Knowl Inf Syst (2018) 57:721–739
https://doi.org/10.1007/s10115-017-1143-0

SHORT PAPER

Learning sequential features for cascade outbreak prediction

Chengcheng Gou[1,2] · **Huawei Shen**[1,2] · **Pan Du**[1] · **Dayong Wu**[1] · **Yue Liu**[1] · **Xueqi Cheng**[1,2]

Received: 16 August 2016 / Revised: 24 March 2017 / Accepted: 27 December 2017 /
Published online: 20 January 2018

Abstract Information cascades are ubiquitous in various online social networks. Outbreak of cascades could cause huge and unexpected effects. Therefore, predicting the outbreak of cascades at early stage is of vital importance to avoid potential bad effects and take relevant actions. Existing methods either adopt regression or classification technique with exhaustive feature engineering or predict cascade dynamics via modeling the stochastic process of cascades using a hard-coded diffusion–reaction function. One salient issue of these methods is that these methods heavily depend on human-defined knowledge, features or functions. In this paper, we propose to use recurrent neural network with long short-term memory to directly learn sequential patterns from information cascades, working in a fully data-driven manner. With the learned sequential patterns, the outbreak of cascade could be accurately predicted. Extensive experiments on both Twitter and Sina Weibo datasets demonstrate that our method significantly outperforms state-of-the-art methods at the prediction of cascade outbreaks.

✉ Chengcheng Gou
gouchengcheng@gmail.com

Huawei Shen
shenhuawei@ict.ac.cn

Pan Du
dupan@software.ict.ac.cn

Dayong Wu
wudayong@ict.ac.cn

Yue Liu
liuyue@ict.ac.cn

Xueqi Cheng
cxq@ict.ac.cn

1 CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China

2 University of Chinese Academy of Sciences, Beijing, China

Keywords Social network · Outbreak prediction · Sequential feature · LSTM · Popularity prediction

1 Introduction

Online social networking services and social media platforms, such as Twitter, Sina Weibo and Facebook, are producing numerous user-generated contents every day. These contents are spread among peoples following their social relationships, leaving us lots of information cascades. Indeed, this kind of information cascades is universal and also identified in other domains such as e-mail [11], blogging [20] and product recommendation [18]. The study of information cascades has important implications to an array of applications, e.g., advertisement campaign, rumor prevention and recommendation. Studies on information cascades focused on characterizing cascades at either micro- or macro-level. At micro-level, researchers aim to determine the probability that one user will propagate a particular piece of information, or to predict when the next propagation will occur. At macro-level, typical studies include popularity prediction and outbreak prediction, formulating the prediction of cascades as a regression problem [24], [26], [28] or a classification problem [8], respectively. Popularity prediction aims to predict the size of information cascades, i.e., predicting how popular a piece of information will become in the future. Generally, the size of cascades is unevenly distributed, i.e., the majority of them have small sizes, while a few could cause huge propagation, called outbreak cascades. The asymmetric distribution of popularity motivates researchers to study the problem of outbreak prediction, identifying the probable outbreak cascades at early stage.

Existing methods for outbreak prediction fall into two main paradigms. Methods in the first paradigm generally work in standard machine learning frameworks, extracting human-defined features, including content features, structural features [2], temporal features [8, 26] and demographical features, and then learning a prediction function based on these features (Fig. 1a) [15–17,24,30]. The capability of these methods depends on how to define appropriate features, which is heuristic in practice and no effective guideline exists. The second kind of methods directly model the stochastic process of cascade dynamics. They focused on how a piece of information gains attention, i.e., the arrival process of attention (Fig. 1b). Typical examples include Dynamic Poisson Processes [1,6], Reinforced Poisson Processes [9,10,25] and Self-Exciting Hawkes Processes [3,23,32]. Generally speaking, these methods learn a parameterized rate function that reflects the arrival rate of attention over time. Without requiring complex feature engineering, these methods leverage several mechanisms underlying cascade dynamics, including "rich get richer," "survival of the fittest" and aging effect. The disadvantages of these methods are twofold. On the one hand, the form of rate function is hard-coded and context specific. For example, logarithmic normal function is applicable to citation dynamics, while exponential function is often used for modeling retweeting dynamics. On the other hand, these methods only exploit the historical information of the target cascades without the capability of learning from other cascades. In sum, we still lack a fully data-driven method that can automatically learn features from information cascades for outbreak prediction. Recently, several data-driven methods were proposed, including DeepCas [21] and DeepHawkes [5].

In this paper, we propose to use recurrent neural network with long short-term memory (LSTM) [4], namely LSTMIC, to directly learn sequential features from information cascades, working in a fully data-driven manner. Here, we employ LSTM to model the long-term

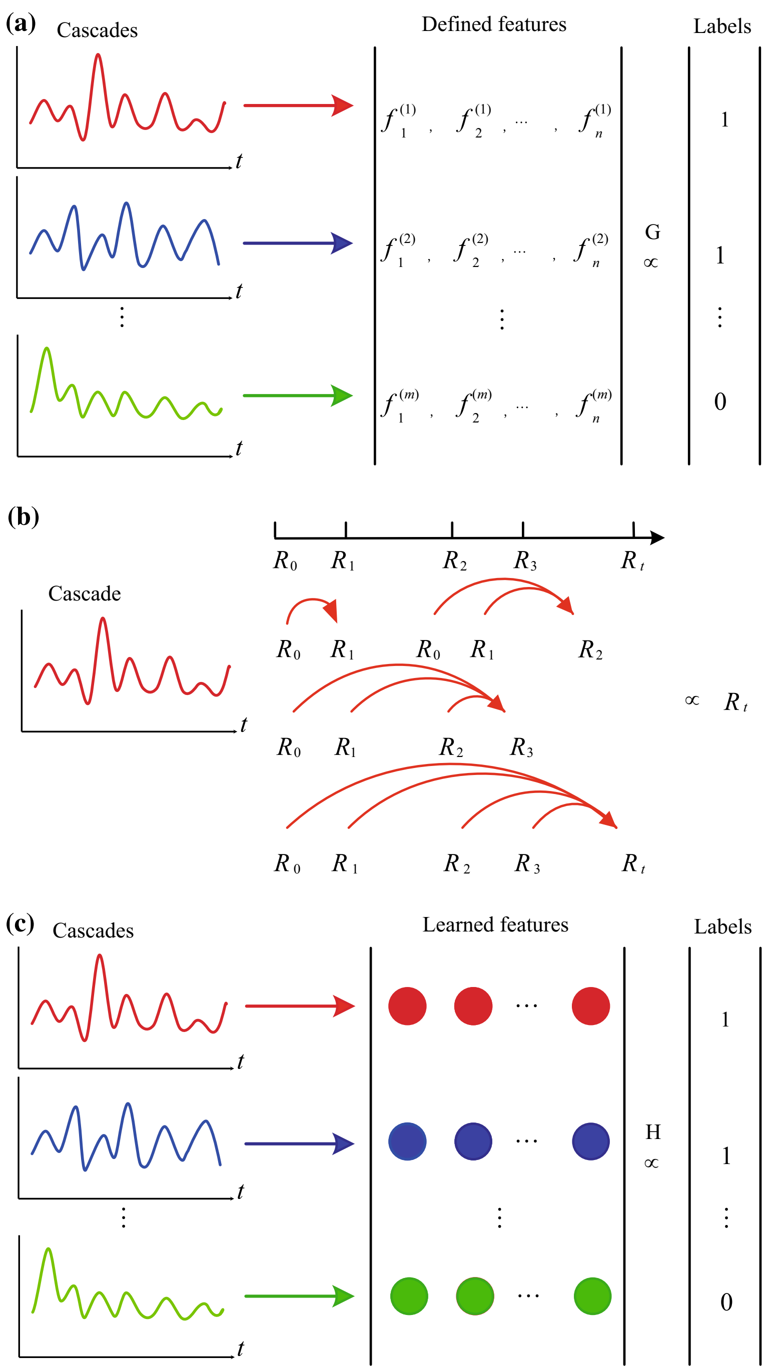

(a)
Cascades
Defined features
Labels
t
$f_1^{(1)}, f_2^{(1)}, \cdots, f_n^{(1)}$
$f_1^{(2)}, f_2^{(2)}, \cdots, f_n^{(2)}$
$f_1^{(m)}, f_2^{(m)}, \cdots, f_n^{(m)}$
G
∝
1
1
0
(b)
Cascade
R_0 R_1 R_2 R_3 R_t
∝ R_t
(c)
Cascades
Learned features
Labels
H
∝
1
1
0

Fig. 1 Three paradigms for cascade prediction. **a** Human-defined features are directly used for outbreak prediction. **b** Human-defined functions are used to model and predict cascade dynamics. **c** Features are learned from cascades, in a fully data-driven manner

dependency among time moments of retweeting in the same information cascade. We call this kind of long-term dependencies in retweeting dynamics as sequential features, such as the “rich get richer” and aging effect [27]. With the learned sequential patterns, the outbreak of cascades could be accurately predicted. The benefits of LSTMIC are twofold: (1) sequential features are learned directly from information cascades with labels, without requiring complex and ineffective feature engineering. This benefit distinguishes our method from existing methods based on human-defined features. (2) Compared with the methods based on stochastic process, LSTMIC need not to designate the form of rate functions underlying retweeting dynamics. This benefit makes our method to be highly flexible and applicable to various domains. We evaluate LSTMIC on Sina Weibo and Twitter datasets. Experimental results demonstrate that our method significantly outperforms state-of-the-art methods [8,24] at the prediction of cascade outbreaks.

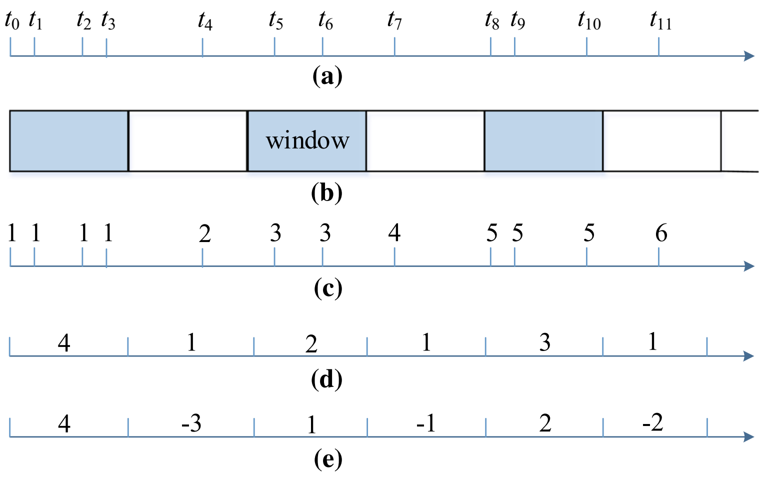
2 Related work

Among the related works [8,15,16], the most relevant to our work is Orthogonal Sparse LOgistic Regression (OSLOR) [8] due to the identical goals. A modified logistic regression with $L1$ and orthogonal regularizations on the behavior vectors of users was exploited to distinguish outbreak cascades. A similar task is outbreak detection [19] rather than prediction; the most common outbreaks detection methods are to locate key node and place sensors there to monitor. The models of popularity prediction can be easily modified for outbreak prediction with a predefined critical threshold. They were divided into two classes. The one class of methods is the machine learning- based methods such as regression, Nonnegative Matrix Factorization (NMF) and survival analysis. The regression methods include Szabo and Huberman (S–H) model [26], Multivariate Linear Regression (MLR) and MRBF [24]. These models are based on the observation that there are strong correlations between early and late popularity of posts [26]. Cui et al. [7] proposed a Hybrid Factor Nonnegative Matrix Factorization (HF-NMF) to model the item-level social influence and can predict the click number of messages. Yu et al. [29] proposed the NEWER model, and it exploits the survival analysis to predict cascade process by modeling behavioral dynamics of social users. The other kind of methods is based on the counting process, which model the retweet actions arrival processes of posts, such as Dynamic Poisson Process [1,6,9], Reinforced Poisson Process [10,25] and Self-Exciting Hawkes Process [3,23,32].

Other relevant works include the researches of rumor detection and the user interaction in cascades, etc. Ma et al. [22] proposed a RNN-based model to detect rumors in social networks, and its research objective is an event, not a single message. Goyal et al. [12] developed a probabilistic model to learn influence between users in social networks. Wang et al. [28] proposed a model to learn user-specific latent influence and susceptibility from information cascades.

3 Problem formulation

The problem focused on in this work is to predict in early stage whether a cascade will break out or not in the future. Suppose there are a total of n information cascades $(c_1, c_2, \ldots, c_n)$. For each cascade c_i, we have its retweeting moments over the time period $[0, T_i]$, characterized by a series of time moments $t_k (0 \leqslant t_k \leqslant T_i)$, as shown in Fig. 2a. Note that all the time



t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10} t_{11}
(a)
window
(b)
1 1 1 1 2 3 3 4 5 5 5 6
(c)
4 1 2 1 3 1
(d)
4 -3 1 -1 2 -2
(e)

Fig. 2 Sequential features in time windows. **a** Raw time series. **b** Time windows. **c** Transformed time series. **d** Frequency series. **e** Slope series

moments mean the length of time relative to the moment when the cascade starts. Indication time T_i refers to when we perform the prediction, and the reference time T_r refers to when we intend to predict the outbreak of a cascade, with $T_i < T_r$. Suppose n_i is the total number of retweets of c_i at T_r, and we say the i-th cascade c_i is an outbreak cascade if $n_i > u$, where u is a predefined threshold. Let $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)$ be the vector of the actual classes of cascades, $y_i \in \{0, 1\}$. Here, $y_i = 1$, if c_i is a outbreak cascade, and $y_i = 0$ otherwise. We have

$$y_i = \frac{1}{2}\{1 + \mathrm{sign}\,(n_i - u)\} \tag{1}$$

where sign(·) is a sign function. For example, if x is positive, sign(x) is equal to 1, otherwise − 1. Let $\hat{\boldsymbol{y}} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n)$ be the predicted vector, and the i-th dimension represents the outbreak probability of the i-th cascade, $\hat{y}_i \in [0, 1]$. The task of outbreak prediction is to predict $\hat{\boldsymbol{y}}$ at T_r. Let $\hat{y}_i = h_{\boldsymbol{\theta}}(\boldsymbol{X}_i)$ be the hypothesis that will be learned, the formulation of $\hat{y}_i$ is given in Eq. (3). The objective is to minimize the cross-entropy loss:

$$\mathop{\mathbf{argmin}}_{\boldsymbol{\theta}} -\frac{1}{n}\sum_{i=1}^{n}(y_i \log(h_{\boldsymbol{\theta}}(\boldsymbol{X}_i)) + (1 - y_i)\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{X}_i))) \tag{2}$$

where $\boldsymbol{\theta}$ is the model parameters. The $\boldsymbol{X}_i$ is the representation of the i-th cascade up to T_i. As shown in Fig. 2, using a fixed-length time window, e.g., 10 s, to discretize the time series, for each cascade we have the following three types of representations.

- *Retweeting time* In this representation, we project the original retweeting time moments into discretized time windows, as shown in Fig. 2c, offering an extension of the original representation of retweeting time moments with the length of time window as a hyper parameter.
- *Frequency* It is the retweet number in each time window, as shown in Fig. 2d. The retweet number is an important features reflecting the rate of accruing attentions of cascades.
- *Slope* The slope in a time window equals to the retweet number in the current window minus that in the previous window. The slope can be viewed as the acceleration of cascade propagation and reflects the trend of cascades.

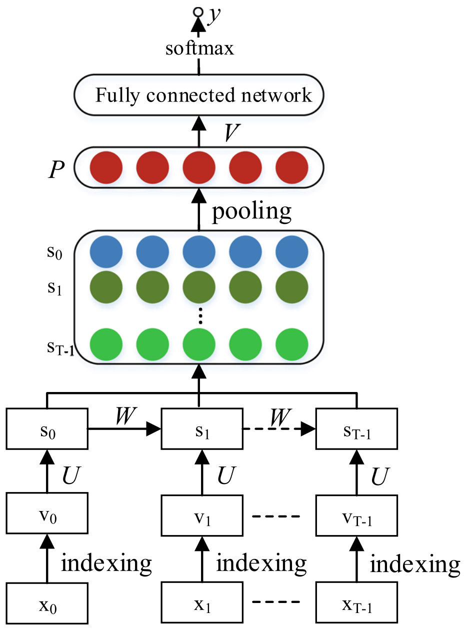


Fig. 3 Architecture of RNN for outbreak prediction, where temporal recurrence is used to capture the long-term dependencies in cascades

All the above feature series characterize the dynamics of cascades diffusion, and the RNN network introduced in Fig. 3 was used to automatically learn the sequential features containing in discrete time features series and identify the outbreak cascades. The differences of the three types of inputs were analyzed hereinafter. We use T to denote the length of the representation of cascade, without differentiating which type of representations is used.

4 LSTMIC model

In this section, we describe the proposed LSTMIC model for outbreak prediction. Before diving into the details of model, we first clarify why we choose the LSTM model to learn the sequential features of retweeting dynamics, and what kind of long-term dependencies are expected to be captured by the LSTMIC model.

4.1 Long-term dependency in cascades

Cascade is generally represented as a series of time moments that reflect when retweets occur. Empirical studies indicate that cascade exhibits high temporal correlation or dependency [26]. Based on these kinds of dependency, linearly or logarithmically, several methods are developed to predict future popularity of cascades based on its early popularity. However, one key problem is still up in the air, i.e., how to model the long-term dependency of time moments in retweeting dynamics. Existing methods either specify certain kind of functions, e.g., exponential function, logarithmic normal function and power-law function, to characterize the temporal dependency, or learn the long-term dependency using multivariate regression models. However, these methods all directly work on time moments themselves or their derivatives, such as time intervals and retweeting counts in a fixed-length window, failing to capture the multi-dimensional correlations. To combat these drawbacks, in this paper, we propose to use recurrent neural network to learn a multi-dimensional representation for each time moment and their long-term dependency. With the learned representation as sequential features, the outbreak of cascades could be accurately predicted.

4.2 Architecture of RNN

A recurrent neural network (RNN) can use their internal memory to process arbitrary sequences of inputs. This makes them applicable to sequential data modeling. In our case, we use RNN to model the series mentioned in Sect. 3. Sequential features of the cascades before indication time T_i will be output as an embedding representation. See Fig. 3, where $x_t(0 \leqslant t < T)$ is the t-th element in $\boldsymbol{X}_i$, $indexing$ is a fixed hash operation that maps x_t to a vector v_t, U is the input weight matrix, W is the recurrent weight matrix, s_t is the hidden-layer state of the RNN, which is used to retain the previous states in a cascade and can be viewed as a cascade representation at time step t, P is the aggregated representation by pooling operations after the s_t of each step t has been calculated. The pooling operation is average pooling, max pooling or last pooling. The pooled features are fed into a fully connected network layer for classification. The formulation of the above RNN model for outbreak prediction can be expressed as

$$\begin{aligned} v_t &= indexing(x_t) \\ s_t &= f(Uv_t + Ws_{t-1}) \\ P &= pooling(s_0, s_1, \ldots, s_{T-1}) \\ \hat{y}_i &= softmax(PV) \end{aligned} \tag{3}$$

where the U, W and V are the input, recurrent and output layers parameters matrices to be learned, v_t is the embedding representation of x_t and it is initialized randomly and updated in the phase of training, $f(\cdot)$ is an activation function such as $sigmoid(\cdot)$, $\tanh(\cdot)$ and $ReLU(\cdot)$. The LSTM [14] is used to combat the vanishing gradients problem.

4.3 Pooling operation

The pooling operation is a transformation of the hidden-layer states. The results of pooling operation can be viewed as a representation of cascades. It contains the high-level abstract sequential patterns which are beneficial for distinguishing the outbreak cascades. Let s_t be the vector of hidden layers' outputs at time step t. Specifically, there are three types of pooling operations used in this work: average pooling, max pooling, and last pooling.

– *Average pooling* Let $S = \{\vec{s_t} \in \mathbb{R}^n | t = 0, 1, \ldots, T-1\}$ be a set of input vectors to be operated, and average pooling over S can be formalized as

$$f_a(S) = \frac{1}{T} \sum_{t=0}^{T-1} \vec{s_t}$$

Note that average pooling is a linear operation, which supposes the independence among input vectors.

- *Max pooling* To aggregate a set of vectors, max pooling builds one vector by selecting the maximum value of each dimension. It can be formalized as

$$f_m(S) = \begin{bmatrix} \max(\vec{s}_0[1], \ldots, \vec{s}_{T-1}[1]) \\ \max(\vec{s}_0[2], \ldots, \vec{s}_{T-1}[2]) \\ \vdots \\ \max(\vec{s}_0[n], \ldots, \vec{s}_{T-1}[n]) \end{bmatrix}$$

  where $\vec{s}_t[k]$ denotes the k-th dimension in $\vec{s}_t$, max pooling is a nonlinear operation in contrary to the average pooling.
- *Last pooling* Last pooling selects the hidden-layer state vector of the final time step. It can be simply formalized as

$$f_l(S) = \vec{s}_{T-1}$$

  $\vec{s}_T$ maintains the information of previous steps, and it can be viewed as the cascade embedding representation by time moment $T-1$.

4.4 Training details

LSTMIC was trained through stochastic gradient descent with the Adadelta update rule [31] over shuffled mini-batches. The size of mini-batches in training sets is 16. Momentum or weight decay was not used. The initial parameters were sampled from a univariate Gaussian distribution with mean 0 and variance 1, and the singular value decomposition (SVD) was applied to the orthogonal initialization of weight matrices. For datasets, we randomly selected 80% data as the training set and 20% data as the test set. In the training set, we randomly selected 10% data as the validation set. To reduce overfitting, we applied dropout [13] to the penultimate layer at training time. The major time cost of LSTMIC is the RNN network. The BPTT is used to train the RNN. Let k be the training epochs, L is the length of series, M is the truncated steps, H is the nodes number of hidden layer, I is the length of time vectors, K is the iteration times in training phase. Then the time complexity of forward pass is $O(IH + HH) * L$, and the time complexity of backward pass is $O((2IH + HH)ML * (sample\_num/bachsize) * K)$, and we can see that the training time is the major cost in our model.

5 Experiments

In this section, we described the experimental datasets, analyzed the effect of parameters, compared the performance of LSTMIC with the state-of-the-art approaches and discussed why the LSTMIC worked.

5.1 Data description and data processing

We evaluated the proposed method on two datasets. The Twitter dataset[1] [32] contains 166,076 tweets and retweets on Twitter from October 7 to October 21, 2011. The 15 days were divided into two parts, the first 7 days were used for training and the next 8 days were used for test. There were 94,256 tweets in test set. Without loss of generality, top 2% tweets were selected as outbreak tweets according to their final retweets and the minimum retweet

[1] http://snap.stanford.edu/seismic/.

number of outbreak tweets was selected as outbreak threshold. Additionally, a retweet time gap was defined to distinguish the outbreak tweets and normal tweets, which was an order of magnitude smaller than the outbreak threshold. Therefore, in Twitter dataset, there were 1801 outbreak tweets, the outbreak threshold was 1000 and the gap was 100. It meant that the tweets whose retweet number less than 900 were seen as non-outbreak tweets. To avoid imbalanced data, we randomly select 1801 non-outbreak tweets as negative samples.

The Sina Weibo dataset was published by WISE 2012 Challenge.[2] It was collected from the Sina Weibo via the provided API, and the content of tweets was removed according to Sina Weibo's Terms of Services. The tweets from August 14, 2009, to December 31, 2010, were extracted for experiments. There were about 0.31 million users, 13 million relations among users and 1.45 million tweets in the data. We used the first 11 months' posts as the training and test data. The remaining 1 month was used for the retweet cascades to unfold. The tweets without retweet were removed, and there were 237,489 remaining tweets. For Sina Weibo dataset, there were 409 outbreak tweets, the outbreak threshold was 150 and the gap was 10. The tweets whose retweet number less than 140 were seen as non-outbreak tweets. The retweet series of cascades were attained by the "rtMid" tag in records. Four hundred and nine non-outbreak tweets were randomly selected as negative samples. The characteristics of the Sina Weibo dataset were analyzed similarly as [32].

Finally, the retweet time series and corresponding users series of all tweets on both datasets were extracted. Note that the Twitter dataset did not provide the user identification and there were only retweet time series for experiments.

5.2 Baselines for comparison

Various methods that might be used for outbreak prediction were considered. The first three are the regression based, the 4th is node selection based and the 5th is the point process based. If the methods predict the exact retweet number of a tweet, a predefined threshold is set to decide the class of tweets.

- Univariate linear regression (LR) [26]: it is based on the observation that the log-transformed popularity of a given post is strongly correlated with its early popularity. The model can be defined as

$$\ln N_p(T_r) = \beta + \ln N_p(T_i) + \xi,$$

  where $N_p(T_r)$ is the popularity of post p at time T_r, $N_p(T_i)$ is the popularity of post p at time T_i, and ξ denotes the Gaussian noise.
- Multivariate linear regression (MLR) [24]: the model can be defined as

$$N_p(T_r) = \Theta \cdot X_p,$$

  where $\Theta = (\theta_1, \theta_2, \ldots, \theta_{T_i})$ is the vector of model parameters. X_p is the feature vector, defined as

$$X_p = (x_p(1), x_p(2), \ldots, x_p(T_i))^T,$$

  $x_p(i)$ is the number of retweets at the i-th time interval. The model allows to assign different weights to each time interval compared with S–H model.
- MRBF [24] mode: it is a variant of the MLR model and is defined as

$$N_p(T_r) = \Theta \cdot X_p + \sum_{p_c \in C} \omega_{p_c} \cdot \mathrm{RBF}_{p_c}(p),$$

[2] http://www.wise2012.cs.ucy.ac.cy/challenge.html.

where C is the training examples set chosen as centers and ω_{p_c} is the weight associated with Gaussian radial basis function (RBF) feature for p_c. The RBF is defined as

$$\text{RBF}_{p_c}(p) = e^{\left(-\frac{\|X(p)-X(p_c)\|^2}{2\cdot\sigma^2}\right)},$$

The RBF feature is used to capture some particular patterns of certain types of posts.

- Orthogonal Sparse LOgistic Regression (OSLOR) [8] model: It is a variant of logistic regression and jointly optimizes node selection and outbreak prediction. The loss are a combination of orthogonal and $L1$ regularizations. The model can be defined as

$$h(X_{i\cdot}^t) = \frac{1}{1+\exp(-\theta_0 - X_{i\cdot}^t\theta)},$$

where θ is the weight vector, X^t is the cascade status matrix and $X_{i\cdot}^t$ is the i-th cascade status at time t. It aims to minimize the following objective:

$$F(\theta) = -\log L(\theta) + \frac{\beta}{4}\sum_{i,j}(\theta_i X_{\cdot i}^T X_{\cdot j}\theta_j)^2 + \gamma\,\|\theta\|_1\,,$$

here $L(\theta)$ is the likelihood function of $h(X_{i\cdot}^t)$. The $L1$ regularization is to limit the powerful users, and the orthogonal regularization is to make the behaviors of the powerful users complementary.

- Self-Exciting Model of Information Cascades (SEISMIC) [32] model: it is an extension to the Hawkes process, and the intensity λ_t of the process R_t at time t is defined as

$$\lambda_t = p_t \cdot \sum_{t_i \leqslant t, i \geqslant 0} n_i \phi\,(t - t_i)\,, \quad t \geqslant t_0.$$

where p_t is infectiousness, t_i is retweeting time, n_i is node degree and $\phi(s)$ is the distribution function of human reaction time. p_t is itself a stochastic process. The maximum likelihood estimate (MLE) of p_t after smoothing is

$$\hat{p}_t = \frac{\sum_{i=1}^{R_t} K_t(t - t_i)}{\sum_{i=0}^{R_t} n_i \int_{t_i}^{t} K_t(t-s)\phi(s-t_i)\mathrm{d}s},$$

here $K_t(s)$, $s > 0$, is a kernel function for smoothing.

5.3 Evaluation metrics

Outbreak prediction is a binary classification problem. Therefore, we used Precision, Recall and $F1$ to evaluate the performances of the proposed method and baselines. Let χ denote the set of testing samples, with the notations in Sect. 3, we have

$$\text{Precision} = \frac{\sum_{i\in\chi} y_i \cdot \hat{y}_i}{\sum_{i\in\chi} \hat{y}_i}$$

$$\text{Recall} = \frac{\sum_{i\in\chi} y_i \cdot \hat{y}_i}{\sum_{i\in\chi} y_i}$$

$$F1 = \frac{2\cdot\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}}$$

5.4 Experiment results

Before comparing the proposed method with baselines, we first determined the settings for the proposed methods by evaluating the effect of the size of time windows, the dimension of time vectors and the pooling operations, with the input representation of retweeting time.

5.4.1 *Effect of time window length*

First, we evaluated the effect of time window length. The window length determined the final number of unique tokens. Shorter window lengths mean more fine-grained token representations, sparser data and more complex models. The choice of window length was a trade-off between model complexity and data sparsity. We varied the length of windows from 1 to 60 s, with the indication time T_i being 1 h and the dimension of time vector being as 10. As shown in Fig. 4, for each dataset, the performance of LSTMIC was insensitive to the length of time windows, reflecting the scale invariance of information cascades.

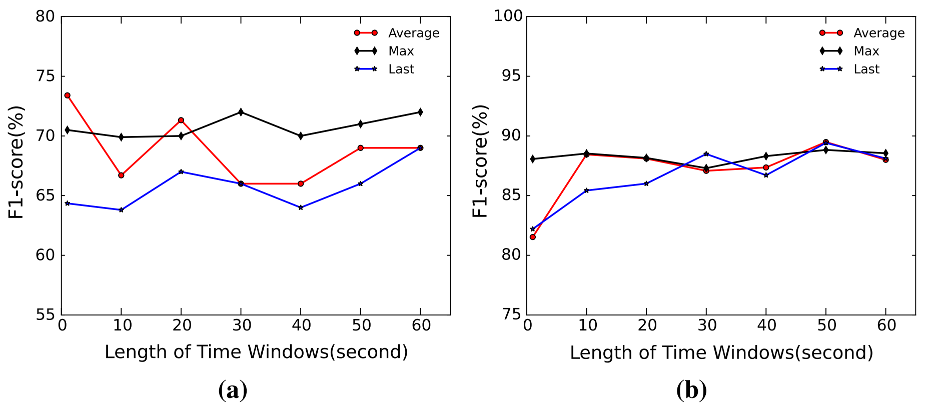


Fig. 4 Effect of the length of time windows and pooling operations. **a** $F1$-score@Weibo. **b** $F1$-score@Twitter

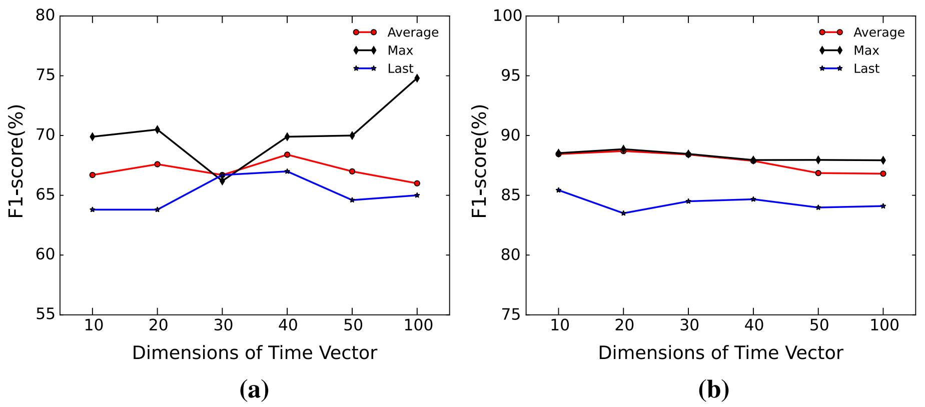


Fig. 5 Effect of time vector length and pooling operations. **a** $F1$-score@Weibo. **b** $F1$-score@Twitter

Table 1 Performance evaluation on Weibo and Twitter data, $L(*)$ represents LSTMIC with input type $*$, including slope, freq, time

Method	Weibo (%)			Twitter (%)		
	Prec	Recall	$F1$	Prec	Recall	$F1$
LR	47.76	100	64.61	81.54	93.13	86.94
MLR	100	32.53	48.94	97.86	72.06	82.97
MRBF	97.38	32.85	48.93	97.82	72.1	82.99
OSLOR	49.26	91.98	64.08	–	–	–
SEISMIC	47.76	100	64.61	50.44	99.56	66.94
L(slope)	77.59	57.69	66.17	80.9	84.23	82.54
L(freq)	76.67	58.97	66.67	83.29	81.5	82.38
L(time)	75	65.38	**69.86**	92.11	87.17	**89.49**

The bold values indicate significance level is 0.05

5.4.2 Effect of time vector dimension

Figure 5a, b shows the effect of time vector dimension on the LSTMIC. The number of time vector dimension increased from 10 to 60, T_i is set to 1 h and the length of time windows was set to 10. The length of time vector did not have a major impact on the performance of $F1$ score. As the increase in the vector length, the time vector has enough even excess spaces to save the cascade features; however, the model complexity is significantly increased and might cause the problem of overfitting.

5.4.3 Effect of pooling operation

Next, we investigated the pooling operations. Among three pooling operations, the max pooling had best mean performance, as depicted in both Figs. 4 and 5. It was reasonable that both the average pooling and last pooling were linear operations on the inputs, while the max pooling was a nonlinear operation. Moreover, the last pooling was only dependent on the last hidden-layer state, which did not capture the cascades diffusion patterns very well even with LSTM RNN. The average pooling equally treated all hidden-layer states, whereas the max pooling could learn the important locations of hidden-layer states. In summary, 10-s long time window with max pooling had good performance on both data; therefore, these parameters were used for the following analysis.

5.4.4 Comparison with baselines

We firstly determined the parameters used in baseline methods. For the MLR and MRBF models, the length of time interval was set to 10 min, the total number of retweets was sampled at the regular intervals up to the time T_i, which could be seen as the retweets number deltas up to T_i. The number of centers which were used to compute the Gaussian RBF features was 5. The OSLOR model was implemented only on Weibo data because user identification was not available in Twitter data. The number of involved users has a significant impact on the time complexity of OSLOR. Therefore, only the users involving in no less than 3 cascades were considered, leaving us 237 unique users. The max iterative times were 100. For the LSTMIC model, the length of time window and vector dimension was set to 10. The max iterative times was 100. The max pooling was selected because of its excellent performance with various parameters on both datasets. We first set the indication time T_i to 1 h and will discuss the variant T_i later.

LSTMIC with different inputs were, respectively, denoted as L(slope), L(freq) and L(time). As listed in Table 1, L(time) outperformed all the traditional methods measured by $F1$. In Weibo data, the strongest competitors were LR and OSLOR with $F1$ 64.61%. The performance of SEISMIC was comparable to them, SEISMIC was selected as a baseline because it is the state of the art in the stochastic process-based models for popularity prediction. Although SEISMIC explicitly modeled the retweet time series of diffusion processes, its performances highly depended on the statistical characteristics of data, e.g., the memory kernel function $\phi(s)$. In Twitter data, the strongest competitor was also LR and OSLOR with $F1$ 86.94%, verifying that the early popularity and involved users were useful features for outbreak prediction. In addition, the $F1$ scores of methods on Twitter were far more than that on Weibo, and it was because that the Twitter data excluded the tweets whose retweets number was less than 50.

Among all the competitors, the view pattern, which was one type of sequential patterns, had been implicitly considered in the MLR and MRBF. The MLR, a simplified version of MRBF, did not capture all types of outbreak cascades. Therefore, as an important supplement for view pattern, the Gaussian kernel distance was applied to capture the diverse features of individuals. The $F1$ score of MRBF was 82.99% in Twitter and 48.93% in Weibo. However, the time interval length was a difficult parameter to select, and the sequential patterns might be different in variant time interval length. To summarize, the proposed LSTMIC demonstrated an obvious improvement over the baselines. It showed that LSTMIC was able to learn more accurate cascades features by itself and adapt to different scenes.

5.4.5 Analysis

In this section, the performances of LSTMIC were explored in detail.

1. We first demonstrated that LSTMIC had the ability to learn the sequential features, e.g., the rise and fall of a curve. Synthetic data with special shapes were generated from the function curves, $y = x^2$ and $y = -x^2 + 100$, $x \in [-10, 10]$ and x is an integer. The $x + 11$ were seen as time windows number, and the y were seen as the retweets number in corresponding windows. The y were multiplied with random variables to generate more samples, the random perturbations were sampled from uniform$(1, 2)/10$, where uniform represented uniform distribution. 1000 series like x^2 and 1000 series like $-x^2 + 100$ were generated. Then, three types of inputs were constructed on the artificial series. LSTMIC was used to classify the two classes of curves.

Table 2 compares the performance of three types of input series on the synthetic data, uniform(a, b) meant sampling uniformly between a and b. The only difference between two synthetic data was the range of stochastic perturbations. For convenience, uniform$(1, 2)$ and uniform$(1, 5)$ referred to the two data, respectively. Firstly, LSTMIC was good for learning the shape of a curves from all types of input series, with the $F1$ scores greater than 90%. The curve shape underline the input series was seen as a long-term dependency, and it could characterize the dynamics of cascade diffusion. Secondly, among the three types of inputs, the time series obtained the best results on both synthetic data. Next, we investigated what caused the different results with three equivalent inputs. The number of tokens in series was stated, respectively, i.e., 21, 50 and 60 for time, frequency and slope series on uniform$(1, 5)$ data. The more the tokens were, the sparser the series were. Therefore, LSTMIC might not be able to learn the tokens representations well in sparse series data.

To prove the assumption, for every token, the absolute of difference of other tokens was used to rank the distances of tokens; for each token' representation, the Euclidean distances

Table 2 Performance on different input series

Series	Uniform(1, 2)/10 (%)			Uniform(1, 5)/10 (%)		
	Prec	Recall	$F1$	Prec	Recall	$F1$
Time	100	100	100	100	100	100
Freq	92.04	100	95.85	93.4	91.24	92.31
Slope	100	100	100	91	91.83	91.39

Table 3 Pearson's r between tokens and representations

	Time	Freq	Slope
Pearson	0.127	– 0.083	0.013

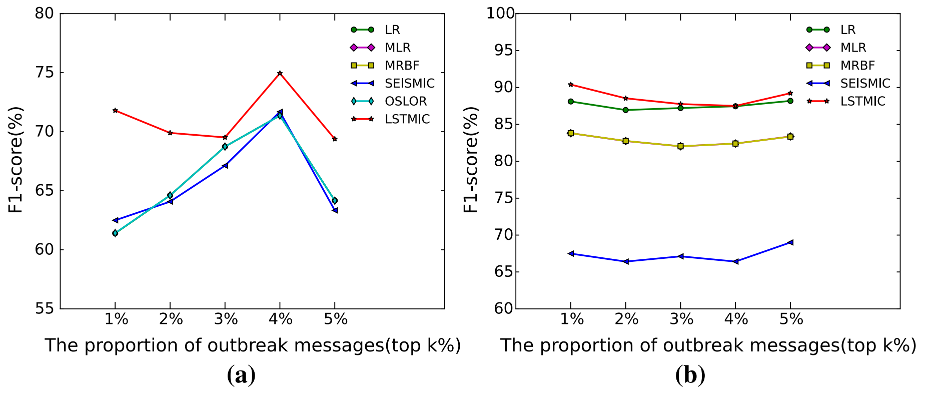


Fig. 6 $F1$-score with varying threshold. **a** $F1$-score@Weibo. **b** $F1$-score@Twitter

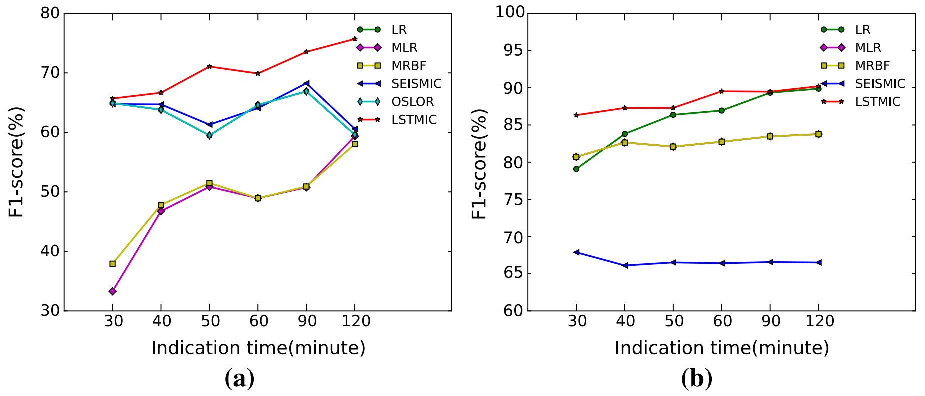
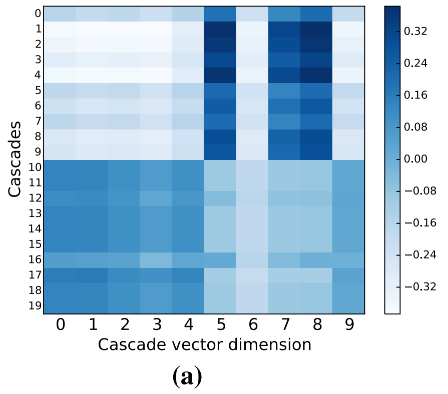


Fig. 7 Effect of indication time **a** $F1$-score@Weibo. **b** $F1$-score@Twitter

were used to measure its similarity of other tokens. The Pearson's r was used to compute the correlation between two rankings. As listed in Table 3, the time tokens had the highest correlation coefficient, the coefficient was the mean of all tokens in each series.

2. Effects of the outbreak threshold. The threshold is a key for model training, by which the outbreak messages and non-outbreak messages are divided. The top $k\%$ retweeting



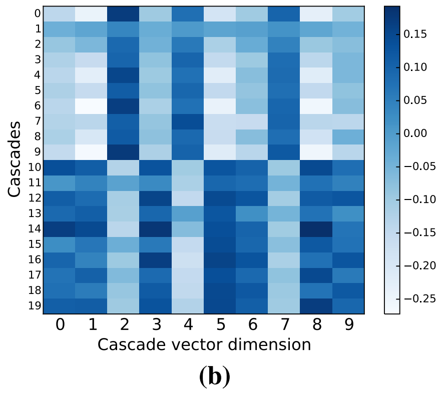


Fig. 8 Learned representations of cascades **a** Weibo. **b** Twitter

messages were reviewed as outbreak messages. The k was varied from 1 to 5. Figure 6 shows that the LSTMIC had better performance on different thresholds. In addition, the performance of LSTMIC was stable compared with other models.

3. Effects of the indication time. Figure 7 shows the $F1$ scores with variant indication time on two datasets. The proposed LSTMIC outperformed other competitors with variant indication time on each dataset, which demonstrated the robustness of LSTMIC. As the increase in the indication time T_i, the $F1$ scores of LSTMIC were increasing, which proved it was able to learn the fine-grain features in different datasets. Specially, LSTMIC had good performances at the very early stage of cascade diffusion. In summary, LSTMIC has a consistent performance on different scenes and is robust along with variant time indication. In particular, LSTMIC specializes in early warning of outbreak cascades.
4. Visualization of cascades representations. The outputs of pooling operation were viewed as the cascade representations. We used visualization tools to see whether our model had learned discriminative features of cascades. As shown in Fig. 8, the vertical axis was the cascade index from 0 to 19, the top 10 rows were vector representations of outbreak cascades, the bottom 10 rows were non-outbreak cascades, and the horizontal axis was the dimension index of cascade vectors, which were numbered from 0 to 9, and the color codes showed the values. The cascades vectors learned by LSTMIC had a distinct division between outbreak and non-outbreak cascades, e.g., the 5th, 7th and 8th dimensions of outbreak cascade vectors in Weibo had darker colors than these of non-outbreak cascades.

6 Conclusions and future work

In this paper, we proposed to use recurrent neural network with long short-term memory to predict cascade outbreak. Different from existing methods that depend on human-defined features or functions, the proposed method is a fully data-driven method, where sequential features are automatically learned from information cascades to characterize the long-term dependency in cascades. With the learned sequential patterns, the outbreak cascades could be accurately predicted. Extensive experimental results on Sina Weibo and Twitter datasets demonstrate that our method significantly outperforms state-of-the-art methods at the prediction of cascade outbreak. As future work, we will investigate the potential connection between the LSTMIC model with stochastic process-based methods, further exploring its potentials at modeling retweeting dynamics with both high prediction accuracy and good explanations.

Acknowledgements This work was funded by the National Basic Research Program of China (973 Program) under Grant Nos. 2014CB340401 and 2013CB329602, the National Natural Science Foundation of China under Grant Nos. 61425016, 61472400, 61572467, 61433014, and the National High Technology Research and Development Program of China (863 Program) under Grant Nos. 2014AA015204 and 2015AA015803, the Key Technologies R&D Program under Grant No. 2017YFB0803302. H. W. Shen is also funded by Youth Innovation Promotion Association CAS.

References

1. Agarwal D, Chen B-C, Elango P (2009) Spatio-temporal models for estimating click-through rate. In: WWW, vol 6, pp 21–30
2. Bao P, Shen H, Huang J et al (2013) Popularity prediction in microblogging network: a case study on sina weibo. In: WWW, pp 117–118
3. Bao P, Shen H-w, Jin X et al (2015) Modeling and predicting popularity dynamics of microblogs using self-excited hawkes processes. In: WWW, pp 9–10
4. Bengio Y, Simard P, Frasconi P (1994) Learning long-term dependencies with gradient descent is difficult. IEEE Trans Neural Netw 5(2):157–166
5. Cao Q, Shen H, Cen K et al DeepHawkes: Bridging the gap between prediction and understanding of information cascades. In: Proceedings of the 26th ACM conference on information and knowledge management (CIKM 2017). pp 1149–1158
6. Crane R, Sornette D (2008) Robust dynamic classes revealed by measuring the response function of a social system. PNAS 105(41):15649–15653
7. Cui P, Wang F, Liu S et al (2011) Who should share what? Item-level social influence, pp 185–194
8. Cui P, Jin S, Yu L et al (2013) Cascading outbreak prediction in networks: a data-driven approach. In: KDD, pp 901–909
9. Gao J, Shen H, Liu S et al (2016) Modeling and predicting retweeting dynamics via a mixture process, In: Proceeding WWW '16 Companion Proceedings of the 25th International Conference Companion on World Wide Web, pp 33–34
10. Gao S, Ma J, Chen Z (2015) Modeling and predicting retweeting dynamics on microblogging platforms. In: WSDM, pp 107–116
11. Golub B, Jackson MO (2010) Using selection bias to explain the observed structure of internet diffusions. PNAS 107(24):10833–10836
12. Goyal A, Bonchi F, Lakshmanan LV (2010) Learning influence probabilities in social networks. In: WSDM, p 241
13. Hinton GE, Srivastava N, Krizhevsky A et al (2012) Improving neural networks by preventing co-adaptation of feature detectors. In: CORR, arXiv:1207.0580
14. Hochreiter S, Schmidhuber JJ (1997) Long short-term memory. Neural Comput 9(8):1–32
15. Hong L, Dan O, Davison BD (2011) Predicting popular messages in Twitter. In: WWW, pp 57–58
16. Kupavskii A, Ostroumova L, Umnov A et al (2012) Prediction of retweet cascade size over time. In: CIKM, pp 2335–2338
17. Kupavskii A, Umnov A, Gusev G et al (2013) Predicting the audience size of a tweet. In: ICWSM, pp 693–696
18. Leskovec J, Adamic LA, Huberman BA (2007) The dynamics of viral marketing. ACM Trans Web 1(1):5–es
19. Leskovec J, Krause A, Guestrin C et al (2007) Cost-effective outbreak detection in networks. In: SIGKDD, pp 420–429
20. Leskovec J, McGlohon M, Faloutsos C et al (2007) Cascading behavior in large blog graphs. In: SDM, pp 9:3–9:56
21. Li C, Ma J, Guo X et al. DeepCas: An end-to-end predictor of information cascades. In: Proceedings of the 26th international conference on world wide web companion (WWW '17). pp 577–586

22. Ma J, Gao W, Mitra P et al (2016) Detecting rumors from microblogs with recurrent neural networks detecting rumors from microblogs with recurrent neural networks. In: Proceedings of the 25th international joint conference on artificial intelligence (IJCAI 2016), pp 3818–3824
23. Mishra S, Rizoiu M-A, Xie L (2016) Feature driven and point process approaches for popularity prediction. In: Proceedings of international conference on information and knowledge management-CIKM '16, p 10
24. Pinto H, Almeida JM, Gonçalves MA (2013) Using early view patterns to predict the popularity of youtube videos. In: WSDM, pp 365–374
25. Shen H, Wang D, Song C et al (2014) Modeling and predicting popularity dynamics via reinforced poisson processes. In: AAAI, no 3, pp 291–297
26. Szabo G, Huberman BA (2010) Predicting the popularity of online content. Commun ACM 53(8):80–88
27. Wang D, Song C, Barabási A-L (2013) Quantifying long-term scientific impact. Science 342(6154):127–159
28. Wang Y, Shen H, Liu S et al (2015) Learning user-specific latent influence and susceptibility from information cascades. In: Proceedings of the 29th AAAI conference on artificial intelligence, no 3, pp 477–483
29. Yu L, Cui P, Wang F et al (2015) From micro to macro: uncovering and predicting information cascading process with behavioral dynamics. In: Proceedings of the IEEE international conference on data mining, pp 559–568
30. Zaman TR, Herbrich R, Stern D (2010) Predicting information spreading in twitter. In: Workshop on computational social science and the wisdom of crowds, NIPS, vol 104, pp 17599–17601
31. Zeiler MD (2012) ADADELTA: an adaptive learning rate method. arXiv preprint arXiv:1212.5701
32. Zhao Q, Erdogdu MA, He HY et al (2015) SEISMIC: a self-exciting point process model for predicting tweet popularity. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining-KDD '15, pp 1513–1522

Chengcheng Gou PhD, was born in 1985. His main research interests include web search and mining, machine learning and social network.

Huawei Shen PhD, was born in 1980. His main research interests include web search and mining, machine learning and social network.

Pan Du PhD, was born in 1981. His main research interests include web search and mining, machine learning and social network.

Dayong Wu PhD, was born in 1977. Her main research interests include natural language process, web mining and machine learning.

Yue Liu PhD, was born in 1971. She is currently an associate professor. Her main research interests include information retrieval and web mining.

Xueqi Cheng PhD, was born in 1971. He is currently a professor and PhD supervisor in the Institute of Computing Technology, Chinese Academy of Sciences. His major research interests include network information security, large-scale information retrieval and knowledge mining.