Fully Exploiting Cascade Graphs for Real-time Forwarding Prediction

Xiangyun Tang[1], Dongliang Liao[2*], Weijie Huang[2], Jin Xu[2†], Liehuang Zhu[1†], Meng Shen[13†]
[1]School of Cyberspace Security, Beijing Institute of Technology, China
[2]Data Quality Team, WeChat, Tencent Inc., China
[3]Cyberspace Security Research Center, Peng Cheng Laboratory, China
xiangyunt@bit.edu.cn, {brightliao, wainhuang, jinxxu}@tencent.com, {liehuangz, shenmeng}@bit.edu.cn

Abstract

Real-time forwarding prediction for predicting online contents' popularity is beneficial to various social applications for enhancing interactive social behaviors. Cascade graphs, formed by online contents' propagation, play a vital role in real-time forwarding prediction. Existing cascade graph modeling methods are inadequate to embed cascade graphs that have hub structures and deep cascade paths, or they fail to handle the short-term outbreak of forwarding amount. To this end, we propose a novel real-time forwarding prediction method that includes an effective approach for cascade graph embedding and a short-term variation sensitive method for time-series modeling, making the best of cascade graph features. Using two real world datasets, we demonstrate the significant superiority of the proposed method compared with the state-of-the-art. Our experiments also reveal interesting implications hidden in the performance differences between cascade graph embedding and time-series modeling.

Introduction

The dramatic growth of mobile Internet and social media facilitates the fast propagation of online contents, such as tweets on Twitter, images on Instagram and videos on YouTube. Popularity prediction of online contents is beneficial to many social applications such as recommendation and advertising, drawing wide attentions (Kipf and Welling 2016; Liao et al. 2019; Saito, Nakano, and Kimura 2008). Forwarding amount, the number of times that an online content has been forwarded (or retweeted/reposted) through social media, is one of the most important measurement for measuring the popularity of online contents (Gao et al. 2019; Zhou et al. 2021; Li et al. 2017; Chen et al. 2019b,a), Real-time forwarding prediction aims to model preceding propagation process of online contents and predicts its forwarding amount at the next specific observation time.

In social media, users share and exchange an interesting online content with their friends, thus the propagation of an online content generally starts with the author and spreads through social networks, forming a *cascade graph*. Many methods exploited and innovated cascade graph embedding

[*]The first two authors contributed equally.
[†]Corresponding authors.

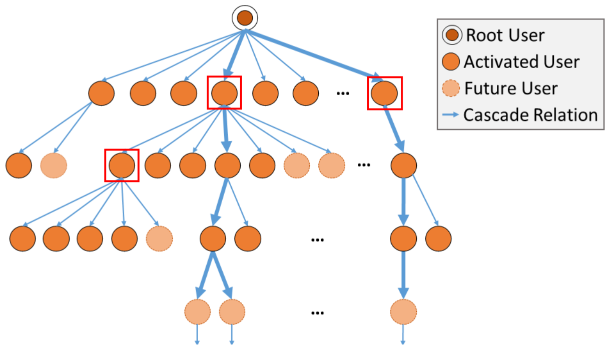

Figure 1: An example of cascade graphs. The social hubs are circled by the red boxes. The thickened cascade relation lines mark the deep propagation paths. Please note that the cascade graph here is presented as tree pattern to illustrate its features, which generally has DAG pattern.

for improving forwarding prediction (Gao et al. 2019; Zhou et al. 2021; Gao et al. 2020). However, it is still pending to reach a competent cascade graph embedding methods and challenging to facilitate real-time forwarding prediction by leveraging cascade graphs.

On the one hand, existing graph embedding methods such as random walk based methods (Li et al. 2017; Grover and Leskovec 2016; Perozzi, Al-Rfou, and Skiena 2014) and Graph Convolution Networks (GCN) based methods (Cao et al. 2019; Chen et al. 2019b; Kipf and Welling 2016; Hamilton, Ying, and Leskovec 2017) are inappropriate for cascade graphs. As visualized in Fig. 1, cascade graphs are generally evolving directed acyclic graphs (DAG) where a directed path represents a content diffusion process through social networks. Information diffusion on social networks is formed as social reinforcement and hub patterns (Weng, Menczer, and Ahn 2013), which makes hub structures and deep cascade paths arise in cascade graphs. Even though GCN based methods can finally travel through all nodes by repeatedly aggregating neighbor node features and updating node features, it is circuitous and rigmarole for modeling deep cascade paths. Random walk based methods randomly select partial pieces of cascade paths, which may lose information about social hubs.

On the other hand, cascade graph embedding tends to emphasize microscopic diffusion process in cascade graphs (Yang et al. 2018; Qiu et al. 2018) and to capture cascade graph structure features. Predicting forwarding amount only via cascade graph embedding ignores many relevant predictive features from the time-series of macroscopic history variation of cascade graph size. In particular, although existing models for time-series modeling have achieved good performances in popularity prediction (Du et al. 2016), such as temporal process approaches (Hawkes intensity process (Rizoiu et al. 2017)), regression models (Yu, Xie, and Sanner 2015) and Recurrent Neural Network (RNN) (Liao et al. 2019). The increasing forwarding amount is very changeable and accompanied by many short-term outbreaks (Rizoiu et al. 2017; Liao et al. 2019), which hampers time-series modeling for real-time forwarding prediction.

In this paper, we propose a novel method, Temporal Cascade Graph modeling (TempCas), to address the above challenges. Specifically, TempCas introduce the following two key techniques, to fully exploit cascade graph features in terms of graph structure and graph size's time-series, and to accomplish accurate real-time forwarding prediction:

1. Competently cascade graph embedding. As shown in Fig. 1, social hub nodes and deep propagation paths are the key to the widespread diffusion of an online content. Rather than random walk on cascade graphs, we implement a heuristic method to sample full *critical paths* on cascade graphs, capturing both the complete diffusion process over social hubs and deep cascade paths. Then Bidirectional Gated Recurrent Unit (BiGRU) with an attention pooling are employed for cascade path embedding. Besides, the "trivial" nodes (e.g., the leaf nodes) of cascade graphs have less propagation influence, but the amount of it implies cascade graph scale. We extract graph scale features as supplement features for cascade graph representations.

2. Short-term time-series variation modeling. Recently, RNNs achieve satisfying performances in time-series modeling, while it is not sensitive to changeable short-term outbreaks. Convolution Neural Network (CNN) can extract local features from time series, but cannot learn long-term sequence changes. Hence, we develop an attention CNN mechanism that captures short-term variation over time on cascade graph size and merges the local features within a fixed window. Then we employ Long Short Term Meomory (LSTM) over the attention CNN to learn the historical trend.

Extensive experiments demonstrate the proposed method significantly outperforms the state-of-the-art. Furthermore, *some interesting implications are discovered* from the performance differences between cascade graph embedding and time-series modeling in diffusion time, scale and early adopter dimensions, on real-time forwarding prediction tasks. The comparison results show that time-series modeling is capable enough for most common real-time forwarding prediction, while cascade graph embedding is more effective for hot-spot detecting, and combination the two types of features show superior performances from all aspects.

Related Work

This paper focuses on cascade graph modeling to predict real-time forwarding amount, and combines cascade graph features in aspects of cascade graph structure and time-series of cascade graph size. In the following, we give brief literature reviews of cascade graph embedding and time-series modeling respectively, then we discuss the existing forwarding prediction methods exploiting cascade graphs.

Time-series based solutions handle the forwarding prediction problem by feature based methods, temporal process methods and deep learning methods. Feature based methods extract hand-crafted features (Piotrkowicz et al. 2017; Shulman, Sharma, and Cosley 2016; Keneshloo et al. 2016) and construct machine learning models for prediction. A number of models were proposed to describe the evolution of accumulation process of information diffusion volumes (Shen et al. 2014; Cui et al. 2013; Gao, Ma, and Chen 2015; Mishra, Rizoiu, and Xie 2016; Rizoiu et al. 2017). Recently, researchers leveraging deep learning frameworks in forwarding predictions have achieved satisfying performances. Deep learning methods can extract features automatically for forwarding prediction (Li, Guo, and Mei 2018; Dou et al. 2018; Zhang et al. 2018). RNNs and CNNs are superior in time-series modeling (Liao et al. 2019; Cao et al. 2017; Du et al. 2016). However, modeling time-series of cascade graph size only is not enough to understand the complex and changeable information diffusion and communication through social networks.

Cascade graph embedding mainly includes random walk based methods and GCN based methods. GCN is a popular and effective graph embedding technology (Kipf and Welling 2016), which updates node features by aggregating neighbor node features. GAN (Veličković et al. 2017) and GraphSAGE (Hamilton, Ying, and Leskovec 2017) are developed from GCN. Random walk based methods sample paths randomly to capture the topological structure of graphs (Grover and Leskovec 2016; Perozzi, Al-Rfou, and Skiena 2014). While we argue that existing graph embedding methods are inadequate for forwarding predictions with cascade graphs that feature hub structures and deep cascade paths.

Many methods have exploited cascade graphs to improve forwarding prediction (Gao et al. 2020; Gao et al. 2019; Zhou et al. 2021; Chen et al. 2019a; Cao et al. 2019). Chen et al. proposed CasCN leveraging CasLaplacian for modeling information diffusion, which is an extension of directed sensor graph (Li et al. 2018). DeepCas (Li et al. 2017) transformed the cascade graph into node sequences by random walk to learn the representation of individual graphs. Molaei, Zare, and Veisi exploited meta-paths as main entities of social networks and learned meta-paths by heterogeneous deep diffusion network. Deephawkes (Cao et al. 2017) employs interpretable factors of Hawkes process for prediction. Existing methods do not take macroscopic history variation of cascade graph size seriously when conducting forwarding prediction with cascade graphs, and we argue that existing graph embedding methods are inappropriate for cascade graphs.

Problem Formulation

We regard the real-time forwarding prediction task as a regression task, predicting real-time forwarding amount of online contents. We divide continuous time into time slots.

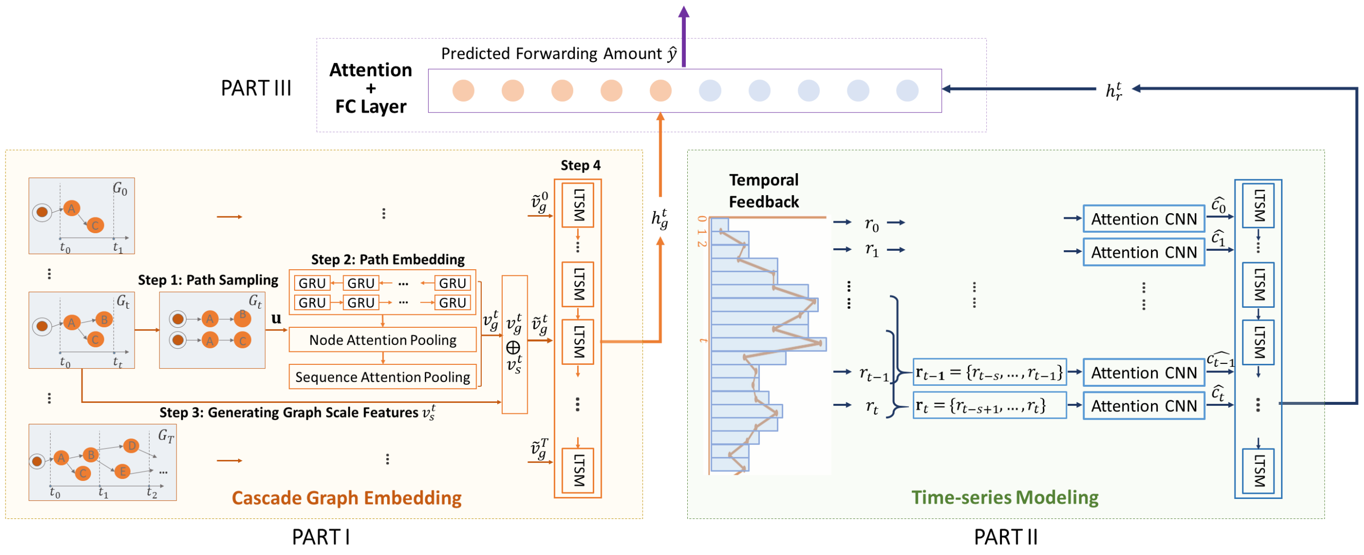

Figure 2: Overview of the proposed method TempCas

At each time slot t, we capture a cascade graph snapshot G_t and count the forwarding amount as the temporal feedback r_t. The temporal feedback r_t is the volume of "forward" in time slot t, which is the increasing of cascade graph size in the time slot. Formally, given a cascade graph snapshot series $\{G_1, G_2, ..., G_t\}$ and a temporal feedback series $\{r_1, r_2, ..., r_t\}$, the objective is to predict the forwarding amount at the next time slot.

Method

The proposed method TempCas contains three main parts: cascade graph embedding, time-series modeling and a fusion layer. *(i)* Cascade graph embedding includes a novel heuristic method to sample full critical paths, and a hierarchical attention network to assemble node features into graph vectors. Graph scale features are concatenated with the graph vector, forming graph embedding outputs. To better understand the temporal variation, a LSTM layer is adopted on the graph embedding output series. *(ii)* Time-series modeling adopts an attention CNN mechanism on temporal feedback series for capturing short-term variation and appends a LSTM layer for modeling the historical trend. *(iii)* The fusion layer integrates the above parts through an attention mechanism and fully connected layers. The overview of TempCas is depicted in Fig. 2.

Cascade Graph Embedding

Given a cascade graph snapshot series, cascade graph embedding is responsible for capturing the cascade graph features in terms of diffusion, scale and temporal, whose workflow with 4 steps is illustrated in Fig. 2.

For cascade graph diffusion features: Social hubs and deep cascade paths influence the cascade graph mostly, while other "trivial" nodes (e.g., the leaf nodes) have little propagation influence. To obtain the cascade graph diffusion features adequately, it is the key to capturing the critical paths that cover the most influential nodes and paths.

Step 1 : cascade path sampling. Our path sampling strategy is formalized in Algorithm 1 where $Child(u_i)$ is the children node set of u_i. Assuming that the reasonable weight ω_i for each node u_i has been learned to reflect node propagation influence in cascade graphs by a weight function WEIGHT(·), the intuition behind Algorithm 1 is by employing the root node as the starting node and iteratively selecting the node with the maximum weight as the next node to join the path being sampling, until a leaf node is selected or reaching the the max length of paths. TempCas can catch a full critical cascade path that carries more diffusion features. In particular, once a node $\hat{u}_l$ is picked, we decay its weight $\omega_{\hat{u}_l}$ by Equation (1) to avoid sampling repeating paths:

$$\omega_i = \omega_i * (1 - \frac{1}{\omega_i}) \tag{1}$$

Ideally, the weight for a node would exactly reflect the propagation influence of the node in a cascade graph snapshot. We examine three candidate weight functions:

(i) Node weights by offspring amount. The large offspring amount indicates the node is in a critical path either containing social hubs or in-depth propagation.

$$\texttt{WEIGHT}(u_i) = o_i := \begin{cases} 1 & if\ Child(u_i)\ is\ \emptyset \\ 1 + \sum_{u_j \in Child(u_i)} o_j & \text{otherwise} \end{cases} \tag{2}$$

(ii) Node weights by depth. The depth de_i of a node indicates the propagation depth from the root node to the node.

$$\texttt{WEIGHT}(u_i) = de_i \tag{3}$$

(iii) Node weights by offspring amount plus depth. Both the offspring amount and depth may represent nodes' position and propagation influence in cascade graphs.

$$\texttt{WEIGHT}(u_i) = syn_i := o_i + de_i \tag{4}$$

Algorithm 1 Path Sampling Strategy

```
Input: A cascade graph snapshot $G_t$; the max length of paths $L$; the max number of paths $K$; a root node $u_{root}$; a node set of the graph $\{u_i\}_{i=1}^n$; a weight function WEIGHT(·); $U \leftarrow \emptyset$.
Output: Sampled paths $U$.
 1: $\{\omega_i\}_{i=1}^n \leftarrow$ WEIGHT($\{u_i\}_{i=1}^n$)
 2: for $k = 1, ...K$ do
 3:     $\hat{C} \leftarrow Child(u_{root})$, $\mathbf{u}_k \leftarrow \emptyset$
 4:     for $l = 1, ...L$ do
 5:         $\hat{u}_l \leftarrow \arg\max_{u_j \in \hat{C}}(\omega_j)$
 6:         $\mathbf{u}_k \leftarrow \mathbf{u}_k \cup \{\hat{u}_l\}$
 7:         Update $\omega_{\hat{u}_l}$ by Equation (1)
 8:         $\hat{C} \leftarrow Child(\hat{u}_l)$
 9:     end for
10:     $U \leftarrow U \cup \{\mathbf{u}_k\}$
11: end for
```

In our experiments (cf. Table 3), we find that the offspring amount weight function (Equation (2)) works best among these weight functions.

Step 2 : cascade path embedding. Given a sampled cascade path consisting of a node sequence $\mathbf{u} = \{u_1, u_2, ..., u_L\}$, a BiGRU layer is conducted to capture the diffusion process on $\mathbf{u}$ and obtain the node representation $\mathbf{h}$:

$$\mathbf{h} = BiGRU(\mathbf{u}) \tag{5}$$

In order to convert node representations to graph leveled representations, two attention pooling mechanisms are adopted. Let $Att(\cdot)$ denotes an attention pooling mechanism (Vaswani et al. 2017). The first attention layer aggregates the node vector $\mathbf{h} = \{h_1, h_2, \cdots, h_L\}$ to a path-level vector p, and the second one aggregates the path-level vector $\mathbf{p} = \{p_1, p_2, \cdots, p_K\}$ to a graph-level vector v_g^t:

$$p = Att(\mathbf{h}) \tag{6}$$

$$v_g^t = Att(\mathbf{p}) \tag{7}$$

For cascade graph scale features (Step 3):

The critical paths cover the most influential nodes and paths, but may lose information of many trivial nodes that contribute to the main part of cascade graphs but have little diffusion influence. Therefore we extract cascade scale features v_s^t as supplementary information to let the model understand the structure and scale of trivial nodes. The cascade scale features used are shown in Table 1.

For cascade graph temporal features (Step 4):

Cascade graphs are evolving over time. Since LSTM is capable of learning long-term dependencies, after concatenating the cascade diffusion features v_g^t and scale features v_s^t as the cascade snapshot representation $\hat{v}_g^t$, we feed its series to a LSTM layer for capturing the cascade temporal features. $\oplus$ represents vector concatenation:

$$\hat{v}_g^t = v_g^t \oplus v_s^t \tag{8}$$

$$h_g^t = LSTM(\hat{v}_g^t) \tag{9}$$

No.	Features
1	Number of nodes
2	Number of leaves
4	The median of number of neighbors
5	Maximum number of neighbors
6	Average number of neighbors (all)
7	Average number of neighbors (top 50%)
8	Average number of neighbors (top 75%)
9	Maximum path length
10	Average path length (all)
11	Average path length (top 50%)
12	Average path length (top 75%)

Table 1: Cascade Graph Scale Features

Time-series Modeling

Short-term outbreaks seriously impact on real-time forwarding prediction. We develop an attention CNN mechanism (Liao et al. 2019) to model short-term outbreaks and stack a LSTM layer to model the impact of historical short-term outbreaks. Let us consider the forwarding amount variation as a figure with one line: CNN is used to capture the burst pattern, then the Attention mechanism can focus on the sudden break or down in the current time window. Concretely, at a time slot t, we take a clipped temporal feedback series $\mathbf{r}_t = \{r_{t-s+1}, r_{t-s+2}, \cdots, r_t\}$ with window size s. We apply CNN with same padding on $\mathbf{r}_t$.

$$\mathbf{c}_t = CNN(\mathbf{r}_t) \tag{10}$$

Then, an attention mechanism is performed to dynamically aggregate the temporal features in the window and capture the short-term outbreaks.

$$\hat{c}_t = Att(\mathbf{c}_t) \tag{11}$$

Finally, LSTM is employed for merging the impact of historical short-term outbreaks:

$$h_r^t = LSTM(\hat{c}_t) \tag{12}$$

Fusion Layer

The time-series modeling results h_r^t contain the temporal variation pattern of cascade graph size. The cascade graph embedding results h_g^t capture the cascade graph features in terms of diffusion, scale and temporal. We concatenate the above two parts and feed it to an attention pooling mechanism and fully-connected layer sequentially, to get prediction outputs. The fusion layer can be defined as follows:

$$h_t = h_v^t \oplus h_g^t \tag{13}$$

$$\hat{y} = FC(Att(h_t)) \tag{14}$$

where $\hat{y}$ is the predicted result. For optimization, we adopt the mean square error loss and L_2 regularization.

Experiments

Dataset

Weibo Dataset[1]. Weibo dataset is collected from a popular Chinese microblog platform (Cao et al. 2017). It contains

[1]https://github.com/CaoQi92/DeepHawkes

Dataset	Weibo	Multimedia
Content Number	119,313	26,893
Graph Snapshot Number	5,727,024	4,033,950
Avg. Nodes per G_t	142.36	4,637.70
Avg. Edges per G_t	174.02	7043.38

Table 2: Dataset Statistics

119, 313 posts generated on June 1st, 2016, and tracks all re-tweets within the next 24 hours.

Multimedia Content Dataset[2]. We collect a multimedia content dataset from a widely used mobile social application. In the application, both media organizations and personal users can publish multimedia contents that are the mixtures of text, images, audios and videos. We randomly sample multimedia contents from August 1, 2019 to September 30, 2019 and track all forwardings of each multimedia content within the next 75 hours. In total, 26,893 multimedia contents are included in the dataset.

The statistics of the two datasets are presented in Table 2. We set each time slot as 30 minutes, so that there are 150 time windows spanning 75 hours on Multimedia Content dataset, and 48 spanning 24 hours on Weibo dataset. We randomly take 80% of data for training, 10% for validation and 10% for evaluation. We conduct Z-Score normalization on the ground truth, transforming data to have a mean of zero and a standard deviation of 1.

Experiment Settings

Three widely used evaluation metrics are adopted to evaluate our approach, including Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R-Squared (R^2).

Baseline methods: TempCas combines technologies of time-series modeling and cascade graph embedding. Thus, we select the state-of-the-art methods in each category as strong baselines. The latest methods exploiting cascade graphs for forwarding predictions are included in baselines.

Time-series based methods take variation features of users' feedbacks as input to predict the forwarding amount. We extract all the predictive features that could be generalized across datasets from recent studies of popularity prediction (Liao et al. 2019; Dou et al. 2018; Mishra, Rizoiu, and Xie 2016). We employ both shallow machine learning methods and deep learning methods: Linear Regression (**LR**); Gradient Boosting Decision Tree (**GBDT**); **CNN** and **LSTM**.

Cascade graph embedding methods learn content diffusion information. In addition to **GCN** (Kipf and Welling 2016), we compare TempCas with following methods. **GraphSAGE** (Hamilton, Ying, and Leskovec 2017) updates node features by aggregating the node features within convolution windows, where mean pooling aggregator is adopted in our experiments. **DeepWalk** (Perozzi, Al-Rfou, and Skiena 2014) generates network representations by truncated random walks on each graph, where graph vectors are generated in the same way as TempCas.

[2]https://github.com/tangguotxy/TempCas

CasCN (Chen et al. 2019b) employs a sequence of subcascade graphs with cascade Laplacian. **DeepCas** (Li et al. 2017) generates node sequences by heuristic random walks, generating graph vectors by BiGRU and an attention mechanism. **Deephawke** (Cao et al. 2017) employs Hawkes' interpretable factors process for prediction.

Please notice that this paper focuses on cascade graph modeling, some existing popularity prediction methods, complemented by other features such as text content (Liao et al. 2019; Piotrkowicz et al. 2017), publishers and users' personal information (Chen et al. 2019a; Zhang et al. 2018), are complementary directions and can be plugged into TempCas. Thus, they are not included in our baselines. Micro-level models of cascade graphs (Zhou et al. 2021; Qiu et al. 2018; Wang et al. 2017) that focus on individual user actions/responses to specific information items, are also excluded.

Versions of TempCas: We derive variants of TempCas according to the three weight functions: **TempCas** adopts the weight function of offspring amount (Equation (2)). **TempCas-d** adopts the depth weight function (Equation (3)). **TempCas-s** adopts the weight function of the synthesis of offspring amount and depth (Equation (4)).

For ablation study, we include two simplified versions: **TempCasT** is the time-series modeling part of our model, and **TempCasG** is the cascade graph embedding part.

Hyper-Parameters: In the cascade graph embedding part of TempCas, the max path length L is fixed as 8 for multimedia contents and 4 for Weibo posts, and the max path number k for each graph is set as 100 for multimedia contents and 50 for Weibo posts. The hidden size of the hierarchical attention network and the LSTM layer is set as 64 and 128 respectively. In time-series modeling, the kernel size of CNN is set as 5 and the hidden size of CNN and LSTM is set as 128. The window size of attention CNN is 12 for multimedia contents and 5 for Weibo posts. We adopt 2 dense layers for the final output, where the hidden dimensions are 128 and 1 respectively. At last, we leverage the Xavier initialization and Adam optimizer for parameter learning. L_2 factor is set as 10^{-5}. For baselines, we follow the setting of GraphSAGE where the number of aggregating operations is set as 2. The sequence length and sequence number for DeepWalk and DeepCas is set the same as our work. For other hyper-parameters of GraphSAGE, CasCN, Deephawke and Deepcas, we follow the setting of their works. Other hyper-parameters of all the baselines are tuned to obtain the best results on validation sets.

Results Analysis

Overall performance: The overall performance of all competing methods is displayed in Table 3. The last row shows the performance of the complete version of our method with the weight function of offspring amount, outperforming all baselines with a significant drop of RMSE and MAE.

As can be seen from Table 3, the model effectiveness of cascade graph embedding based is relatively weaker than time-series based. In fact, when facing the hot-spot online

Methods	Multimedia Dataset			Weibo Dataset		
	MAE	RMSE	R^2	MAE	RMSE	R^2
LR	0.2809	0.7830	0.5343	0.0454	0.7181	0.4442
GBDT	0.1764	0.6319	0.6968	0.0420	0.7073	0.4577
CNN	0.2394	0.6174	0.7072	0.0343	0.2819	0.9454*
LSTM	0.1533	0.4190	0.8711	0.0374	0.2412	0.5382
TempCasT	0.1502*	0.4153*	0.8854*	0.0312*	0.2163*	0.9012
GCN	0.4418	0.9800	0.1977	0.0592	0.7476	0.0624
GraphSAGE	0.4200	0.9809	0.2578	0.0520	0.6180	0.0508
DeepWalk	0.4078	0.9796	0.2559	0.0518	0.6859	0.1214
TempCasG	0.3106*	0.7744*	0.5104*	0.0384*	0.5297*	0.5424*
DeepCas	0.4107	0.9773	0.2462	0.0518	0.6856	0.1768
CasCN	0.7891	1.7462	0.0027	0.0582	0.6605	0.0266
DeepHawke	0.5206	0.5775	0.4112	0.0391	0.3043	0.2511
TempCas-d	0.0869	0.2521	0.9461	0.0359	0.1039	0.9365
TempCas-s	0.0614	0.2840	0.9528	0.0173	0.1905	0.9030
TempCas	**0.0599***	**0.1479***	**0.9807***	**0.0133***	**0.0781***	**0.9704***

Table 3: Overall Performance. The best performance is boldfaced; the highest score in each category is labeled with '*'; MAE and RMSE are both the lower the better, R^2 is the bigger the better.

contents, cascade graph embedding based methods are more accurate than time-series based methods. We present a detailed discussion for the performance comparison in the last subsection of experiments.

TempCasT stacks an attention CNN and a LSTM layer for capturing short-term variation, obtaining the best results among other time-series based forwarding prediction methods. Random walk based methods (i.e. DeepWalk and DeepCas), randomly selecting a node in graphs as the start node for path sampling, tend to truncate the complete deep propagation paths and may fail to capture the social hub nodes. GCN based methods (i.e. GCN, GraphSage and CasCN) generate node embeddings by aggregating features from nodes' local neighborhoods, which is indirect for modeling the cascade process. Compared with these graph embedding methods, TempCasG generates full critical paths starting from the publisher of online contents and ending to the latest content consumers, which models the information diffusion process adequately and achieves better results.

The two versions of TempCas according to the different weight functions, TempCas-d and TempCas-s, all have performance degradation, compared to TempCas. It empirically proves the effectiveness of the offspring amount weight function. Offspring amount implies the propagation influence of nodes in terms of depth and width. Selecting the next node with the maximum offspring amount directly grasps the central node.

Effectiveness of time-series modeling: To obtain stronger baselines and highlight the effectiveness of our time-series modeling in forwarding prediction, we enhance the graph modeling methods in baselines by concatenating their graph modeling results with our time-series modeling results and conducting the same "Fusion layer" of TempCas. We name enhanced methods as **EnGCN**, **EnGraphSAGE**, **EnDeepWalk** and **EnDeepCas**. The average results on Multimedia dataset are reported in Table 4(a). Compared with the results in Table 3, the performances of these enhanced methods are improved a lot. Cascade graph embedding just captures information propagation mode, failing to fully express the history popularity variation of forwarding amount. The point is that cascade graph embedding can supplement the comprehension of information diffusion and communication through social networks.

Methods	Multimedia Dataset		
	MAE	RMSE	R^2
EnGCN	0.1500	0.3931	0.8964
EnGraphSage	0.2231	0.5713	0.7183
EnDeepWalk	0.1564	0.3494	0.9005
EnDeepCas	0.2260	0.4576	0.8195

(a) Enhanced graph embedding methods.

Methods	Multimedia Dataset		
	MAE	RMSE	R^2
CNN	0.2355	0.5964	0.7255
LSTM	0.1495	0.3446	0.8897
TempCasT	0.1538	0.3018	0.9151
GCN	0.3389	0.8200	0.4416
GraphSAGE	0.3143	0.7888	0.5217

(b) Evaluating the effectiveness of graph scale features.

Table 4: Model Analysis

Methods	K	Node Cov.	Avg. Len.	Avg. Offs
DeepWalk	32	0.0290	1.2006	1.5601
DeepCas		0.0284	1.1747	1.4589
TempCas		0.0649*	2.6854*	1.8628*
DeepWalk	64	0.1151	1.2209	1.5045
DeepCas		0.1122	1.1901	1.4879
TempCas		0.2154*	2.2844*	1.5964*
DeepWalk	128	0.2217	1.2317	1.5562
DeepCas		0.2148	1.1932	1.5069
TempCas		0.3718*	2.0652*	1.8838*

Table 5: Path Sampling Strategy Performance. K is the max number of paths in sampling; the highest score for each K value is labeled with '*'.

Effectiveness of graph scale features: To evaluate the function of graph scale features included in our cascade graph embedding part, the graph scale features as supplementary features are concatenated to the graph modeling outputs for GCN and GraphSage and to the temporal feedback for CNN and LSTM. We run these methods on Multimedia dataset and report the average results in Table 4(b). Referring to Table 3, the performances of all these improved methods are better than their previous version. It confirms our intuition that the graph scale features let the model understand the graph structure more explicitly and improve the model capacity of real-time forwarding prediction.

Effectiveness of path sampling strategy: In order to evaluate path selection strategies, we run the path strategies of TempCas, DeepWalk and DeepCas on Multimedia dataset and report the node coverage ($\frac{sampled\ nodes}{nodes\ amount}$), the average length of sampled path, and the average out-degree of sampled nodes. K is the max number of paths in sampling,

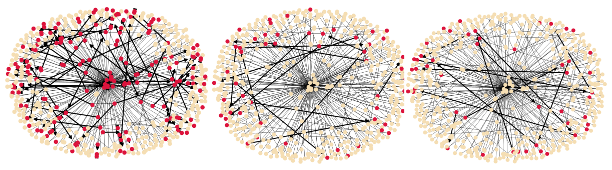

Figure 3: Path strategy visualization (K=64): red nodes are denoted the selected nodes in sampled paths; edges at the sampled paths are thickened.

which is varied from 32 to 128, and the path length L is fixed as 10 in this subsection. As shown in Table 5, no matter which value K takes, TempCas always has the highest node coverage, average length and average out-degree, which means that the path strategy of TempCas is capable of catching more comprehensive graph information than the path strategies of DeepWalk and DeepCas. TempCas fixes the root node (i.e., authors of online contents) as the start node, hence it obtains the full cascade paths and covers more nodes.

Fig. 3 shows a legible illustration where three path strategies are conducted on a same cascade graph. Following the setting of DeepCas (Li et al. 2017), DeepCas-deg is with transition probabilities proportional to node degrees. It is obvious that TempCas covers more hub nodes and deep cascade paths (i.e. selected nodes are nearer to the center). Since the start node is sampled with the probability assigned by node degree in DeepCas and sampled randomly in DeepWalk, when the proportion of trivial nodes is large, they may choose trivial nodes as start nodes, causing propagation information loss.

Interpreting the Performance Differences

Table 3 empirically shows time-series based methods perform better than cascade graph embedding based. Here, we try to explore the reasons underling it, and confirm the necessity of combining cascade graph embedding and cascade graph size's time-series modeling in Tempcas.

To do this, for multimedia contents on Multimedia dataset, we correlate RMSE results with the corresponding final forwarding amount, depicting the average results in Fig. 4. At the left of Fig. 4, the pink line denoting TempCasG is always higher than the two dashdot lines denoting TempCasT and RNN. While the forwarding amount exceeds $25K$, the RMSE results of TempCasG are lower than the results of TempCasT and RNN. Notice that a smaller RMSE means a better prediction result. Of course, multimedia contents with the forwarding amount more than $25K$ make up a small portion of the total. We still could make the following assumptions: the larger forwarding amount containing richer cascade graph information may lead to the more powerful prediction capability of cascade graph embedding based methods. In other words, *when facing hot-spot contents, cascade graph embedding based methods tend to perform better than time-series based methods.*

Fig. 5 reinforces the above conclusions, where we report

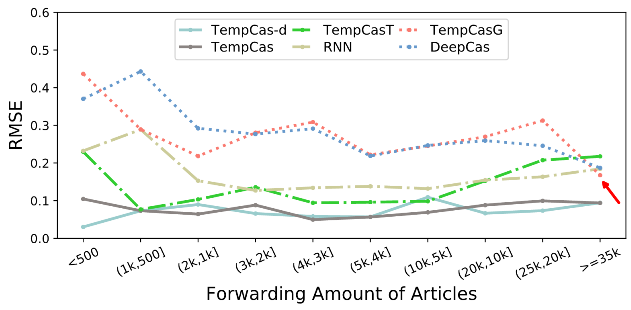

Figure 4: Relationship between model accuracy and content popularity. Lower RMSE results mean better performances. Dotted, dashdot and solid lines mark the graph based, the time-series based and the methods with combining features respectively. The red arrow marks the reversals that the performance of the graph based method TempCasG exceeds the two time-series based methods TempCasT and RNN.

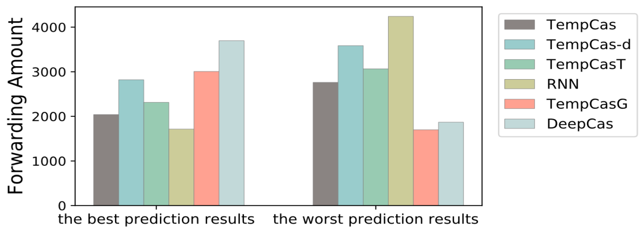

Figure 5: Forwarding amounts of the top 500 multimedia contents having RMSE extremum.

the average of final forwarding amount, for the top 500 multimedia contents having the smallest predicted RMSE (the best prediction results) and the top 500 having the largest predicted RMSE (the worst prediction results). Fig. 5 shows that graph based methods (i.e., TempCasG and DeepCas) obtain their best prediction results with the richest user feedback information and get their worst results with the smallest forwarding amount. By contrast, for time-series methods TempCasT and RNN, wealthy user feedback information leads to the worst results. From this we draw the conclusion that *time-series based methods cannot understand well with the cascade size information under complex social networks.*

Discussion

In this paper, we propose a Temporal Cascade Graph modeling (TempCas) method that measures cascade graphs to facilitate real-time forwarding prediction. TempCas consists of a novel approach for cascade graph embedding that captures cascade graph features in terms of diffusion, scale and temporal, and includes a short-term variation sensitive method for modeling the historical variation of cascade graph size. Experiments on two real world datasets demonstrate the proposed method significantly outperforms the state-of-the-art.

Acknowledgments

Under the tutorship of Dongliang Liao, a WeChat colleague and expert in NLP, this work was done when Xiangyun Tang was interning at WeChat, Tencent Inc, which is supported by 2020 Tencent Rhino-Bird Elite Training Program. This work is supported by the Key Research and Development Program for Guangdong Province (2019B010136001), Science and Technology Planning Project of Guangdong Province LZC0023 and LZC0024.

References

Cao, Q.; Shen, H.; Cen, K.; Ouyang, W.; and Cheng, X. 2017. Deephawkes: Bridging the gap between prediction and understanding of information cascades. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 1149–1158. ACM.

Cao, Q.; Shen, H.; Gao, J.; Wei, B.; and Cheng, X. 2019. Coupled Graph Neural Networks for Predicting the Popularity of Online Content. *arXiv preprint arXiv:1906.09032* .

Chen, G.; Kong, Q.; Xu, N.; and Mao, W. 2019a. NPP: A neural popularity prediction model for social media content. *Neurocomputing* 333: 221–230.

Chen, X.; Zhou, F.; Zhang, K.; Trajcevski, G.; Zhong, T.; and Zhang, F. 2019b. Information Diffusion Prediction via Recurrent Cascades Convolution. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, 770–781. IEEE.

Cui, P.; Jin, S.; Yu, L.; Wang, F.; Zhu, W.; and Yang, S. 2013. Cascading outbreak prediction in networks: a data-driven approach. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 901–909. ACM.

Dou, H.; Zhao, W. X.; Zhao, Y.; Dong, D.; Wen, J.-R.; and Chang, E. Y. 2018. Predicting the Popularity of Online Content with Knowledge-enhanced Neural Networks. In *Proceedings of the 24th ACM SIGKDD international conference on Knowledge discovery and data miningD*. ACM.

Du, N.; Dai, H.; Trivedi, R.; Upadhyay, U.; Gomez-Rodriguez, M.; and Song, L. 2016. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1555–1564. ACM.

Gao, L.; Zhou, B.; Jia, Y.; Tu, H.; Wang, Y.; Chen, C.; Wang, H.; and Zhuang, H. 2020. Deep Learning for Social Network Information Cascade Analysis: a survey. In *2020 IEEE Fifth International Conference on Data Science in Cyberspace (DSC)*, 89–97.

Gao, S.; Ma, J.; and Chen, Z. 2015. Modeling and predicting retweeting dynamics on microblogging platforms. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, 107–116. ACM.

Gao, X.; Cao, Z.; Li, S.; Yao, B.; Chen, G.; and Tang, S. 2019. Taxonomy and Evaluation for Microblog Popularity Prediction. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 13(2): 1–40.

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

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, 1024–1034.

Keneshloo, Y.; Wang, S.; Han, E.-H.; and Ramakrishnan, N. 2016. Predicting the popularity of news articles. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, 441–449. SIAM.

Kipf, T. N.; and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* .

Li, C.; Guo, X.; and Mei, Q. 2018. Joint Modeling of Text and Networks for Cascade Prediction. In *Twelfth International AAAI Conference on Web and Social Media*.

Li, C.; Ma, J.; Guo, X.; and Mei, Q. 2017. Deepcas: An end-to-end predictor of information cascades. In *Proceedings of the 26th international conference on World Wide Web*, 577–586. International World Wide Web Conferences Steering Committee.

Li, Y.; Yu, R.; Shahabi, C.; and Liu, Y. 2018. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *ICLR*.

Liao, D.; Xu, J.; Li, G.; Huang, W.; Liu, W.; and Li, J. 2019. Popularity Prediction on Online Articles with Deep Fusion of Temporal Process and Content Features. In *Thirty-Three AAAI Conference on Artificial Intelligence (AAAI'19)*.

Mishra, S.; Rizoiu, M.-A.; and Xie, L. 2016. Feature driven and point process approaches for popularity prediction. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 1069–1078. ACM.

Molaei, S.; Zare, H.; and Veisi, H. 2019. Deep Learning Approach on Information Diffusion in Heterogeneous Networks. *arXiv preprint arXiv:1902.08810* .

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 701–710. ACM.

Piotrkowicz, A.; Dimitrova, V.; Otterbacher, J.; and Markert, K. 2017. Headlines matter: Using headlines to predict the popularity of news articles on twitter and facebook. In *Eleventh International AAAI Conference on Web and Social Media*.

Qiu, J.; Tang, J.; Ma, H.; Dong, Y.; Wang, K.; and Tang, J. 2018. DeepInf Social Influence Prediction with Deep Learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, 2110–2119. New York, NY, USA: Association for Computing Machinery. ISBN 9781450355520. doi:

10.1145/3219819.3220077. URL https://doi.org/10.1145/3219819.3220077.

Rizoiu, M.-A.; Xie, L.; Sanner, S.; Cebrian, M.; Yu, H.; and Van Hentenryck, P. 2017. Expecting to be hip: Hawkes intensity processes for social media popularity. In *Proceedings of the 26th International Conference on World Wide Web*, 735–744. International World Wide Web Conferences Steering Committee.

Saito, K.; Nakano, R.; and Kimura, M. 2008. Prediction of information diffusion probabilities for independent cascade model. In *International conference on knowledge-based and intelligent information and engineering systems*, 67–75. Springer.

Shen, H.; Wang, D.; Song, C.; and Barabási, A.-L. 2014. Modeling and predicting popularity dynamics via reinforced poisson processes. In *Twenty-eighth AAAI conference on artificial intelligence*.

Shulman, B.; Sharma, A.; and Cosley, D. 2016. Predictability of popularity: Gaps between prediction and understanding. In *Tenth International AAAI Conference on Web and Social Media*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention Is All You Need. *CoRR* abs/1706.03762.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* .

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

Weng, L.; Menczer, F.; and Ahn, Y.-Y. 2013. Virality prediction and community structure in social networks. *Scientific reports* 3: 2522.

Yang, C.; Sun, M.; Liu, H.; Han, S.; Liu, Z.; and Luan, H. 2018. Neural Diffusion Model for Microscopic Cascade Prediction. *arXiv preprint arXiv:1812.08933* .

Yu, H.; Xie, L.; and Sanner, S. 2015. The lifecyle of a youtube video: Phases, content and popularity. In *Ninth International AAAI Conference on Web and Social Media*.

Zhang, W.; Wang, W.; Wang, J.; and Zha, H. 2018. User-guided hierarchical attention network for multi-modal social image popularity prediction. In *Proceedings of the 2018 World Wide Web Conference*, 1277–1286. International World Wide Web Conferences Steering Committee.

Zhou, F.; Xu, X.; Trajcevski, G.; and Zhang, K. 2021. A survey of information cascade analysis: Models, predictions, and recent advances. *ACM Computing Surveys (CSUR)* 54(2): 1–36.