Neural Diffusion Model for Microscopic Cascade Study

Cheng Yang, Maosong Sun, Haoran Liu, Shiyi Han, Zhiyuan Liu, Huanbo Luan

Abstract—The study of information diffusion or cascade has attracted much attention over the last decade. Most related works target on studying cascade-level *macroscopic* properties such as the final size of a cascade. Existing *microscopic* cascade models which focus on user-level modeling either make strong assumptions on how a user gets infected by a cascade or limit themselves to a specific scenario where "who infected whom" information is explicitly labeled. The strong assumptions oversimplify the complex diffusion mechanism and prevent these models from better fitting real-world cascade data. Also, the methods which focus on specific scenarios cannot be generalized to a general setting where the diffusion graph is unobserved.
To overcome the drawbacks of previous works, we propose a Neural Diffusion Model (NDM) for general microscopic cascade study. NDM makes relaxed assumptions and employs deep learning techniques including attention mechanism and convolutional network for cascade modeling. Both advantages enable our model to go beyond the limitations of previous methods, better fit the diffusion data and generalize to unseen cascades. Experimental results on diffusion identification task over four realistic cascade datasets show that our model can achieve a relative improvement up to 26% against the best performing baseline in terms of F1 score.

Index Terms—Information Diffusion, Neural Network

1 Introduction

Information diffusion is a ubiquitous and fundamental event in our daily lives, such as the spread of rumors, the contagion of viruses and the propagation of new ideas and technologies. The diffusion process, also called a *cascade*, has been studied over a broad range of domains. Though some works believe that even the eventual size of a cascade cannot be predicted [1], recent works [2], [3], [4] have shown the ability to estimate the size, growth and many other key properties of a cascade. Nowadays the modeling of cascades play an important role in many real-world applications, *e.g.* production recommendation [5], [6], [7], [8], [9], epidemiology [10], [11], social networks [12], [13], [14] and the spread of news and opinions [15], [16], [17]. Most previous works focus on the study of *macroscopic* properties such as the total number of users who share a specific photo [2] and the growth curve of the popularity of a blog [3]. However, macroscopic cascade study is a rough estimate of cascades and cannot be adapted for *microscopic* questions as shown in Fig. 1. Microscopic cascade study, which pays more attention to user-level modeling instead of cascade-level, is much more powerful than macroscopic estimation and allows us to apply user-specific strategies for real-world applications. For example, during the adoption of a new product, microscopic cascade study can help us deliver advertisements to those users that are most likely to buy the product at each stage. In this paper, we focus on the study of microscopic level.

Fig. 1. Macroscopic cascade study v.s. microscopic cascade study.

Though useful and powerful, the microscopic study of cascades faces great challenges because the real-world diffusion process could be rather complex [18] and usually partially observed [11], [19]:

Complex mechanism. Since the mechanism of how a specific user gets infected [1] is sophisticated, traditional cascade models based on strong assumptions and simple formulas may not be the best choice for microscopic cascade modeling. Existing cascade models [20], [21], [22], [23] which could be adopted for microscopic analysis mostly ground in Independent Cascade (IC) model [12]. IC model

- *Cheng Yang is with Beijing University of Posts and Telecommunications and also with the Department of Computer Science and Technology, Tsinghua University.*
  E-mail: albertyang33@gmail.com
- *Maosong Sun (corresponding author), Zhiyuan Liu and Huanbo Luan are with the Department of Computer Science and Technology, Tsinghua University, Beijing, China.*
  {sms,liuzy}@mail.tsinghua.edu.cn, luanhuanbo@gmail.com
- *Haoran Liu is with the Department of Electric Engineering, Tsinghua University, Beijing, China.*
  E-mail: liu-hr15@mails.tsinghua.edu.cn
- *Shiyi Han is with the Department of Computer Science, Brown University, U.S.A.*
  E-mail: hanshiyi123@gmail.com

1. We use "infected" and "activated" alternatively to indicate that a user is influenced by a cascade.

assigns a static probability $p_{u,v}$ to user pairs (u, v) with pairwise independent assumptions, where the probability $p_{u,v}$ indicates how likely user v will get infected by user u when u is infected. Other diffusion models [24], [25] make even stronger assumptions that the infected users are only determined by the source user. Though intuitive and easy to understand, these cascade models are based on strong assumptions and oversimplified probability estimation formulas, both of which limit the expressivity and ability to fit complex real-world cascade data [26]. The complex mechanism of real-world diffusions encourages us to explore more sophisticated models, *e.g.* deep learning techniques, for cascade modeling.

Incomplete observation. On the other hand, the cascade data is usually partially observed indicates that we can only observe those users getting infected without knowing who infected them. However, to the best of our knowledge, existing deep-learning engined microscopic cascade models [27], [28] are based on the assumption that the diffusion graph where a user can only infect and get infected by its neighbors is already known. For example, when we study the retweeting behavior on the Twitter network, "who infected whom" information is explicitly labeled in retweet chain and the next infected user candidates are restricted to the neighboring users rather than the whole user set. While in most diffusion processes such as the adoption of a product or the contamination of a virus, the diffusion graph is unobserved [11], [19], [29]. Therefore, these methods consider a much simpler problem and cannot be generalized to a general setting where the diffusion graph is unknown.

To fill in the blank of general microscopic cascade study and address the limitations of traditional cascade models, we propose a neural diffusion model based on relaxed assumptions and employ up-to-date deep learning techniques, *i.e.* attention mechanism and convolutional neural network, for cascade modeling. The relaxed assumptions enable our model to be more flexible and less constrained, and deep learning tools are good at capturing the complex and intrinsic relationships that are hard to be characterized by hand-crafted features. Both advantages allow our model to go beyond the limitations of traditional methods based on strong assumptions and oversimplified formulas and better fit the complicated cascade data. Following the experimental settings in [23], we conduct experiments on diffusion identification task over four realistic cascade datasets to evaluate the performances of our proposed model and other state-of-the-art baseline methods. Experimental results show that our model can achieve a relative improvement up to 26% against the best performing baseline in terms of F1 score.

To conclude, our contributions are 3-fold:

- To the best of our knowledge, our work is the first attempt to employ deep learning techniques for general microscopic cascade study where the diffusion graph is unknown.
- We design a neural diffusion model based on relaxed assumptions compared with the pairwise independence assumption in traditional cascade models and allow our model to better fit real-world cascades and generalize to unseen data.
- Experimental results on diffusion identification task over four realistic datasets demonstrate the effectiveness and robustness of our proposed model. Compared with the best performing baseline, our model can achieve a relative improvement up to 26% on F1 score.

2 Related Works

We organize related works into macroscopic and microscopic cascade studies. In terms of methodology, our work is also related to network representation learning methods.

2.1 Macroscopic Cascade Study

Most previous works focused on macroscopic level estimation such as the eventual size of a cascade [4] and the growth curve of popularity [3]. Macroscopic cascade study methods can be further classified into feature-based approaches, generative approaches, and deep-learning based approaches. Feature-based approaches formalized the task as a classification problem [2], [30] or a regression problem [31], [32] by applying SVM, logistic regression and other machine learning algorithms on hand-crafted features including temporal [33] and structural [2] features. Generative approaches considered the growth of cascade size as an arrival process of infected users and employed stochastic processes, such as Hawkes self-exciting point process [4], [34], for modeling. With the success of deep learning techniques in various applications, deep-learning based approaches, e.g. DeepCas [26] and DeepHawkes [35], were proposed to employ Recurrent Neural Network (RNN) for encoding cascade sequences into feature vectors instead of hand-crafted features. Compared with hand-crafted feature engineering, deep-learning based approaches have better generalization ability across different platforms and give better performance on evaluation tasks.

2.2 Microscopic Cascade Study

Our work is more related to microscopic cascade study which focuses on user-level modeling. We classify related works into three categories: IC-based approaches, embedding-based approaches, and deep-learning based approaches.

IC model [12], [15], [36], [37] is one of the most popular diffusion models which assumed independent diffusion probability through each link. Extensions of IC model further considered time delay information by incorporating a predefined time-decay weighting function, such as continuous time IC [38], CONNIE [19], NetInf [20] and Netrate [21]. Infopath [22] was proposed to infer dynamic diffusion probabilities based on information diffusion data and study the temporal evolution of information pathways. MMRate [39] inferred multi-aspect transmission rates by incorporating aspect-level user interactions and various diffusion patterns. All above methods learned the probabilities from cascade sequences. Once a model is trained, it can be used for microscopic evaluation tasks by simulating the generative process using Monte Carlo simulation.

Embedding-based approaches encoded each user into a parameterized real-valued vector and trained the parameters by maximizing an objective function. Embedded

IC [23] followed the pairwise independence assumption in IC model and modeled the diffusion probability between two users by a function of their user embeddings. Other embedding-based diffusion models [24], [25] made even stronger assumptions that infected users are determined only by the source user and the content of information item. As shown in previous work [26], such models with strong assumptions oversimplify the reality and generally show poor performance on real application tasks.

Existing deep-learning based approaches [27], [28] focused on the retweeting and sharing behaviors in a social network where "who infected whom" information is explicitly labeled in retweet chain. The next infected user candidates are also restricted to the neighboring users when the diffusion graph is known. However, the diffusion graph is usually unknown for most diffusion processes [11], [19]. For example, during the contamination of a virus, by whom a patient gets infected is unobserved. Existing deep-learning based methods considered a much simpler problem and cannot be generalized to a general setting where the diffusion graph is unobserved. To the best of our knowledge, our work is the first attempt to employ deep learning techniques for general microscopic cascade study where the diffusion graph is unknown.

2.3 Network Representation Learning

Researchers have explored many algorithms to represent nodes in a network by real-valued vectors. By projecting topology structure into vectors, we can apply machine learning techniques for many network applications, *e.g.* classification. Most network representation learning works focus on task unspecific learning where the downstream task is unknown. Early stage works [40] use eigenvector computation to learn node embeddings. With the success of neural networks, people also employ simple neural networks for representation learning [41], [42]. For task specific learning, a certain task such as classification [43] and recommendation [44] is specified and the network embeddings serve as the bottom layer of their model as what we will do in this paper. In terms of diffusion identification task, Embedded IC [23] is proposed and will be used as our baseline method.

3 Data Observation

In this section, we will conduct data observation on real-world datasets and investigate the intrinsic relationships between activated users in a diffusion sequence. In specific, we will try to figure out whether consecutively activated users are more likely to be relevant and thus appear in more diffusion sequences together. We will first introduce the datasets.

3.1 Datasets

We collect four real-world cascade datasets that cover a variety of applications for evaluation. A cascade is an item or some kind of information that spreads through a set of users. Each cascade consists of a list of $(user, timestamp)$ pairs where each pair indicates the fact that the user gets infected at the timestamp.

TABLE 1
Statistics of Datasets.

Dataset	# Users	# Links	# Cascades	Avg. Length
Lastfm	982	506,582	23,802	7.66
Irvine	540	62,605	471	13.63
Memetracker	498	158,194	8,304	8.43
Twitter	19,546	18,687,423	6,158	36.74

Lastfm is a music streaming website. We collect the dataset from [45]. The dataset contains the full history of nearly $1,000$ users and the songs they listened to over one year. We treat each song as an item spreading through users and remove the users who listen to no more than 5 songs.

Irvine is an online community for students at University of California, Irvine collected from [46]. Students can participate in and write posts on different forums. We regard each forum as an information item and remove the users who participate in no more than 5 forums.

Memetracker [2] collects a million of news stories and blog posts and track the most frequent quotes and phrases, *i.e.* memes, for studying the migration of memes across a group of people. Each meme is considered to be an information item and each URL of websites or blogs is regarded as a user. Following the settings of previous works [23], we filter the URLs to only keep the most active ones to alleviate the effect of noise.

Twitter dataset [47] concerns tweets containing URLs posted on Twitter during October 2010. The complete tweeting history of each URL is collected. We consider each distinct URL as a spreading item over Twitter users. We filter out the users with no more than 5 tweets. Note that the scale of Twitter dataset is competitive and even larger than the datasets used in previous neural-based cascade modeling algorithms [23], [28].

Note that all the above datasets have no explicit evidence about by whom a user gets infected. Though we have the following relationship in Twitter dataset, we still cannot trace the source of by whom a user is encouraged to tweet a specific URL unless the user directly retweets.

We list the statistics of datasets in Table 1. Since we have no interaction graph information between users, we assume that there exists a link between two users if they appear in the same cascade sequence. Each virtual "link" will be assigned a parameterized probability in traditional IC model and thus the space complexity of traditional methods is relatively high especially for large datasets. We also calculate the average cascade length of each dataset in the last column.

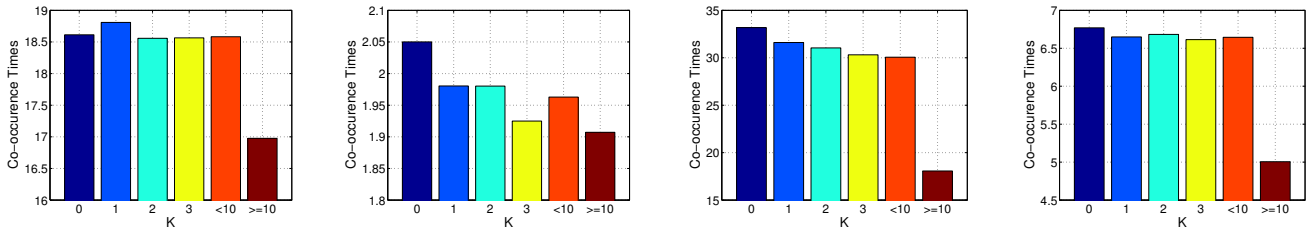
3.2 Statistical Analysis

Now we will try to reveal the correlation patterns between users by statistical results. By intuition, two consecutively infected users in a cascade sequence are more likely to have connections, *e.g.* one infects another, and thus participate in many other diffusion sequences together.

To demonstrate this statement, we consider the following statistics: given the fact that user u_i and u_j are infected

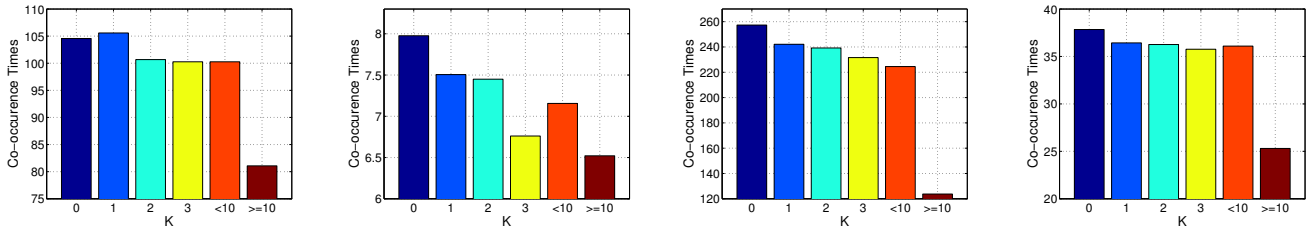
2. http://www.memetracker.org

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2019.2939796, IEEE Transactions on Knowledge and Data Engineering



(a) Lastfm (b) Irvine (c) Memetracker (d) Twitter

Fig. 2. Statistical results of the expectation of co-occurrence times of two random users given the fact that they are both infected by a cascade with K users infected between them. Here < 10 and ≥ 10 are average co-occurrence times for $K < 10$ and $K \geq 10$.



(a) Lastfm (b) Irvine (c) Memetracker (d) Twitter

Fig. 3. Statistical results of the expectation of co-occurrence times of two random users given the fact that they are both infected by a cascade with K users infected between them and they are the **top 5%** user pairs in terms of co-occurrence times satisfying the previous condition.

in a cascade sequence with K users infected between them in this sequence, what will be the expectation of the number of cascade sequences that user u_i and u_j both participate in? Here $K = 0$ indicates that user u_i and u_j are consecutively activated. If the intuition is true, then the expectation should decrease as K increases.

Fig. 2 presents the statistical results of all four datasets. Here we list the results for $K = 0, 1, 2, 3$ and the average for $K < 10$ and $K \geq 10$. The statistics show that the expectations of co-occurrence times for $K < 10$ are consistently larger than those for $K \geq 10$. Note that the gap is not very large for some datasets due to the long-tail effect. Therefore, we further present the results only for the top 5% user pairs in terms of co-occurrence times for each K in Fig. 3. We can see the differences more clearly in this setting.

These statistical results demonstrate that consecutively infected users in a cascade sequence are more likely to be relevant. By saying two users are "relevant", there could be a direct diffusion path between them or they are both likely to be infected by a third one. Also, we find that not only the most recently infected user will be relevant to the next infected one: As shown in Fig. 2 and 3, all recent infected users ($K = 0, 1, 2, 3$) could be relevant with minor differences (more relevant for smaller K). We will build our model based on these findings in next section.

4 METHOD

In this section, we will start by formalizing the problem and introducing the notations. Then we propose two heuristic assumptions according to the data observations as our basis and design a Neural Diffusion Model (NDM) using deep learning techniques. Finally, we will introduce the overall optimization function and other details of our model.

4.1 Problem Formalization

A cascade dataset records the information that an item spreads to whom and when during its diffusion. For example, the item could be a product and the cascade records who bought the product at what moment. However, in most cases, there exists no explicit interaction graph between the users [23], [37]. Therefore, we have no explicit information about how a user was infected by other users.

Formally, given user set $\mathcal{U}$ and observed cascade sequence set $\mathcal{C}$, each cascade $c_i \in \mathcal{C}$ consists a list of users $\{u_0^i, u_1^i \dots u_{|c_i|-1}^i\}$ ranked by their infection time, where $|c_i|$ is the length of sequence c_i and $u_j^i \in \mathcal{U}$ is the j-th user in the sequence c_i. Note that we only consider the order of users getting infected and ignore the exact timestamps of infections in this paper as previous works did [23], [28], [29].

In this paper, our goal is to learn a cascade model which can predict the next infected user u_{j+1} given a partially observed cascade sequence $\{u_0, u_1 \dots u_j\}$. The learned model is able to predict the entire infected user sequence based on the first few observed infected users and thus be used for microscopic evaluation tasks illustrated in Figure 1. In our model, we add a virtual user called "Terminate" to the user set $\mathcal{U}$. At training phase, we append "Terminate" to the end of each cascade sequence and allow the model to predict next infected user as "Terminate" to indicate that no more users will be infected in this cascade.

Further, we represent each user by a parameterized real-valued vector to project users into vector space. The real-valued vectors are also called embeddings. We denote the embedding of user u as $emb(u) \in \mathbb{R}^d$ where d is the dimension of embeddings. In our model, a larger inner product between the embeddings of two users indicates a stronger correlation between the users. The embedding layer is used as the bottom layer of our model by projecting

a user into corresponding vector as shown in Figure 4.

4.2 Model Assumptions

In traditional Independent Cascade (IC) model [12] settings, all previously infected users can activate a new user independently and equally regardless of their orders of getting infected. Many extensions of IC model further considered time delay information such as continuous time IC (CTIC) [38] and Netrate [21]. However, none of these models tried to find out which users are actually active and more likely to activate other users at the moment. To address this issue, we propose the following assumption.

Assumption 1. Given a recently infected user u, users that are strongly correlated to user u including user u itself are more likely to be active.

This assumption is intuitive and straight-forward. As a newly activated user, u should be active and may infect other users. The users strongly correlated to user u are probably the reason why user u gets activated recently and thus more likely to be active than other users at the moment. We further propose the concept of "active user embedding" to characterize all such active users.

Definition 1. For each recently infected user u, we aim to learn an active user embedding $act(u) \in \mathbb{R}^d$ which represents the embedding of all active users related to user u, and can be used for predicting the next infected user in next step.

The active user embedding $act(u_j)$ characterizes the potential active users related to the fact that user u_j gets infected. From the data observations, we can see that all recently infected users could be relevant to the next infected one. Therefore, the active user embeddings of all recently infected users should contribute to the prediction of next infected user, which leads to the following assumption.

Assumption 2. All recently infected users should contribute to the prediction of next infected user and be processed differently according to the order of getting infected.

Compared with the strong assumptions made by IC-based and embedding-based method introduced in related works, our heuristic assumptions allow our model to be more flexible and better fit cascade data. Now we will introduce how to build our model based on these two assumptions, *i.e.* extracting active users and unifying these embeddings for prediction.

4.3 Extracting Active Users with Attention Mechanism

For the purpose of computing active user embeddings, we propose to use attention mechanism [48], [49] to extract the most likely active users by giving them more weights than other users. As shown in Figure 4, the active embedding of user u_j is computed as a weighted sum of previously infected users:

$$act(u_j) = \sum_{k=0}^{j} w_{jk} emb(u_k), \tag{1}$$

where the weight of u_k is

$$w_{jk} = \frac{\exp(emb(u_j) emb(u_k)^T)}{\sum_{m=0}^{j} \exp(emb(u_j) emb(u_m)^T)}. \tag{2}$$

Note that $w_{jk} \in (0, 1)$ for every k and $\sum_{m=0}^{j} w_{jm} = 1$. w_{jk} is the normalized inner product between the embeddings of u_j and u_k which indicates the strength of correlation between them.

From the definition of active user embedding $act(u_j)$ in Eq. 1, we can see that the user embeddings $emb(u_k)$ which have a larger inner product with $emb(u_j)$ will be allocated a larger weight w_{jk}. This formula naturally follows our assumption that users strongly correlated to user u including user u itself should be paid more attention.

To fully utilize the advantages of a neural model, we further employ the multi-head attention [49] to improve the expressibility. Multi-head attention projects the user embeddings into multiple subspaces with different linear projections. Then multi-head attention performs attention mechanism on each subspace independently. Finally, multi-head attention concatenates the attention embeddings in all subspaces and feeds the result into a linear projection again.

Formally, in a multi-head attention with h heads, the embedding of i-th head is computed as

$$head_i = \sum_{k=0}^{j} w_{jk}^i emb(u_k) W_i^V, \tag{3}$$

where

$$w_{jk}^i = \frac{\exp(emb(u_j) W_i^Q (emb(u_k) W_i^K)^T)}{\sum_{m=0}^{j} \exp(emb(u_j) W_i^Q (emb(u_m) W_i^K)^T)}, \tag{4}$$

$W_i^V, W_i^Q, W_i^K \in \mathbb{R}^{d \times d}$ are head-specific linear projection matrices. In particular, W_i^Q and W_i^K can be seen to project user embeddings into *receiver* space and *sender* space respectively for asymmetric modeling.

Then we have the active user embedding $act(u_j)$

$$act(u_j) = [head_1, head_2 \dots head_h] W^O, \tag{5}$$

where [] indicates concatenation operation and $W^O \in \mathbb{R}^{hd \times d}$ projects the concatenated results into d-dimensional vector space.

Multi-head attention allows the model to "divide and conquer" information from different perspectives (*i.e.* subspaces) independently and thus is more powerful than the traditional attention mechanism.

4.4 Unifying Active User Embeddings for Prediction with Convolutional Network

Different from previous works [21], [50] which directly give a time-decay weight that assumes larger weights for the most recently infected users, we propose to use a parameterized neural network to handle the active user embeddings at different positions. Compared with a predefined exponential-decay weighting function [21], a parameterized neural network can be learned automatically to fit the real-world dataset and capture the intrinsic relationship between active user embedding at each position and next infected user prediction. In this paper, we consider Convolutional Neural Network (CNN) to meet this purpose.

CNN has been widely used in image recognition [51], recommender systems [52] and natural language processing [53]. CNN is a *shift-invariant* neural network and allows

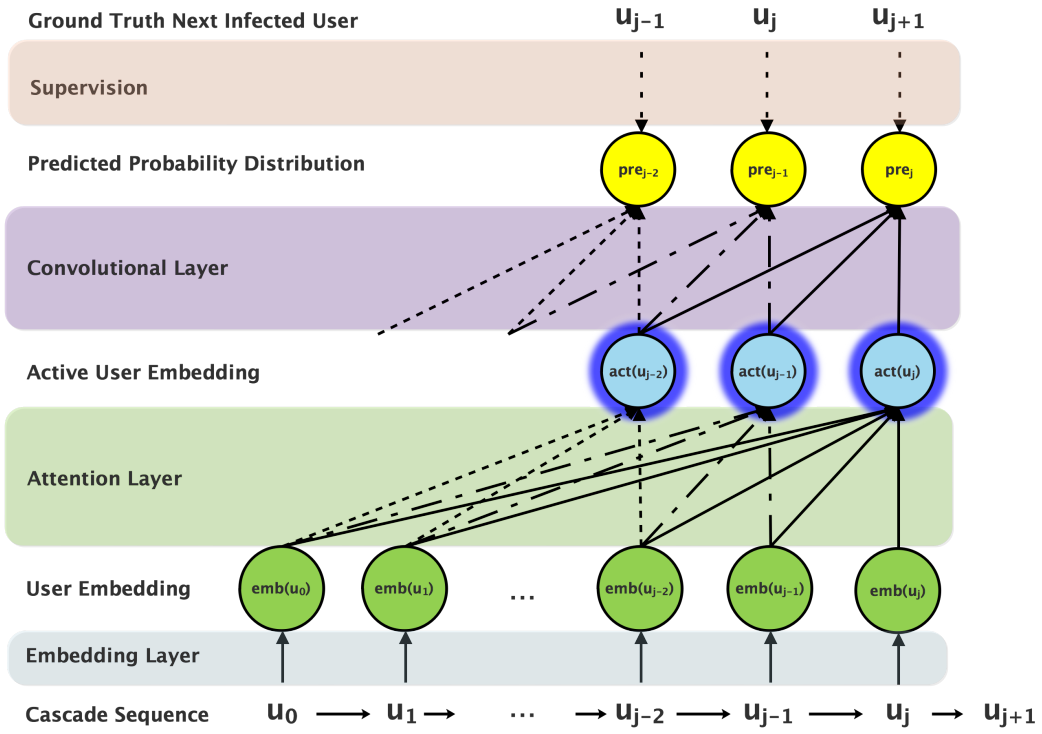


Fig. 4. An overview of our Neural Diffusion Model (NDM). NDM sequentially predicts the next infected user based on the active embeddings (the blue nodes) of recent activated users and the active embeddings is computed by an attention layer over the user embeddings (the green nodes) of all previous infected users.

us to assign position-specific linear projections to the embeddings.

Figure 4 illustrates an example where the window size of our convolutional layer $win = 3$. The convolutional layer first converts each active user embedding $act(u_{j-n})$ into a $|\mathcal{U}|$-dimensional vector by a position-specific linear projection matrix $W_n^C \in \mathbb{R}^{d \times |\mathcal{U}|}$ for $n = 0, 1 \dots win - 1$. Then the convolutional layer sums up the projected vectors and normalizes the summation by *softmax* function.

Formally, given partially observed cascade sequence $(u_0, u_1 \dots u_j)$, the predicted probability distribution $pre_j \in \mathbb{R}^{|\mathcal{U}|}$ is

$$pre_j = \text{softmax}(\sum_{n=0}^{win-1} act(u_{j-n}) W_n^C), \tag{6}$$

where $\text{softmax}(x)[i] = \frac{\exp(x[i])}{\sum_p \exp(x[p])}$ and $x[i]$ denotes the i-th entry of a vector x. Each entry of pre_j represents the probability that the corresponding user gets infected at next step.

Since the initial user u_0 plays an important role in the whole diffusion process, we further take u_0 into consideration:

$$pre_j = \text{softmax}(\sum_{n=0}^{win-1} act(u_{j-n}) W_n^C + act(u_0) W_{init}^C \cdot F_{init}), \tag{7}$$

where $W_{init}^C \in \mathbb{R}^{d \times |\mathcal{U}|}$ is the projection matrix for initial user u_0 and $F_{init} \in \{0, 1\}$ is a hyperparameter which controls whether incorporate initial user for prediction or not.

4.5 Overall Architecture, Model Details and Learning Algorithms

We naturally maximize the log-likelihood of all observed cascade sequences to build the overall optimization function.

$$\mathcal{L}(\Theta) = \sum_{c_i \in \mathcal{C}} \sum_{j=0}^{|c_i|-2} \log pre_j^i[u_{j+1}^i], \tag{8}$$

where $pre_j^i[u_{j+1}^i]$ is the predicted probability of ground truth next infected user u_{j+1}^i at position j in cascade c_i, and Θ is the set of all parameters need to be learned, including user embeddings $emb(u) \in \mathbb{R}^d$ for each $u \in \mathcal{U}$, projection matrices in multi-head attention $W_n^V, W_n^Q, W_n^K \in \mathbb{R}^{d \times d}$ for $n = 1, 2 \dots h$, $W^O \in \mathbb{R}^{hd \times d}$ and projection matrices in convolutional layer $W_{init}^C, W_n^C \in \mathbb{R}^{d \times |\mathcal{U}|}$ for $n = 0, 1 \dots \text{win} - 1$. Note that our model is general and can also be adapted for the case where "who infected whom" info is provided by converting a training cascade into a set of "true" infection sequences where each user is infected by its precedent user.

Implementation Details. We implement our model using PyTorch [3] and optimize the parameters by gradient descent with Adam optimizer [54]. We further employ layer normalization [55] and residual connection [56] operation to active user embedding to avoid gradient explosion or vanishment problem that may occur in deep neural networks. In other words, the active user embedding $act(u)$ is replaced by $LayerNorm(emb(u) + act(u))$ instead where

3. http://pytorch.org

the $LayerNorm(\cdot)$ function encourages the output to have zero mean and unit variance. We also use dropout [57] to the attention mechanism to prevent our model from overfitting and the dropout rate is set to 0.1. Since the same user will not be infected twice, we mask the users that are already infected in the Eq. 7 so that they won't be predicted. We release our source code at github [4] and all the details are listed. Hyperparameter settings will be introduced in next section.

Complexity. The space complexity of our model is $O(d|\mathcal{U}|)$ where d is the embedding dimension which is much less than the size of user set. Note that the space complexity of training traditional IC model will go up to $O(|\mathcal{U}|^2)$ because we need to assign an infection probability between each pair of potential linked users. Therefore, the space complexity of our neural model is less than that of traditional IC methods.

The computation of a single active embedding takes $O(|c_i|d^2)$ time where c_i is the length of corresponding cascade and the next infected user prediction in Eq. 7 step takes $O(d|\mathcal{U}|)$ time. Hence the time complexity of training a single cascade is $O(\sum_{c_i \in \mathcal{C}}(|c_i|^2 d^2 + |c_i|d|\mathcal{U}|))$ which is competitive with previous neural-based models such as embedded IC model [23]. But as we will show in the experiments, our model converges much faster than embedded IC model and is capable of handling large-scale dataset.

5 Experiments

We conduct experiments on microscopic diffusion identification [5] task as previous works did [23] to evaluate the performance of our model and various baseline methods. We will first introduce the baseline methods, evaluation metrics and hyperparameter settings. Then we will present the experimental results and give further analysis about the evaluation.

5.1 Baselines

We consider a number of state-of-the-art baselines to demonstrate the effectiveness of our algorithm. Most of the baseline methods will learn a transition probability matrix $M \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{U}|}$ from cascade sequences where each entry M_{ij} represents the probability that user u_j gets infected by u_i when u_i is activated.

Netrate [21] considers the time-varying dynamics of diffusion probability through each link and defines three transmission probability models, *i.e.* exponential, power-law and Rayleigh, which encourage the diffusion probability to decrease as the time interval increases. In our experiments, we only report the results of exponential model since the other two models give similar results.

Infopath [22] also targets on inferring dynamic diffusion probabilities based on information diffusion data. Infopath employs stochastic gradient to estimate the temporal dynamics and studies the temporal evolution of information pathways.

Embedded IC [23] explores representation learning technique and models the diffusion probability between two users by a function of their user embeddings instead of a static value. Embedded IC model is trained by stochastic gradient descent method.

LSTM is a widely used neural network framework [58] for sequential data modeling and has been used for cascade modeling recently. Previous works employ LSTM for some simpler tasks such as popularity prediction [26] and cascade identification with known diffusion graph [27], [28]. Since none of these works are directly comparable to ours, we adopt LSTM network for comparison by adding a softmax classifier to the hidden state of LSTM at each step for next infected user prediction.

5.2 Hyperparameter Settings for Neural Models

Though the parameter space of neural network based methods is much less than that of traditional IC models, we have to set several hyperparameters to train neural models. To tune the hyperparameters, we randomly select 10% of training cascade sequences as validation set. Note that all training cascade sequences including the validation set will be used to train the final model for testing.

For Embedded IC model, the dimension of user embeddings is selected from $\{25, 50, 100\}$ as the original paper did [23]. For LSTM model, the dimensions of user embeddings and hidden states are set to the best choice from $\{16, 32, 64, 128\}$. For our model NDM, the number of heads used in multi-head attention is set to $h = 8$, the window size of convolutional network is set to $win = 3$ and the dimension of user embeddings is set to $d = 64$. Note that we use the same set of (h, win, d) for all the datasets. The flag F_{init} in Eq. 7 which determines whether the initial user is used for prediction is set to $F_{init} = 1$ for Twitter dataset and $F_{init} = 0$ for the other three datasets. We will show the robustness of our model in parameter sensitivity subsection.

Note that neural models, *i.e.* Embedded IC, LSTM and NDM, are based on matrix multiplication operations and thus naturally benefit from the GPU acceleration. Therefore, we train these three methods on a GPU device (GeForce GTX TITAN X) instead of a CPU device (Intel Xeon E5-2620 @ 2.0GHz).

5.3 Microscopic Diffusion Identification

To compare the ability of cascade modeling, we evaluate our model and all baseline methods on the microscopic diffusion identification task. We follow the experimental settings in Embedded IC [23] for a fair comparison. We randomly select 90% cascade sequences as training set and the rest as test set. For each cascade sequence $c = (u_0, u_1, u_2 \dots)$ in the test set, only the initial user u_0 is given and all successively infected users $G^c = \{u_1, u_2 \dots u_{|G^c|}\}$ need to be predicted. Note that we completely ignore exact timestamp information in this work and the time order among sequences is omitted for simplification. We will explore a more reasonable setting as future work by taking timestamp information into consideration.

All baseline methods and our model are required to identify a set of users and the results will be compared with ground truth infected user set G. For baseline methods that

4. https://github.com/albertyang33/NeuralDiffusionModel

5. Previous work called this task as "diffusion prediction" while we name it as "diffusion identification" as the time ordering is not fully considered in this setting.

TABLE 2
Experimental results on microscopic diffusion identification.

Metric	Dataset	Method					Improvement
		Netrate	Infopath	Embedded IC	LSTM	NDM	
Macro-F1	Lastfm	0.017	0.030	0.020	0.026	**0.056**	+87%
	Memetracker	0.068	0.110	0.060	0.102	**0.139**	+26%
	Irvine	0.032	0.052	0.054	0.041	**0.076**	+41%
	Twitter	-	0.044	-	0.103	**0.139**	+35%
Micro-F1	Lastfm	0.007	0.046	0.085	0.072	**0.095**	+12%
	Memetracker	0.050	0.142	0.115	0.137	**0.171**	+20%
	Irvine	0.029	0.073	0.102	0.080	**0.108**	+6%
	Twitter	-	0.010	-	0.052	**0.087**	+67%

TABLE 3
Experimental results on microscopic diffusion identification at early stage where only the first five infected users are predicted in each cascade.

Metric	Dataset	Method					Improvement
		Netrate	Infopath	Embedded IC	LSTM	NDM	
Macro-F1	Lastfm	0.018	0.028	0.010	0.018	**0.048**	+71%
	Memetracker	0.071	0.094	0.042	0.091	**0.122**	+30%
	Irvine	0.031	0.030	0.027	0.018	**0.064**	+106%
	Twitter	-	0.040	-	0.097	**0.123**	+27%
Micro-F1	Lastfm	0.016	0.035	0.013	0.019	**0.045**	+29%
	Memetracker	0.076	0.106	0.040	0.094	**0.126**	+19%
	Irvine	0.028	0.030	0.029	0.020	**0.065**	+117%
	Twitter	-	0.050	-	0.093	**0.118**	+27%

ground in IC model, *i.e.* Netrate, Infopath and Embedded IC, we will simulate the infection process according to the learned pairwise diffusion probability and their corresponding generation process. For LSTM and our model, we can sequentially sample a user according to the probability distribution of softmax classifier at each step.

Note that the ground truth infected user set could also be partially observed because the datasets are crawled within a short time window. Therefore, for each test sequence c with $|G^c|$ ground truth infected users, all the algorithms are only required to identify the first $|G^c|$ infected users in a single simulation. Also note that the simulation may terminate and stop infecting new users before activating $|G^c|$ users.

We conduct 1000 times Monte Carlo simulations for each test cascade sequence c for all algorithms and compute the infection probability P_u^c of each user $u \in \mathcal{U}$. We evaluate the identification results using two classic evaluation metrics: Macro-F1 and Micro-F1.

Macro-F1. Macro-averaged F1 first computes the precision pre_c, recall rec_c and F1 score f_c locally for each test cascade sequence c in the test set $\mathcal{C}_{test}$. Then macro-averaged F-measure takes the average over all test cascade sequences:

$$pre_c = \frac{\sum_{u \in G^c} P_u^c}{\sum_{u \in \mathcal{U}} P_u^c}, rec_c = \frac{\sum_{u \in G^c} P_u^c}{|G^c|}, f_c = \frac{2pre_c \cdot rec_c}{pre_c + rec_c},$$

$$Macro - F1 = \frac{\sum_{c \in |\mathcal{C}_{test}|} f_c}{|\mathcal{C}_{test}|}.$$

Micro-F1. Micro-averaged F1 computes precision pre, recall rec globally by averaging over all identifications and serves as a complementary view by giving larger weights to longer cascades:

$$pre = \frac{\sum_{c \in |\mathcal{C}_{test}|} \sum_{u \in G^c} P_u^c}{\sum_{c \in |\mathcal{C}_{test}|} \sum_{u \in \mathcal{U}} P_u^c}, rec = \frac{\sum_{c \in |\mathcal{C}_{test}|} \sum_{u \in G^c} P_u^c}{\sum_{c \in |\mathcal{C}_{test}|} |G^c|},$$

$$Micro - F1 = \frac{2pre \cdot rec}{pre + rec}.$$

To further evaluate the performance of cascade identification at early stage, we conduct additional experiments by only predicting the first five infected users in each test cascade. We present the experimental results in Table 2 and 3. Here "-" indicates that the algorithm fails to converge in 72 hours. The last column represents the relative improvement of NDM against the best performing baseline method. We have the following observations:

(1) NDM consistently and significantly outperforms all the baseline methods. As shown in Table 2, the relative improvement against the best performing baseline is at least 26% in terms of Macro-F1 score. The improvement on Micro-F1 score further demonstrates the effectiveness and robustness of our proposed model. The results also indicate that well-designed neural network models are able to surpass traditional cascade methods on cascade modeling.

(2) NDM has even more significant improvements on cascade identification task *at early stage*. As shown in Table 3, NDM outperforms all baselines by a large margin on both Macro and Micro F1 scores. Note that it's very important to identify the first wave of infected users accurately for real-world applications because a wrong identification will lead to error propagation in following stages. A precise identification of infected users at early stage enables us to better control the spread of information items through users. For example, we can prevent the spread of a rumor by warning the most vulnerable users in advance and promote the spread of a product by paying the most potential customers more attention. This experiment demonstrates that NDM has the ability to be used for real-world applications.

(3) NDM is capable of handling large-scale cascade datasets. Previous neural-based method, Embedded IC, fails to converge in 72 hours on Twitter dataset with around 20 thousand users and 19 million of potential links. In contrast,

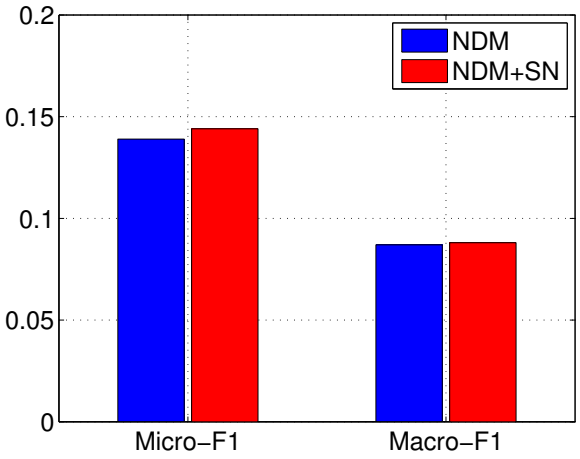


Fig. 5. Comparisons between NDM and NDM+SN on diffusion identification.

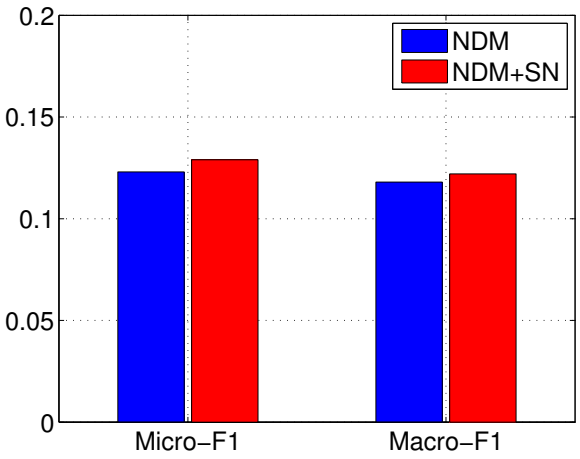


Fig. 6. Comparisons between NDM and NDM+SN on diffusion identification at early stage where only the first five infected users are predicted.

NDM converges in 6 hours on this dataset with the same GPU device, which is at least 10 times faster than Embedded IC. This observation demonstrates the scalability of NDM.

5.4 Benefits from Social Network

Sometimes the underlying social network of users is available, *e.g.* the Twitter dataset used in our experiments. In the Twitter dataset, a network of Twitter followers is observed though the information diffusion is not necessarily passed through the edges of the social network. We hope that the diffusion identification process could benefit from the observed social network structure. We apply a simple modification on our NDM model to take advantage of the social network. Now we will introduce the modification in detail.

Firstly, we embed the topological social network structure into real-valued user features by DeepWalk [41], a widely used network representation learning algorithm. The dimension of network embeddings learned by DeepWalk is set to 32 which is half of the dimension $d = 64$ which

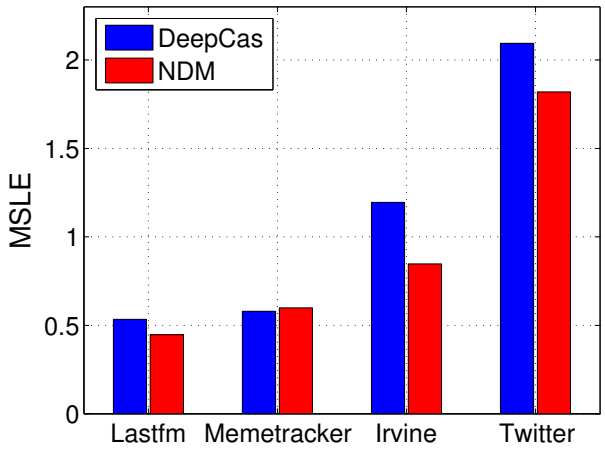


Fig. 7. Experimental results on cascade size identification. MSLE is the lower the better.

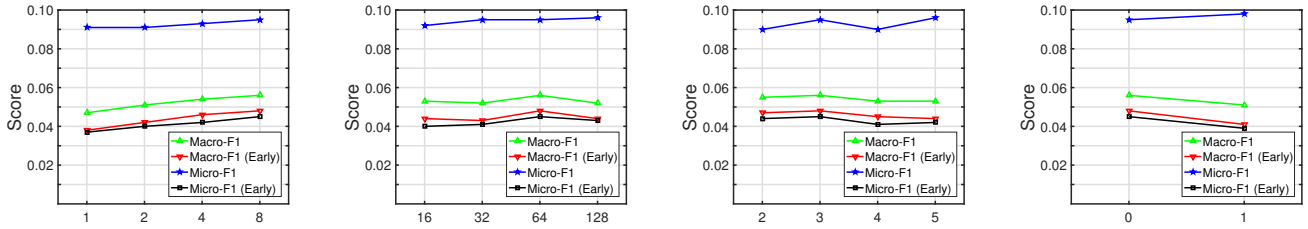
is the representation size of our model. Secondly, we use the learned network embeddings to initialize the first 32 dimensions of the user representations of our model and fix them during the training process without changing any other modules. In other words, a 64-dimensional user representation is made up of a 32-dimensional fixed network embedding learned by DeepWalk from social network structure and another 32-dimensional randomly initialized trainable embedding. We name the modified model with Social Network considered as NDM+SN for short. This is a simple but useful implementation and we will explore a more sophisticated model to take the social network into modeling directly in future work. Fig. 5 and 6 show the comparison between NDM and NDM+SN.

Experimental results show that NDM+SN is able to improve the performance on diffusion identification task slightly with the help of incorporating social network structure as prior knowledge. The relative improvement of Micro-F1 is around 4%. The results demonstrate that our neural model is very flexible and can be easily extended to take advantage of external features.

5.5 Macroscopic Cascade Size Identification

Though our proposed model aims at microscopic cascade modeling, we will explore the ability of macroscopic cascade size identification of our model in this subsection. For cascade size identification, all models are asked to predict the final size of a cascade given the first 5 infected users. We employ the same datasets and data splits in microscopic identification.

Recall that we append "Terminate" as a virtual user to the end of each cascade sequence and allow the model to predict next infected user as "Terminate" to indicate that no more users will be infected in this cascade. Similar to the experimental setting in section 5.3, we run Monte Carlo simulations for each test cascade sequence to estimate the eventual size of a cascade. We use Mean Square Log-transformed Error (MSLE), which is employed in previous work on cascade size prediction [26], [35], as evaluation



(a) # heads $h \in \{1, 2, 4, 8\}$ (b) # dimensions $d \in \{16, 32, 64, 128\}$ (c) # window size $win \in \{2, 3, 4, 5\}$ (d) Flag $F_{init} \in \{0, 1\}$

Fig. 8. Parameter sensitivity of hyperparameters in Lastfm Dataset. Macro-F1 (Early) and Micro-F1 (Early) correspond to the identification performance at early stage.

metric. In specific, MSLE$= \frac{1}{|C|}\sum_{i=1}^{|C|}(\log(|c_i|) - \log(pred_i))^2$ where $pred_i$ is the predicted size of cascade c_i.

We consider DeepCas [26], the state-of-the-art cascade size prediction algorithm, as our baseline method. The experimental results are shown in Figure 7.

Experimental results show that NDM gives comparative and even better performance against DeepCas [26] on cascade size identification task, though NDM is not directly optimized for macroscopic level identification. Compared with DeepCas which treats cascade size prediction as a regression problem, NDM actually utilizes more information in the training data, *i.e.* the detailed infected users and the ordering of their infections. To conclude, our proposed microscopic diffusion model can also be used for macroscopic cascade size identification and achieve good performance by utilizing more information. These observations would shed light on future direction of macroscopic level prediction and encourage a unified cascade model for both microscopic and macroscopic identification.

5.6 Parameter Sensitivity

In this subsection, we will take Lastfm dataset as an illustrative example to present how hyperparameter settings affect the performance of our model. We use the best set of hyperparameter settings as our basis, *i.e.* number of heads $h = 8$, window size of convolutional network $win = 3$, dimension of user embeddings $d = 64$ and flag of using initial user for prediction $F_{init} = 0$. Then we vary each hyperparameter while keeping others fixed. Figure 8 shows the performance on diffusion identification under different hyperparameter settings.

We can see that the performance of NDM is stable when we vary the hyperparameters within a reasonable range. NDM does not encounter serious overfitting problem when we double the dimension of embeddings d to 128. This experiment demonstrates the robustness of our model.

5.7 Interpretability

Admittedly, the interpretability is usually a weak point of neural network models. Compared with feature engineering methods, neural-based models encode a user into a real-valued vector space and there is no explicit meaning of each dimension of user embeddings. In our proposed model, each user embedding is projected into 16 subspaces by an 8-head attention mechanism. Intuitively, the user embedding in each subspace represents a specific role of the user. But it is quite hard for us to link the 16 embeddings to interpretable hand-crafted features. We will consider the alignment between user embeddings and interpretable features based on a joint model in future work.

TABLE 4
The scale of learned projection matrices in convolutional layer measured by Frobenius norm $||\cdot||_F^2$.

Dataset	W_{init}^C	W_0^C	W_1^C	W_2^C
Lastfm	32.3	60.0	49.2	49.1
Memetracker	13.3	16.6	13.3	13.0
Irvine	13.9	13.9	13.7	13.7
Twitter	130.3	93.6	91.5	91.5

Fortunately, we still have some findings in the convolutional layer. Recall that $W_n^C \in \mathbb{R}^{d\times|\mathcal{U}|}$ for $n = 0, 1, 2$ are position-specific linear projection matrices in convolutional layer and W_{init}^C is the projection matrix for the initial user. All four matrices are randomly initialized before training. In a learned model, if the scale of one of these matrices is much larger than that of other ones, then the prediction vector is more likely to be dominated by the corresponding position. For example, if the scale of W_0^C is much larger than that of other ones, then we can infer that the most recent infected user contributes most to the next infected user prediction.

Following the notations in Eq. 7, we set $F_{init} = 1$ for all datasets in this experiment and compute the square of Frobenius norm of learned projection matrices as shown in Table 4. We have the follow observations:

(1) For all four datasets, the scales of W_0^C, W_1^C and W_2^C are competitive and the scale of W_0^C is always a little bit larger than that of the other two. This observation indicates that the active embeddings $act(u_j), act(u_{j-1}), act(u_{j-2})$ of all three recently infected users will contribute to the prediction of u_{j+1}. Also, the most recent infected user u_j is the most important one among the three. This finding naturally matches our intuitions and verifies *Assumption* 2 proposed in method section.

(2) The scale of W_{init}^C is the largest on Twitter dataset. This indicates that the initial user is very important in diffusion process on Twitter. This is partly because Twitter dataset contains the complete history of the spread of a URL and the initial user is actually the first one tweeting the URL. While in the other three datasets, the initial user is only the first one within the time window of crawled data.

Note that we set hyperparameter $F_{init} = 1$ only for Twitter dataset in diffusion identification task because we find that the performances are competitive or even worse on the other three datasets if we set $F_{init} = 1$.

6 Conclusion

In this paper, we propose a Neural Diffusion Model (NDM) for microscopic cascade modeling. To go beyond the limitations of traditional cascade models based on strong assumptions and oversimplified formulas, we build our model based on two heuristic assumptions and employ deep learning techniques including convolutional neural network and attention mechanism to implement the assumptions. Experimental results on diffusion identification task demonstrate the effectiveness and robustness of our proposed model. In addition, NDM greatly outperforms baseline methods on diffusion identification *at early stage*, which shows the applicability and feasibility of NDM for real-world applications.

For future works, we will consider linking neural-based model with hand-crafted features and statistics to improve the interpretability of learned models. An intelligible model is always welcome and can help us better understand the motivations and behaviors of users in a diffusion process.

The incorporation of extra information for cascade modeling is also an intriguing direction. For example, the timestamp information and the description of information items can be used for more accurate cascade modeling.

Acknowledgments

This work was supported by the 973 Program (No. 2014CB340501), the Major Project of the National Social Science Foundation of China (13&ZD190) and the National Natural Science Foundation of China (No. 61772302). This work is also part of the NExT++ project, supported by the National Research Foundation, Prime Ministers Office, Singapore under its IRC@Singapore Funding Initiative.

References

[1] M. J. Salganik, P. S. Dodds, and D. J. Watts, "Experimental study of inequality and unpredictability in an artificial cultural market," *science*, 2006.

[2] J. Cheng, L. Adamic, P. A. Dow, J. M. Kleinberg, and J. Leskovec, "Can cascades be predicted?" in *Proceedings of WWW*. ACM, 2014.

[3] L. Yu, P. Cui, F. Wang, C. Song, and S. Yang, "From micro to macro: Uncovering and predicting information cascading process with behavioral dynamics," in *ICDM*, 2015.

[4] Q. Zhao, M. A. Erdogdu, H. Y. He, A. Rajaraman, and J. Leskovec, "Seismic: A self-exciting point process model for predicting tweet popularity," in *Proceedings of SIGKDD*, 2015.

[5] P. Domingos and M. Richardson, "Mining the network value of customers," in *Proceedings of SIGKDD*. ACM, 2001.

[6] J. Leskovec, A. Singh, and J. Kleinberg, "Patterns of influence in a recommendation network," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2006, pp. 380–389.

[7] J. Leskovec, L. A. Adamic, and B. A. Huberman, "The dynamics of viral marketing," *ACM Transactions on the Web (TWEB)*, vol. 1, no. 1, p. 5, 2007.

[8] D. J. Watts and P. S. Dodds, "Influentials, networks, and public opinion formation," *Journal of consumer research*, 2007.

[9] S. Aral and D. Walker, "Identifying influential and susceptible members of social networks," *Science*, 2012.

[10] H. W. Hethcote, "The mathematics of infectious diseases," *SIAM review*, vol. 42, no. 4, pp. 599–653, 2000.

[11] J. Wallinga and P. Teunis, "Different epidemic curves for severe acute respiratory syndrome reveal similar impacts of control measures," *American Journal of epidemiology*, 2004.

[12] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of SIGKDD*. ACM, 2003, pp. 137–146.

[13] T. Lappas, E. Terzi, D. Gunopulos, and H. Mannila, "Finding effectors in social networks," in *Proceedings of SIGKDD*, 2010.

[14] P. A. Dow, L. A. Adamic, and A. Friggeri, "The anatomy of large facebook cascades." *ICWSM*, 2013.

[15] D. Gruhl, R. Guha, D. Liben-Nowell, and A. Tomkins, "Information diffusion through blogspace," in *Proceedings of WWW*, 2004.

[16] D. Liben-Nowell and J. Kleinberg, "Tracing information flow on a global scale using internet chain-letter data," *Proceedings of the national academy of sciences*, vol. 105, no. 12, pp. 4633–4638, 2008.

[17] J. Leskovec, L. Backstrom, and J. Kleinberg, "Meme-tracking and the dynamics of the news cycle," in *Proceedings of SIGKDD*, 2009.

[18] D. M. Romero, B. Meeder, and J. Kleinberg, "Differences in the mechanics of information diffusion across topics: idioms, political hashtags, and complex contagion on twitter," in *Proceedings of WWW*. ACM, 2011, pp. 695–704.

[19] S. Myers and J. Leskovec, "On the convexity of latent social network inference," in *Proceedings of NIPS*, 2010.

[20] M. Gomez Rodriguez, J. Leskovec, and A. Krause, "Inferring networks of diffusion and influence," in *Proceedings of SIGKDD*. ACM, 2010.

[21] M. G. Rodriguez, J. Leskovec, D. Balduzzi, and B. Schölkopf, "Uncovering the structure and temporal dynamics of information propagation," *Network Science*, vol. 2, no. 1, pp. 26–65, 2014.

[22] M. Gomez Rodriguez, J. Leskovec, and B. Schölkopf, "Structure and dynamics of information pathways in online media," in *Proceedings of WSDM*. ACM, 2013.

[23] S. Bourigault, S. Lamprier, and P. Gallinari, "Representation learning for information diffusion through social networks: an embedded cascade model," in *Proceedings of WSDM*. ACM, 2016.

[24] S. Bourigault, C. Lagnier, S. Lamprier, L. Denoyer, and P. Gallinari, "Learning social network embeddings for predicting information diffusion," in *Proceedings of WSDM*. ACM, 2014.

[25] S. Gao, H. Pang, P. Gallinari, J. Guo, and N. Kato, "A novel embedding method for information diffusion prediction in social network big data," *IEEE Transactions on Industrial Informatics*, 2017.

[26] C. Li, J. Ma, X. Guo, and Q. Mei, "Deepcas: An end-to-end predictor of information cascades," in *Proceedings of WWW*, 2017.

[27] W. Hu, K. K. Singh, F. Xiao, J. Han, C.-N. Chuah, and Y. J. Lee, "Who will share my image?: Predicting the content diffusion path in online social networks," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 2018, pp. 252–260.

[28] J. Wang, V. W. Zheng, Z. Liu, and K. C.-C. Chang, "Topological recurrent neural network for diffusion prediction," in *ICDM*. IEEE, 2017, pp. 475–484.

[29] Z. T. Kefato, N. Sheikh, and A. Montresor, "Di: Diffusion network inference through representation learning," 2017.

[30] P. Cui, S. Jin, L. Yu, F. Wang, W. Zhu, and S. Yang, "Cascading outbreak prediction in networks: a data-driven approach," in *Proceedings of SIGKDD*. ACM, 2013.

[31] O. Tsur and A. Rappoport, "What's in a hashtag?: content based prediction of the spread of ideas in microblogging communities," in *Proceedings of WSDM*. ACM, 2012, pp. 643–652.

[32] L. Weng, F. Menczer, and Y.-Y. Ahn, "Predicting successful memes using network and community structure." in *ICWSM*, 2014.

[33] H. Pinto, J. M. Almeida, and M. A. Gonçalves, "Using early view patterns to predict the popularity of youtube videos," in *Proceedings of WSDM*, 2013.

[34] S. Gao, J. Ma, and Z. Chen, "Modeling and predicting retweeting dynamics on microblogging platforms," in *Proceedings of WSDM*. ACM, 2015.

[35] Q. Cao, H. Shen, K. Cen, W. Ouyang, and X. Cheng, "Deephawkes: Bridging the gap between prediction and understanding of information cascades," in *Proceedings of CIKM*. ACM, 2017.

[36] J. Goldenberg, B. Libai, and E. Muller, "Talk of the network: A complex systems look at the underlying process of word-of-mouth," *Marketing letters*, 2001.

[37] K. Saito, R. Nakano, and M. Kimura, "Prediction of information diffusion probabilities for independent cascade model," in *Knowledge-based intelligent information and engineering systems*. Springer, 2008, pp. 67–75.

[38] K. Saito, M. Kimura, K. Ohara, and H. Motoda, "Learning continuous-time information diffusion model for social behavioral data analysis," in *Asian Conference on Machine Learning*. Springer, 2009, pp. 322–337.
[39] S. Wang, X. Hu, P. S. Yu, and Z. Li, "Mmrate: inferring multi-aspect diffusion networks with multi-pattern cascades," in *Proceedings of SIGKDD*. ACM, 2014, pp. 1246–1255.
[40] L. Tang and H. Liu, "Relational learning via latent social dimensions," in *Proceedings of SIGKDD*. ACM, 2009, pp. 817–826.
[41] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of SIGKDD*, 2014.
[42] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of WWW*. International World Wide Web Conferences Steering Committee, 2015, pp. 1067–1077.
[43] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.
[44] C. Yang, M. Sun, W. X. Zhao, Z. Liu, and E. Y. Chang, "A neural network approach to jointly modeling social networks and mobile trajectories," *TOIS*, vol. 35, no. 4, p. 36, 2017.
[45] Ò. Celma Herrada, "Music recommendation and discovery in the long tail," 2009.
[46] T. Opsahl and P. Panzarasa, "Clustering in weighted networks," *Social networks*, 2009.
[47] N. O. Hodas and K. Lerman, "The simple rules of social contagion," *Scientific reports*, vol. 4, p. 4343, 2014.
[48] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.
[49] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.
[50] M. G. Rodriguez, D. Balduzzi, and B. Schölkopf, "Uncovering the temporal dynamics of diffusion networks," *arXiv preprint arXiv:1105.0697*, 2011.
[51] Y. LeCun *et al.*, "Lenet-5, convolutional neural networks," *URL: http://yann. lecun. com/exdb/lenet*, 2015.
[52] A. Van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," in *Advances in neural information processing systems*, 2013, pp. 2643–2651.
[53] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of ICML*. ACM, 2008.
[54] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[55] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.
[56] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of CVPR*, 2016, pp. 770–778.
[57] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.
[58] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

Cheng Yang is an assistant professor in Beijing University of Posts and Telecommunications. He received his B.E. degree and Ph.D. degree from Tsinghua University in 2014 and 2019, respectively. His research interests include natural language processing and network representation learning. He has published several top-level papers in international journals and conferences including ACM TOIS, EMNLP, IJCAI and AAAI.

Maosong Sun is a professor of the Department of Computer Science and Technology, Tsinghua University. He got his BEng degree in 1986 and MEng degree in 1988 from Department of Computer Science and Technology, Tsinghua University, and got his Ph.D. degree in 2004 from Department of Chinese, Translation, and Linguistics, City University of Hong Kong. His research interests include natural language processing, Chinese computing, Web intelligence, and computational social sciences. He has published over 150 papers in academic journals and international conferences in the above fields. He serves as a vice president of the Chinese Information Processing Society, the council member of China Computer Federation, the director of Massive Online Education Research Center of Tsinghua University, and the Editor-in-Chief of the Journal of Chinese Information Processing.

Haoran Liu is a 4-th year undergraduate student of the Department of Electric Engineering, Tsinghua University. His research interests include network representation learning and machine learning.

Shiyi Han is a 1-st year master student in Computer Science department at Brown University. He got his B.E. degree from Beihang University in 2018. His research interests include natural language processing and machine learning.

Zhiyuan Liu is an associate professor of the Department of Computer Science and Technology, Tsinghua University. He got his BEng degree in 2006 and his Ph.D. in 2011 from the Department of Computer Science and Technology, Tsinghua University. His research interests are natural language processing and social computation. He has published over 40 papers in international journals and conferences including ACM Transactions, IJCAI, AAAI, ACL and EMNLP.

Huanbo Luan is the deputy director of NExT++ Research Center at both Tsinghua University and National University of Singapore. He received his B.S. degree in computer science from Shandong University in 2003 and Ph.D degree in computer science from Institute of Computing Technology, Chinese Academy of Sciences in 2008. His research interests include natural language processing, multimedia information retrieval, social media and big data analysis.