SRACas: A Social Role-Aware Graph Neural Network-Based Model for Popularity Prediction of Information Cascades

Zhenhua Huang[1], Yuhang He[2], Shaojie Wang[1], Zhenyu Wang[3], Ruifeng Xu[2(✉)], and Sharad Merothra[4]

[1] Anhui University, Hefei, China
wsj.ahu@gmail.com
[2] Harbin Institute of Technology (Shenzhen), Shenzhen, China
xuruifeng@hit.edu.cn
[3] South China University of Technology, Guangzhou, China
[4] University of California Irvine, Irvine, USA

Abstract. Popularity prediction of information cascades is a fundamental and challenging task in social network data analysis. Social roles impact users' behaviors and change the structure and popularity of information cascades. Existing deep learning-based methods utilize several independent sub-cascade graphs or paths to learn cascade representations, which lose vital information about social roles and dynamics between sub-cascades at different moments. We propose a social role-aware cascade (SRACas) model that exploits the social influences of nodes on previous and subsequent sub-cascade graphs within an observation window to facilitate the social role learning of nodes. A temporal-aware differential loss is also proposed to discriminate the structures of neighboring sub-cascades and captures the dynamics of sub-cascades. Under the techniques of local graph attention, social role-aware attention, and temporal-aware loss, SRACas learns a better latent representation of cascades at both the node level, sub-cascade level, and cascade level. Moreover, there lacks a platform with standard prepossessing procedures that allow convenient configuration and fair competition between information cascade prediction models. An open platform OpenCas is built with uniform preprocesses to verify the faithful performance of the compared methods. Extensive experiments show that SRACas achieved significant improvements over existing methods on classic real-world datasets.

Keywords: Information Cascade · Graph Neural Networks · Social Role · Popularity Prediction

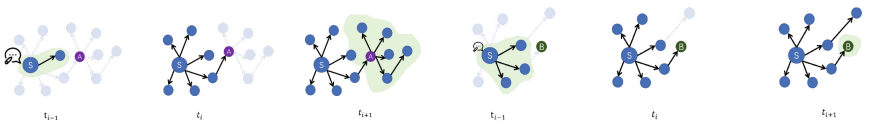
Z. Huang and Y. He—Equal Contribution.

X. Wang et al. (Eds.): DASFAA 2023, LNCS 13945, pp. 21–30, 2023.
https://doi.org/10.1007/978-3-031-30675-4_2

1 Introduction

Popularity prediction of information cascades (also called cascade prediction in [3,12]) benefits many practical applications, e.g., fake news and rumor control, viral marketing, advertising, scientific impact quantification [11]. Understanding the nature of cascades and predicting their future popularity has drawn the interest of many scholars. However, cascade prediction remains challenging due to complex social influences, and its accuracy is still unsatisfactory.

The deep learning-based approaches have recently achieved state-of-the-art performances in information cascade prediction [2,3,6,8,9,12]. DeepCas [6], DeepHawkes [2], and TopoLSTM [8] are mainly based on recurrent neural networks. Spatial features are also found helpful in improving the performance of popularity prediction and are learned by the graph neural networks (GNNs) in current works [3,9,12]. Social roles impact retweet behaviors and significantly influence information diffusion [10]. However, current methods, e.g., CasCN [3], VaCas [12] and CasFlow [9] learn node features in isolated sub-cascade graphs, ignoring the social roles of nodes and influences between nodes in different sub-cascades. As described in Fig. 1, the sub-cascade graphs at t_i in two cases are topologically the same. But the node A and B hold different social roles and influences in the cases. A is an opinion leader that leads several following retweets, while B is a common node. Node representations learned independently in a sub-cascade graph can hardly distinguish the situations and reflect the nodes' social roles. However, by reviewing previous and subsequent sub-cascades, we can easily discriminate the social roles of node A and B and know their impacts on information diffusion. As represented in t_i and t_{i+1} in Fig. 1(b), the structure and size of cascades do not always change drastically over time. The adjacent sub-cascades may be slightly different from the previous sub-cascades. The existing models rarely consider the subtle differences between adjacent sub-cascade graphs and lose subtle dynamic information of cascades. Moreover, current models based on dense adjacency matrices are memory-intensive and inefficient or perform much worse in larger cascades.



(a) Case 1 (b) Case 2

Fig. 1. Cases describing how social roles impact information diffusion. The purple node A is an opinion leader, and green node B is a common node. (Color figure online)

To address the abovementioned problems of cascade prediction, we propose a novel social role-aware cascade prediction model named SRACas. It utilizes local graph attention and a social role-aware attention mechanism which enables

each node to consider both its neighboring nodes' representation and features of sub-cascade graphs from previous and subsequent moments. The social roles and influences are embedded in nodes' representations in this way. A temporal-aware differential loss is also employed to learn the subtle differences between sub-cascade graphs, which benefits the training and convergence of models. To support larger cascades, sparse matrices and efficient computations are also applied.

In addition, although there is a survey on cascade prediction [11], a framework aggregating current cascade models is lacking. It is necessary to unify experimental steps before training models and reorganize the implementations of current models. Therefore, a framework, OpenCas, is built to address the issue.

Our main contributions are as follows:

- ***Novel Cascade Model.*** We propose a novel cascade model, SRACas, which employs local graph attention and a social role-aware attention mechanism to learn better social role representations of nodes that leverages bidirectional sub-cascades. A temporal-aware differential loss is also applied in SRACas to capture the subtle differences and dynamics of sub-cascade graphs, which is another key to learning better temporal features of cascades.
- ***Support Larger Cascades.*** Some advanced cascade prediction models (e.g., CasCN [3], VaCas [12], and CasFlow [9]) are unable to work or perform much worse in larger cascades. In contrast, the performance of our approach is most stable in larger cascades leveraging graph sparse encoding.
- ***Much Better Performance.*** Extensive experiments on two real-world scenarios demonstrate that SRACas significantly outperforms strong baselines, with the MSLE reduced by from 12.0% to 14.6%, from 6.9% to 9.7% in the dataset Sina Weibo and APS Citation, respectively.
- ***OpenCas.*** We built a cascade prediction framework OpenCas, which enables convenient and fair comparative experiments between different models under the same data preprocessing and partitions. OpenCas improves the performance of classic models and simplifies environment configurations[1]

2 Preliminaries

Cascade Graph. A source of a cascade can be an academic publication, a tweet, a microblog, etc. A cascade graph can be represented as a sub-cascade graph sequence that evolves from an initial source node v_0 at time t_0. Node v_i participates in the cascade at time t_i. A subcascade $\mathcal{G}^{t_j}$ is a graph at moment t_j. We formalize the cascade sequence as $C_i = \{\mathcal{G}^{t_0}, \mathcal{G}^{t_1}, ..., \mathcal{G}^{t_n}\}$, where $\mathcal{G}^{t_j} = (V^{t_j}, E^{t_j}, t_j)$ is a snapshot graph of the cascade C at time t_j. V^{t_j} and E^{t_j} are the sets of nodes and edges of sub-cascade $\mathcal{G}^{t_j}$ until time $t_j \geq 0$, respectively. $V^{t_j} = \{v_0(t_j), v_1(t_j), v_2(t_j), ..., v_i(t_j)\}$. The set of edges E^{t_j} records how information propagates between users in V^{t_j}.

Popularity Prediction. Following previous works [2,3,6,12], the popularity prediction of information cascades is formalized as a regression problem. Given

[1] The details of OpenCas are referred to https://github.com/zhenhuascut/OpenCas.

a cascade $C_i = \{\mathcal{G}^{t_0}, \mathcal{G}^{t_1}, ..., \mathcal{G}^{t_n}\}$, the incremental popularity is defined as $\Delta S = |V^{t_p}| - |V^{t_o}|$, where t_o and t_p are the observation time and the prediction time, respectively. The main objective is to train a model f that learns the representation of a cascade within the observation time t_o to predict ΔS.

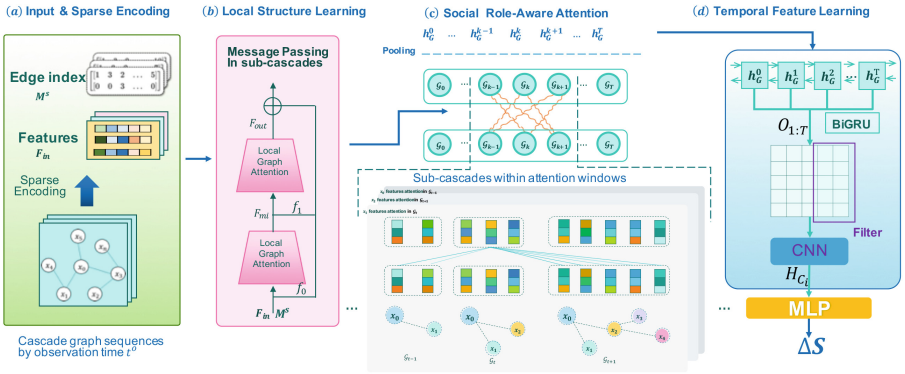


Fig. 2. Overview of SRACas.

3 Proposed Method

The overall training process of SRACas is sketched in Fig. 2. It mainly contains the following modules:

3.1 Local Structure Learning

Given a sub-cascade graph $\mathcal{G}^t$, the first step is to capture the basic structural information and obtain node-level representations. Different from CasCN [3] that neighboring nodes have similar weights. We believe that different nodes play different roles in information diffusion, e.g., opinion leaders, structural holes, and peripheral nodes. Local graph attention similar to GAT [7] is used to calculate the importance of a node and aggregate the features of nodes. The α_{uv} and alignment coefficients e^l_{uv} are calculated by:

$$\alpha_{uv} = Softmax_u(e^l_{uv}) = \frac{exp(e^l_{uv})}{\sum_{k \in N(v) \cup \{v\}} exp(e^l_{kv})} \tag{1}$$

$$e^l_{uv} = attn^l(x^l_u, x^l_v) = f_{prop}(h^l_u, h^l_v) \tag{2}$$

where h^l_u is the node embedding in graph aggregation layer l. The function f_{prop} can be in different forms. One simple form is $f_{prop} = (W^l_a)^T[W^l h^l_u, W^l h^l_v]$, where $W^l_a \in R^{2F^{l+1}}$ is a shared attention weights. x^t_v is calculated by a residual network $x^t_v = f_0(h^0_v) + f_1(h^1_v) + h^L_v$. $x^t_v \in R^{F^L}$, f_0 and f_1 are dense layers that change dimension size of h^l_v to F^L. L is the number of layers set to two in this paper.

3.2 Social Role-Aware Attention

The social role-aware attention considers the influences of nodes from nodes of previous and following sub-cascades within $\mathcal{W}$ temporal intervals (attention window), in which the roles a node plays in information diffusion are learned.

The social influence of a node v by node $u \in V^{t-1}$ at the previous moment is calculated by :

$$\alpha_{uv}^{b} = \frac{exp(f_{back}(x_u^{t-1}, x_v^t))}{\sum_{k \in V^{t-1}} exp(f_{back}(x_k^{t-1}, x_v^t))} \tag{3}$$

$$x_v^{t-1} = \sigma(\sum_{u \in V^{t-1}} x_u^{t-1} \alpha_{uv}^{b}) \tag{4}$$

where α_{uv}^{b} is the impacts of node u to node v in V^{t-1}. f_{back} is the aggregation method between x_v^{t-1} and x_v^t. Here, we choose the Euclidean distance of nodes as the function f_{back}.

x_v^{t+1} is calculated in the same way as in x_v^{t-1}. The updated representation x_v^{update} of node v is then calculated by combing the representations of v from sub-cascades within attention window size $\mathcal{W}$. When $\mathcal{W} = 1$, $f_{combine}(x_v^{t-1}, x_v^t, x_v^{t+1}) = x_v^{t-1} + x_v^t + x_v^{t+1}$. $f_{combine}$ can also be designed in a more complicated form, which is left for future works. A larger attention window size (e.g., $\mathcal{W} = 2$, $f_{combine}(x_v^{t-2}, x_v^{t-1}, x_v^t, x_v^{t+1}, x_v^{t+2})$) brings a broader view of sub-cascades but increases the calculation complexity.

Graph Pool. After obtaining nodes' representations, the features of nodes at the moment are pooled to produce a presentation of the sub-cascade graph h_G^t.

3.3 Temporal Feature Learning

Social role-aware attention has learned part of cascades' temporal characteristics, as well as the features of social roles. To learn the long-term temporal dependency of cascades, we apply a temporal feature learning model. Given input representations of sub-cascades $H_G = \{h_G^0, h_G^1, h_G^2, ..., h_G^T\}$, the bidirectional gated recurrent units (BiGRU) computes updated hidden states and produces new representations by concatenating outputs of the forward GRU and backward GRU: $O_{1:T} = \{o_0, o_1, o_2, ..., o_T\} = \overrightarrow{GRU}(H_G)||\overleftarrow{GRU}(H_G)$, $O_{1:T} \in R^{T*2M}$, M is the output hidden size of BiGRU.

We can simply use the last output o_T as the embedding features of a cascade. Motivated by the idea [5], convolutional neural networks are used to capture the long-term dependencies of H_G and produce the final representation H_{C_i} of a cascade C_i.

3.4 Prediction and Loss Function

H_{C_i} is fed into a two-layer multi-layer perception (MLP) to produce predicted popularity increment $\Delta\tilde{S}_i = MLP(H_{C_i})$. To capture subtle dynamics of cascades, a temporal-aware differential loss function $\mathcal{L}_{dif}$ is applied to predict the

differential size of the sub-cascades by:

$$\mathcal{L}_{dif} = \frac{1}{TN_c} \sum_{i=1}^{N_c} \sum^{T} (\Delta S_{di} - \Delta \tilde{S_{di}})^2 \tag{5}$$

$$\Delta S_{di} = (|\mathcal{G}_f| - |\mathcal{G}_p|), \Delta \tilde{S_{di}} = MLP(h_{\mathcal{G}}^f - h_{\mathcal{G}}^p) \tag{6}$$

where ΔS_{di} is the differential size between two neighboring sub-cascade graphs (a following $\mathcal{G}_f$ and a previous $\mathcal{G}_p$). N_c is the number of cascades. A differential representation of sub-cascades is calculated by subtracting $h_{\mathcal{G}}^f$ and $h_{\mathcal{G}}^p$. The differential representation is then fed into an MLP layer to predict ΔS_{di}.

The final loss is calculated by:

$$\mathcal{L}(\Delta S_i, \Delta \tilde{S}_i) = \frac{1}{N_c} \sum_{i=1}^{N_c} (log \Delta S_i - log \Delta \tilde{S}_i)^2 + \lambda \mathcal{L}_{dif} \tag{7}$$

where λ is a hyperparameter and is set to 1.0 by default.

3.5 Sparse Encoding

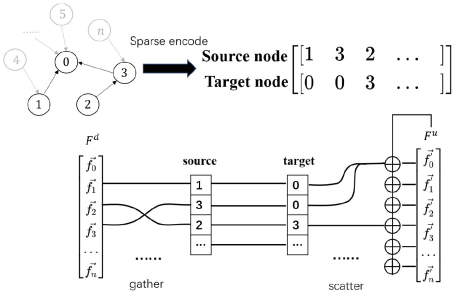


Fig. 3. Sparse graph encoding and computation.

Different from CasCN, VaCas, and CasFlow, which use a full matrix, SRACas applies a sparse matrix to save GPU memories and accelerate computations. The sparse matrix representation records source nodes and target nodes that indicate the edges and directions, as shown in Fig. 3. A matrix is sparsely encoded as:

$$M^s = \begin{bmatrix} s_1 & s_2 & s_3 & s_1 & \dots & s_i \\ t_1 & t_2 & t_2 & t_3 & \dots & t_i \end{bmatrix} \tag{8}$$

where s_i and $t_i (i = 1, 2 \dots, N)$ are a source node and a target node. The gather process aggregates the source nodes with the same target t_i (denoted as the set P_i). The process of the matrix computation is $M^s F^d$, where $F^d \in R^{NF_{in}}$.

The multiplication is improved by accessing the memory and computing in a thread-per-matrix-row pattern [4].

$$F_{t_i}^u = (M^s \cdot F^d)_{t_i} = \sum_{k \in P_i} F_k^d \tag{9}$$

where $F_{t_i}^u \in R^{F_{in}}$ are the representation of node i after a simple sum scatter.

4 Experiments

Two classic tasks are introduced to evaluate the effectiveness of SRACas [2]. One task is to predict the future retweet size of a microblog. The other task predicts the future citation counts of a paper.

4.1 Experimental Setups

Following data preprocessing setups in DeepHawkes [2], the observation time window t_o is set to 1 h, 2 h, and 3 h in the Sina Weibo dataset. The cascades that have less than five retweets within the observation time t_o are removed in experiments. In the dataset APS Citation, we remove papers with fewer than ten citations within the observation time [2]. We set t_o = 5 years, 7 years, and 9 years to predict the size in the 20th year.

The MSLE (mean square log-transformed error) and mSLE (median square log-transformed error) [2] are used as evaluation metrics. We consider the classic and advanced models as strong baselines (Table 1).

4.2 Prediction Performance

Table 1. Performance on the entire cascade set.

	Sina Weibo						**APS Citation**					
t_o	1hour		2hours		3hours		5years		7years		9years	
Metric	MSLE	mSLE	MSLE	mSLE	MSLE	mSLE	MSLE	mSLE	MSLE	mSLE	MSLE	mSLE
DeepCas	3.383	0.810	3.255	0.813	3.186	0.845	1.772	0.730	1.594	0.693	1.579	0.732
TopoLSTM	3.535	1.247	3.386	1.108	3.244	1.101	1.728	0.721	1.615	0.736	1.560	0.644
DeepHawkes	2.307	0.648	2.125	0.597	2.105	0.644	1.347	0.604	1.337	0.598	1.244	0.519
CasCN*	2.862	0.699	2.796	0.774	2.707	0.681	1.397	0.618	1.355	0.563	1.178	0.545
CasFlow*	2.325	0.650	2.256	0.672	2.113	0.668	1.352	0.548	1.345	0.591	1.316	0.571
SRACas	**1.970**	**0.560**	**1.870**	**0.499**	**1.826**	**0.477**	**1.253**	**0.527**	**1.207**	**0.510**	**1.076**	**0.472**

Note that OpenCas optimizes implementations of current models, so performances of some baselines are improved compared to the original papers. Some recent models (e.g., CasCN and CasFlow) are only suitable for predicting cascades with small popularity size [3,9]. Larger cascades form in real-world scenarios, and they are more common in the Sina Weibo dataset than in the APS

Citation dataset. Larger cascades increase the difficulties of cascade prediction and require higher GPU memories, making some models unsuitable. Therefore, a smaller dataset from the Sina Weibo dataset is also used to validate the performance of models by selecting the cascades with no more than over 100 nodes within the observation window following setups [3,9]. The size distributions of the small and large cascades are similar in the APS Citation and Twitter, which is meaningless to be divided.

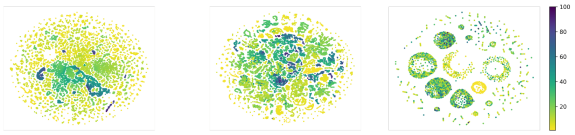
Prediction on the Entire Cascade Set: The entire cascade set preserves cascades with fewer than 1000 nodes within the observation window as described in DeepHawkes [2]. CasCN and CasFlow are infeasible to run experiments directly on the entire set due to their high GPU demands. We apply CasFlow and CasCN by limiting the first 100 participants in the observation window (denote as CasFlow* and CasCN*). SRACas applies sparse encoding to reduce GPU memory usage and enable experiments on the entire cascades. SRACas follows the same training strategies in experiments on both entire and smaller cascade sets. The baseline models outperform their original implementations by re-implementing part of the codes and optimizing training strategies on OpenCas. For example, the performance of DeepCas and DeepHawkes is better than their original papers or previous reports, with the MSLE reduced from 3.63 to 3.38 and from 2.44 to 2.31 [2] at one hour, respectively. SRACas consistently outperforms the baselines by significant margins.

Prediction on Smaller Cascades Set: All the baseline models work on the smaller dataset, but SRACas performs much better, as shown in Table 2. MSLE of DeepCas and DeepHawkes is reduced from 2.958 [3] to 2.494 and from 2.441 [3] to 2.023, respectively. The performance of DeepCas and DeepHawkes is underestimated in previous works. The second lowest MSLE (2.023 by DeepHawkes) is reduced to 1.810 (SRACas) by around 10.5 % at 1 h on Sina Weibo.

Compared with prediction performance on the smaller set, MSLE of CasFlow and DeepHawkes on the entire set is highly decreased by 14% (substantial performance decrease) as shown in Table 2. The performance of CasCN and TopoLSTM even dropped by more than 30% and 40%, respectively. However, SRACas is the most stable: the MSLE only increased from 1.810 to 1.970, and the mSLE increased from 0.534 to 0.560.

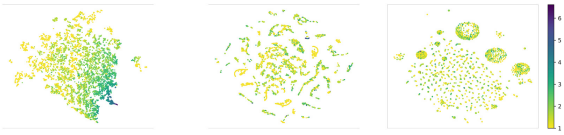
Table 2. Performance on Sina Weibo (smaller set).

	Sina Weibo (Smaller Set)					
t_o	1hour		2hours		3hours	
Metric	MSLE	mSLE	MSLE	mSLE	MSLE	mSLE
DeepCas	2.494	0.758	2.386	0.801	2.316	0.856
TopoLSTM	2.679	0.715	2.595	0.724	2.579	0.703
DeepHawkes	2.023	0.649	1.963	0.536	1.941	0.546
CasCN	2.042	0.553	1.906	0.532	1.867	0.530
CasFlow	2.038	0.547	1.988	0.527	1.920	0.519
SRACas	**1.810**	**0.534**	**1.687**	**0.470**	**1.622**	**0.486**



(a) SRACas (b) CasCN (c) DeepHawkes

Fig. 4. Subcascade representations in Sina Weibo.



(a) SRACas (b) CasCN (c) DeepHawkes

Fig. 5. Differential temporal representations in Sina Weibo.

4.3 Visualization and Explanation

Discussions on Learned Features: To discover the differences in representations learned by different models. We apply t-SNE [1] to project the cascades' or sub-cascade graphs' representations into two-dimensional points. Meaningful representations will gather according to the sizes. Figure 4 represents the sub-cascade graphs' representations with sizes. The aggregation degree of the learned features by SRACas is higher than CasCN and DeepHawkes on relatively large-size sub-cascade graphs.

Furthermore, we explore the differential representation of sub-cascades in different moments by subtracting the representation of $\mathcal{G}_t$ and $\mathcal{G}_{t-1}$ and coloring the data points with incremental sizes between $\mathcal{G}_t$ and $\mathcal{G}_{t-1}$. As shown in Fig. 5, the aggregation effects of CasCN and DeepHawkes are not as obvious as those of SRACas. The nodes in different moments share the same representations in DeepHawkes, which makes it hard to distinguish the structure of sub-cascades. CasCN also fails to distinguish the difference between neighboring sub-cascades, demonstrating the effectiveness of role-aware social attention.

(a) SRACas (b) CasCN (c) DeepHawkes

Fig. 6. Cascade representations in Sina Weibo.

Finally, the representations of entire cascades are visualized in Fig. 6. The learned features by all models are related to the increment size to a certain extent. However, compared to CasCN and DeepHawkes, the points cluster more densely and smoothly according to their incremental popularity in SRACas.

5 Conclusion

This paper proposes a novel model SRACas to solve the problem that current models fail to model the social roles of nodes and discriminate the neighboring sub-cascades, utilizing a social-aware attention mechanism and temporal-aware differential loss. The SRACas achieved significant improvements over strong baseline methods and brought cascade prediction performance to a new level in classic real-world datasets. Furthermore, an open platform OpenCas that aggregates cascade prediction models is built, which simplifies the environmental configuration, data preprocessing, and training of models.

Acknowledgements. This work is supported by the NSF of China (No. 71971002) and the NSF of Guangdong Province, China (No. 2019A1515011792).

References

1. Vincent, D.B., Jean-Loup, G., Renaud, L., Etienne, L.: Fast unfolding of communities in large networks. J. Stat. Mech: Theory Exp. **2008**(10), P10008 (2008)
2. Cao, Q., Shen, H., Cen, K., Ouyang, W., Cheng, X.: Deephawkes: bridging the gap between prediction and understanding of information cascades. In: Proceedings of CIKM, pp. 1149–1158 (2017)
3. Chen, X., Zhou, F., Zhang, K., Goce, T., Zhong, T., Zhang, F.: Information diffusion prediction via recurrent cascades convolution. In: Proceedings of ICDE
4. Matthias, F., Jan, E.L.: Fast graph representation learning with PyTorch Geometric. In: Proceedings of ICLR Workshop on RLGM (2019)
5. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the EMNLP, pp. 1746–1751, Doha, Qatar, October 2014
6. Cheng, L., Jiaqi, M., Xiaoxiao, G., Qiaozhu, M.: Deepcas: an end-to-end predictor of information cascades. In: Proceedings of the 26th WWW, pp. 577–586 (2017)
7. Petar, V., Guillem, C., Arantxa, C., Adriana, R., Pietro, L., Yoshua, B.: Graph attention networks. Proceedings of ICLR (2018)
8. Wang, J., Zheng, V., Liu, Z., Chang, K.: Topological recurrent neural network for diffusion prediction. In: Proceedings of ICDM, pp. 475–484. IEEE (2017)
9. Xu, X., Zhou, F., Zhang, K., Liu, S., Goce, T.: Casflow: exploring hierarchical structures and propagation uncertainty for cascade prediction. TKDE (2021)
10. Yang, Y., et al.: Rain: social role-aware information diffusion. In: Proceedings of AAAI (2015)
11. Fan, Z., Xu, X., Goce, T., Zhang, K.: A survey of information cascade analysis: Models, predictions, and recent advances. ACM Computing Surveys (2021)
12. Zhou, F., Xu, X., Zhang, K., Goce, T., Zhong, T.: Variational information diffusion for probabilistic cascades prediction. In: IEEE INFOCOM, pp. 1618–1627 (2020)