
∞	0.431	0.562	0.431	0.580

Table 2 shows how the performances of GCN-based methods change as the BFS parameter k varies. If $k = 0$, then the pruned tree only contains nodes in U_{int}. If $k = \infty$, no nodes are pruned. As both table shows, the performance peaks when $k = 3$. Using a k smaller than 3 leads to overly aggressive pruning that discards much useful information. On the other hand, not pruning at all results in too much noise, which degrades the performance. These experimental results confirm the effectiveness of our tree pruning strategy.

5 CONCLUSION

We propose a framework for predicting future participants of an information propagation tree. By properly utilizing the tree structure and characteristics of information diffusion, our framework outperformed existing methods on conducted experiments. Although we experimented with Twitter datasets, our framework is suitable for all social media platforms as long as their information diffuses as a tree structure. Future works will focus on several possible extensions: (1) Incorporate content information and temporal information into our framework; (2) Efficient sampling a less limited candidate pool.

6 ACKNOWLEDGMENTS

This research was partially supported by Academia Sinica under grant AS-TP-107-M05 and Ministry of Science and Technology, Taiwan, under grants MOST-106-2923-E-002-012-MY3, MOST-108-2634-F-002-008-, and MOST-108-2218-E-009-051-.

REFERENCES

[1] Simon Bourigault, Sylvain Lamprier, and Patrick Gallinari. 2016. Representation Learning for Information Diffusion Through Social Networks: An Embedded Cascade Model. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining (WSDM '16)*. ACM, New York, NY, USA, 573–582. https://doi.org/10.1145/2835776.2835817

[2] Wojciech Galuba, Karl Aberer, Dipanjan Chakraborty, Zoran Despotovic, and Wolfgang Kellerer. 2010. Outtweeting the Twitterers - Predicting Information Cascades in Microblogs. In *Proceedings of the 3rd Wonference on Online Social Networks (WOSN'10)*. USENIX Association, Berkeley, CA, USA, 3–3. http://dl.acm.org/citation.cfm?id=1863190.1863193

[3] S. Gao, H. Pang, P. Gallinari, J. Guo, and N. Kato. 2017. A Novel Embedding Method for Information Diffusion Prediction in Social Network Big Data. *IEEE Transactions on Industrial Informatics* 13, 4 (Aug 2017), 2097–2105. https://doi.org/10.1109/TII.2017.2684160

[4] Adrien Guille and Hakim Hacid. 2012. A Predictive Model for the Temporal Dynamics of Information Diffusion in Online Social Networks. In *Proceedings of the 21st International Conference on World Wide Web (WWW '12 Companion)*. ACM, New York, NY, USA, 1145–1152. https://doi.org/10.1145/2187980.2188254

[5] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 173–182. https://doi.org/10.1145/3038912.3052569

[6] Wenjian Hu, Krishna Kumar Singh, Fanyi Xiao, Jinyoung Han, Chen-Nee Chuah, and Yong Jae Lee. 2018. Who Will Share My Image?: Predicting the Content Diffusion Path in Online Social Networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 252–260. https://doi.org/10.1145/3159652.3159705

[7] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the Spread of Influence Through a Social Network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03)*. ACM, New York, NY, USA, 137–146. https://doi.org/10.1145/956750.956769

[8] Thomas N. Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *CoRR* abs/1609.02907 (2016). arXiv:1609.02907 http://arxiv.org/abs/1609.02907

[9] Cheng Li, Jiaqi Ma, Xiaoxiao Guo, and Qiaozhu Mei. 2017. DeepCas: An End-to-end Predictor of Information Cascades. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 577–586. https://doi.org/10.1145/3038912.3052643

[10] Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J. Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting Rumors from Microblogs with Recurrent Neural Networks. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI'16)*. AAAI Press, 3818–3824. http://dl.acm.org/citation.cfm?id=3061053.3061153

[11] Jing Ma, Wei Gao, and Kam-Fai Wong. 2017. Detect Rumors in Microblog Posts Using Propagation Structure via Kernel Learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, 708–717. https://doi.org/10.18653/v1/P17-1066

[12] Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Rumor Detection on Twitter with Tree-structured Recursive Neural Networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, 1980–1989. https://www.aclweb.org/anthology/P18-1184

[13] Jiezhong Qiu, Jian Tang, Hao Ma, Yuxiao Dong, Kuansan Wang, and Jie Tang. 2018. DeepInf: Social Influence Prediction with Deep Learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &#38; Data Mining (KDD '18)*. ACM, New York, NY, USA, 2110–2119. https://doi.org/10.1145/3219819.3220077

[14] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1067–1077. https://doi.org/10.1145/2736277.2741093

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 5998–6008. http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

[16] J. Wang, V. W. Zheng, Z. Liu, and K. C. Chang. 2017. Topological Recurrent Neural Network for Diffusion Prediction. In *2017 IEEE International Conference on Data Mining (ICDM)*. 475–484. https://doi.org/10.1109/ICDM.2017.57