Sequential Information Diffusion Model with Disentangled Attention

Haoran Wang[a], Cheng Yang[a,b,*]

[a]*School of Computer Science, Beijing University of Posts and Telecommunications, China*
[b]*Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia, China*

Abstract

Information diffusion prediction is a fundamental task which forecasts how an information item will spread among users. In recent years, deep learning based methods, especially those based on recurrent neural networks (RNNs), have achieved promising results on this task by treating infected users as sequential data. However, existing methods represent all previously infected users by a single vector and could fail to encode all necessary information for future predictions due to the mode collapse problem. To address this problem, we propose to employ the idea of disentangled representation learning, which aims to extract multiple latent factors representing different aspects of the data, for modeling the information diffusion process. Specifically, we employ a sequential attention module and a disentangled attention module to better aggregate the history information and disentangle the latent factors. We further employ Gumbel-Softmax technique to encourage the disentanglement. Experimental results on three real-world datasets show that the proposed model SIDDA significantly outperforms state-of-the-art baseline methods by up to 14% in terms of $hits@N$ metric, which demonstrates the effectiveness of our method.

Keywords: Information Diffusion Prediction, Information Cascade, Deep Learning, Disentangled Representation Learning, Deep Learning

*Corresponding author
Email addresses: `wanghaoran@bupt.edu.cn` (Haoran Wang), `yangcheng@bupt.edu.cn` (Cheng Yang)

1. Introduction

The phenomenon of information diffusion, dubbed as information cascade, is ubiquitous in our daily lives, *e.g.*, the spread of breaking news or virus. Information diffusion prediction is an important and challenging task, which aims at forecasting the future properties or behaviors of an information cascade, such as the eventual size [1, 2, 3] or the next infected user [4, 5, 6]. Current studies of information diffusion prediction have been successfully adopted for many real-world scenarios including epidemiology [7], viral marketing [8], media advertising [9] and the spread of news and memes [10, 11].

During the last decade, deep learning techniques have shown their effectiveness in computer vision [12] and natural language processing areas [13]. Recently, methods [1, 14, 15, 16] based on recurrent neural networks (RNNs) also achieved promising results in information cascade modeling by treating influenced users as sequential data ranked by their infection timestamps. In RNNs, the entire information diffusion history is encoded into a real-valued vector as the hidden state. However, representing all previously infected users by a single vector could fail to encode all necessary information for future predictions due to the mode collapse [17] problem. As illustrated in Fig.1, an information item is spreading through two communities and the cascade sequence has infected 5 users (4 from community A and 1 from community B). An information cascade model need to encode these users. Learned representations of conventional methods can encode the most important factor (community A) for future predictions, but fail to remember others (community B).

To address this problem, we propose to employ the idea of disentangled representation learning, which aims to extract multiple latent factors representing different aspects of the data, for modeling the diffusion history of an information cascade. Though disentangled representation learning was originally proposed in controllable image generation [18], it has been successfully adopted for other areas such as text generation [19] as well. To the best of our knowledge, we are the first to use disentangled representation learning for information diffusion

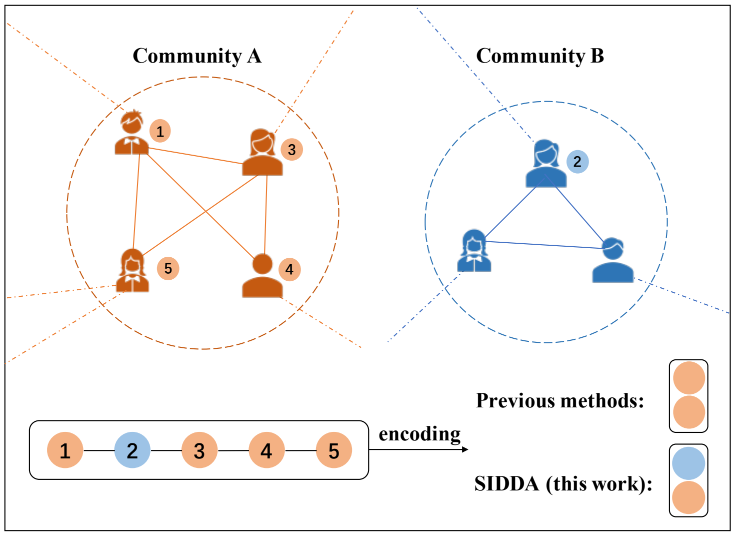

Figure 1: An illustration of mode collapse problem in information cascade modeling.

prediction.

To be more specific, we propose a novel Sequential Information Diffusion model with Disentangled Attention (SIDDA) by applying a sequential attention module and a disentangled attention module on RNNs. The former module can help better aggregate the history information and the latter one will disentangle the hidden state vector into multiple representations. We further employ Gumbel-Softmax [20] technique to encourage bigger differences between the disentangled representations. Consequently, we are able to learn multiple hidden state vectors characterizing different latent factors for future predictions at each step of cascade modeling. Different hidden state vectors can complement each other and provide more comprehensive information to predict the next infected users.

We conduct experiments on three public real-world datasets and employ the task of next infected user prediction for evaluation. Experimental results show that the proposed model SIDDA significantly outperforms state-of-the-art baseline methods by up to 14% in terms of *hits*@50 metric.

Our contributions are as follows:

- To the best of our knowledge, we are the first work to adopt the idea of disentangled representation learning for information cascade modeling.
- We propose SIDDA, a novel method which can learn disentangled representations encoding different factors for information diffusion prediction. We also use Gumbel-Softmax to enhance the effect of disentanglement.
- Experiments on three real-world datasets demonstrate that our proposed method outperforms state-of-the-art baselines significantly.

2. Related Work

2.1. Information Diffusion Prediction

There are two kinds of tasks associated with information diffusion prediction: (1) Studying the growth of cascades and (2) Predicting the next activated node in the cascade. In some recent works [2, 4, 21, 22], these tasks are classified as macroscopic and microscopic predictions.

2.1.1. Macroscopic Information Diffusion Prediction

Macroscopic information diffusion prediction, also known as popularity prediction, aims at predicting cascade sizes in the future [23] or whether an information diffusion process will become viral. Therefore, this task can be seen as a regression problem or a classification problem. Previously, related works are based on feature-based approaches [24] and generative approaches [25]. In recent years, with the success of deep learning, methods based on Recurrent Neural Networks (RNNs) are proposed, *e.g.*, DeepCas [1] and Deephawkes [26]. Some researchers also introduce Graph Neural Networks (GNNs) to model the underlying social graph and diffusion paths, *e.g.*, CoupledGNN [3] and HDGNN [27]. CasCN [2] combines Graph Convolutional Network [28] and RNN together, modeling diffusion paths as a dynamic graph.

2.1.2. Microscopic Information Diffusion Prediction

Our work focuses on microscopic information diffusion prediction, which aims to forecast the next activated node given previously infected users in an information cascade. We classify previous works into two categories: IC-based methods and deep-learning-based methods.

IC model [29] is one of the most widely used models, which assumed independent diffusion probability between nodes. Various extensions like continuous-time IC [30] and Embedded-IC [31], are proposed by incorporating timestamp information and representation learning technique into modeling. However, they have been shown to be sub-optimal by recent deep-learning-based methods.

With the development of deep learning, various kinds of neural networks have been adopted for microscopic information diffusion prediction. Generally, RNN, GNN and attention mechanism are widely used, and give promising results. Topo-LSTM [15] proposes a novel LSTM to model tree-structured cascades. CYAN-RNN [14] and Deep-Diffuse [32] take timestamps into consideration with the help of temporal point processes. DyHGCN [33] proposes a heterogeneous graph convolutional network to model the social graph and dynamic diffusion graph jointly. HDD [34] exploits meta-path in the diffusion graph and learns heterogeneous network representations using GNN. There are also some models fully based on attention mechanisms, *e.g.*, DAN [35], Hi-DAN [36] and NDM [4].

Besides, some works target on both macroscopic and microscopic information diffusion predictions. FOREST [21] makes use of reinforcement learning to incorporate the ability of macroscopic prediction. DMT-LIC [37] is a multi-task learning framework with a shared-representation layer and two different task layers for predictions.

However, existing methods represent previously infected users by a single vector and could encounter the mode collapse problem, which would fail to encode all necessary information for future predictions.

2.2. Disentangled Representation Learning

Disentangled representation learning which aims to learn representations of different latent factors hidden in the observed data [38, 18], has been successfully used in various fields. In the field of recommendation systems, a few works are proposed to learn disentangled item representations [39]. [40] performs self-supervision in the latent space, getting disentangled intentions from item sequences for product recommendations. In the field of natural language processing, SPG [19] proposes an unsupervised way to generate stylistic poetry by maximizing the mutual information between representations and generated text. GNUD [41] explores user interest disentanglement in news recommendation by making use of DisenGCN [42]. Besides, disentangled representation learning is also successfully used in the field of computer vision [43, 44, 45] and the modeling of graph-structured data [46, 42, 47].

As far as we know, we are the first to adopt the idea of disentangled representation learning for information diffusion prediction.

3. Method

In this section, we will formalize the problem of information diffusion prediction and introduce the notations. Then we will propose an RNN-based information diffusion prediction model and a novel attention mechanism to learn disentangled representations. Finally, we will incorporate the Gumbel-Softmax trick to further enhance the effectiveness of disentanglement.

3.1. Problem Definition

Given a node (or user) set V and a cascade set C, we have $V = \{v_1, v_2, ..., v_N\}$ and $C = \{c_1, c_2, ..., c_M\}$. Here N is the size of the node set, and M is the number of cascades. Each cascade $c_i \in C$ spreading among users is a sequence of nodes $\{v^i_1, v^i_2, ..., v^i_{|c_i|}\}$, ordered by their activation timestamps. Following the same setting in previous works [4, 15, 16, 21], we only keep the order of nodes and ignore the exact timestamps. A detailed modeling of timestamp information will be left for future work.

Information diffusion prediction can be formulated as: given the cascade sequence of previously activated nodes $\{v_1^i, v_2^i, ..., v_t^i\}$ in cascade c_i for $t = 1, 2, ..., |c_i| - 1$, predicting the next activated node v_{t+1}^i. In other words, our goal is to build a model that is able to learn the conditional probability function $p(v_{t+1}^i|\{v_1^i, v_2^i, ..., v_t^i\})$ for each cascade c_i where $1 \le i \le M$. We will focus on the modeling of a single cascade and omit the superscript for simplification in the rest of the paper. Notations used in this work are listed in Table 1.

Table 1: Notations

Notation	Description
V	the set of nodes, $V = \{v_1, v_2, ..., v_N\}$
N	the size of node set
C	the set of cascades, $C = \{c_1, c_2, ..., c_M\}$, $c_i = \{v_1^i, v_2^i, ..., v_{\lvert c_i \rvert}^i\}$
M	the size of cascade set
K	the number of disentangled factors
θ	the parameters of the model
D	the dimension of node representations
$\mathbf{X} \in \mathbb{R}^{M \times D}$	the user embedding, included in θ
$\mathbf{x_{v_i}} \in \mathbb{R}^D$	the embedding of node v_i
$\mathbf{P} \in \mathbb{R}^{K \times D}$	the prototype embedding, included in θ
$\mathbf{p_k} \in \mathbb{R}^D$	the embedding of prototype k, *i.e.*, the k^{th} row of $\mathbf{P}$

3.2. Model Architecture

The framework of our method is shown in Fig.2. Firstly, we encode every node into node embedding by an embedding-lookup layer. Then we employ a Gated Recurrent Unit (GRU) as the basis to model the node sequence. Given the infected node sequence, the GRU model can output a hidden state representing the sequential history at each time step. Then a sequential attention module and a disentangled attention module are applied to better aggregate the history information and disentangle the hidden state vector into multiple representations. Finally, disentangled representations will be used for predicting the next infected node.

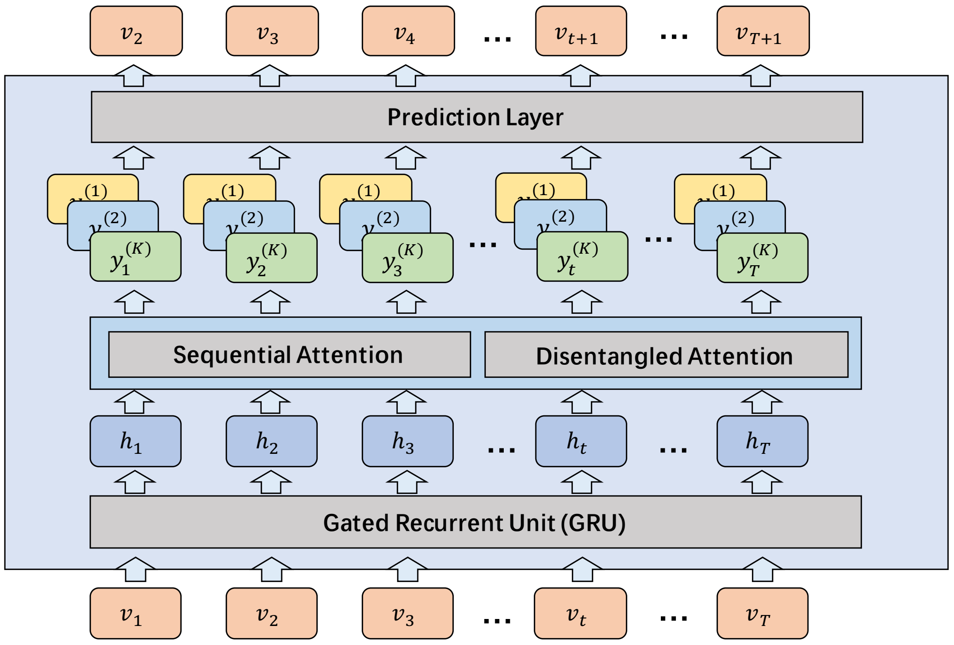

Figure 2: An illustration of our proposed model SIDDA. The model will derive K disentangled representations $\mathbf{y_t^{(k)}}(1 \leq k \leq K)$ at time t, with the help of a disentangled attention module.

3.2.1. Gated Recurrent Unit

Recurrent neural networks(RNNs) have shown great potentials in modeling sequence data. Previous works [15, 16, 21, 14, 32, 48] used RNNs as their basis to model information diffusion cascades. Firstly, we represent each node as a vector $\mathbf{x} \in \mathbb{R}^D$. By feeding the activated node embedding $\mathbf{x_t}$ into an RNN at every timestamp, we can have a hidden state $\mathbf{h_t}$ capturing the history information of all previously activated nodes as the output.

Gated Recurrent Unit (GRU) [49] can alleviate the vanishing gradient problem compared with a standard RNN. It can be considered as a variation of the LSTM [50] but is computationally cheaper. Therefore, we select GRU as our model basis. To make this work self-contained, we will briefly introduce GRU.

Compared with the vanilla RNN, GRU has three extra gates. The reset gate $\mathbf{r_t}$ decides how much of the history information to forget for computing the candidate hidden state, using current node embedding $\mathbf{x_t}$ and last hidden state $\mathbf{h_{t-1}}$. Then the candidate hidden state $\mathbf{n_t}$ will be computed by a recurrent function. The update gate $\mathbf{z_t}$ determines the balance between candidate state n_t and last hidden state $\mathbf{h_{t-1}}$ for computing the next hidden state $\mathbf{h_t}$. The formula of the update gate is the same as that of the reset gate, but its usage and the parameters are different.

$$\begin{aligned}
\mathbf{r_t} &= \sigma(\mathbf{W_{ir}x_t} + \mathbf{b_{ir}} + \mathbf{W_{hr}h_{t-1}} + \mathbf{b_{hr}}) \\
\mathbf{z_t} &= \sigma(\mathbf{W_{iz}x_t} + \mathbf{b_{iz}} + \mathbf{W_{hz}h_{t-1}} + \mathbf{b_{hz}}) \\
\mathbf{n_t} &= \tanh(\mathbf{W_{in}x_t} + \mathbf{b_{in}} + \mathbf{r_t} \odot (\mathbf{W_{hn}h_{t-1}} + \mathbf{b_{hn}})) \\
\mathbf{h_t} &= (1 - \mathbf{z_t}) \odot \mathbf{n_t} + \mathbf{z_t} \odot \mathbf{h_{t-1}}
\end{aligned} \tag{1}$$

Here $\odot$ is the Hadamard (element-wise) product between vectors. $\mathbf{W}_* \in \mathbb{R}^{D\times D}$, $\mathbf{b}_* \in \mathbb{R}^D$ are weight matrices and bias vectors to be learned, where D is the dimension of node embeddings and hidden states.

3.2.2. Sequential Attention Module

Note that every hidden state $\mathbf{h_t}$ in GRU contains information about the input node sequence with a focus around position t. To aggregate the sequential history information of all positions automatically, we propose to employ attention mechanism, which was originally proposed in neural machine translation [51], and learn to assign different attention weights to previous hidden states $\mathbf{h_i}$ ($i \le t$). Formally, the sequential attention weight α_{it} of the i-th hidden state is computed by:

$$\begin{aligned}
\alpha_{it} &= \frac{exp(\frac{1}{\sqrt{D}}e_{it})}{\sum_{j=1}^{t} exp(\frac{1}{\sqrt{D}}e_{jt})} \\
e_{it} &= d(\mathbf{h_i}, \mathbf{h_t})
\end{aligned} \tag{2}$$

where $\frac{1}{\sqrt{D}}$ is a scaling coefficient and $h_i \in \mathbb{R}^D$ is the i-th hidden state. $d(\cdot,\cdot)$ is a distance function, and we use dot product in this work.

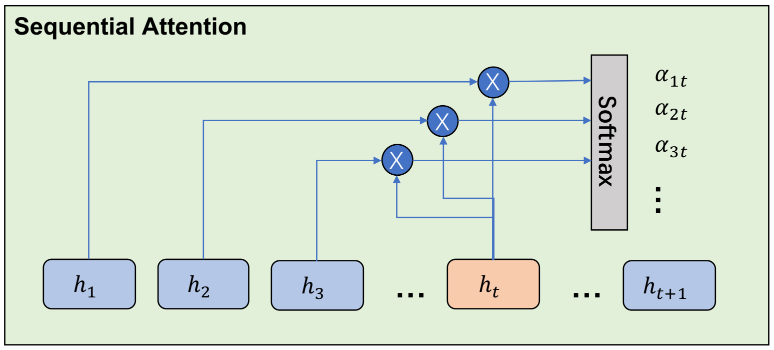

Figure 3: An illustration of the sequential attention module.

3.2.3. Disentangled Attention Module

Inspired by the recent advances in disentangled representation learning [42, 40], we propose our disentangled attention module to learn multiple representations characterizing different latent factors for future predictions at each step of cascade modeling.

We assume that there are K main latent factors that affect the diffusion of an information item. Take Twitter as an example, an information item may spread through K different communities or users are infected by tweets from K topics. We want to disentangle these diverse latent factors from the hidden state $\mathbf{h_t}$ to get the disentangled representations.

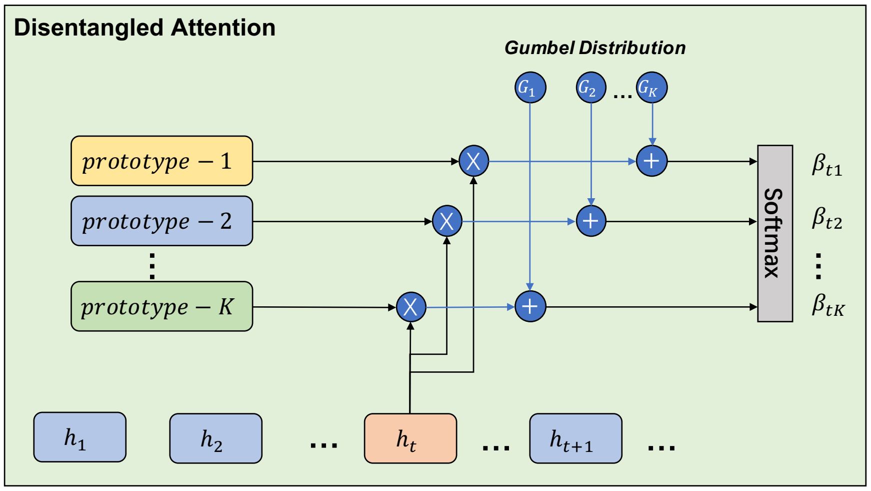

Figure 4: An illustration of the disentangled attention module. Gumbel-Softmax technique is further employed to enhance the disentanglement effect as presented in Section 3.2.4.

To be more specific, we set K prototype embeddings $\mathbf{p_k}$ $(1 \leq k \leq K)$ to represent the K potential factors. We expect these prototype embeddings can capture diverse properties after training. Then for each hidden state h_i, we will compute its similarity with K prototypes by the distance between $\mathbf{h_i}$ and $\mathbf{P}$. The equations are as follows:

$$\begin{aligned} \beta_{ik} &= \frac{exp(\frac{1}{\sqrt{D}} d(\mathbf{h_i}, \mathbf{p_k}))}{\sum_{k'=1}^{K} exp(\frac{1}{\sqrt{D}} d(\mathbf{h_i}, \mathbf{p_{k'}}))} \\ d(\mathbf{h_i}, \mathbf{p_k}) &= \frac{\mathbf{h_i} \cdot \mathbf{p_k}}{\|\mathbf{h_i}\| \|\mathbf{p_k}\|} \end{aligned} \tag{3}$$

where $k = 1, 2, ..., K$ and $i = 1, 2, ..., t$. $\{\mathbf{p_k} \in \mathbb{R}^D : 1 \leq k \leq K\}$ are prototype embeddings of K latent factors. $d(\cdot, \cdot)$ is the distance function, and we select cosine-similarity instead of dot product here because the latter is more vulnerable when it comes to mode collapse [39]. Specifically, the results of dot products are more favorable to those latent classes with higher vector norms. We apply a Layer-Normalization [52] layer on every embedding before computing their distance to avoid Internal Covariate Shift(ICS) and encourage convergence.

Then we apply a Softmax function to compute the attention score β_{ik} for further usage. It is worth noting that by using Softmax, all of the β_{ik} $(1 \leq k \leq$

K) tend to have a similar result. The hidden state h_i will be disentangled to K latent factors equally, which would harm the performance of disentanglement. As shown in Section 3.2.4, we will employ Gumbel-Softmax to alleviate this drawback and enhance the disentanglement effect.

Finally, we can combine the sequence attention and disentanglement attention together, using α_{ij} and β_{ik} to draw the weighted sum result $y_i^{(k)}$ to represent the hidden state under the k-th factor:

$$\mathbf{y_t^{(k)}} = LayerNorm(\sum_{i=1}^{t} \alpha_{it} \cdot \beta_{ik} \cdot \mathbf{h_i}) \tag{4}$$

where $k = 1, 2, ..., K$. When $K = 1$, our method degenerates into a GRU model with an attention mechanism. We name our model as Sequential Information Diffusion model with Disentangled Attention (SIDDA).

3.2.4. Gumbel-Softmax

Recall that we use Softmax in Eq. (3) to compute attention weights of different prototypes. The Softmax function is applied to the cosine-similarity scores, which are in the range $[-1, 1]$. Therefore, the outputs of Softmax tends to be uniform, which means that the hidden state belongs to those latent factors equally. Intuitively, the activation of a specific user belongs to a single latent factor. For example, a tweet one user retweeted usually has one primary topic that interests her. Therefore, the soft attention weights over prototypes should be done in a hard manner, *i.e.*, a sampling way. The most straightforward way is to use a hard selection to produce a one-hot vector to represent the latent label of the hidden state. However, the model will be non-differentiable if we use a hard-selection operation. Fortunately, reparameterizing tricks can help address this dilemma. Gumbel-Softmax distribution is a continuous distribution that approximates samples from a categorical distribution and also works with back-prorogation optimization [53].

Specifically, Gumbel-Softmax trick [20] provides an efficient way of sampling from categorical distribution by adding a random noise:

$$\mathbf{z} = onehot(argmax_i[log\pi_i + G_i]) \tag{5}$$

By sampling G from a fixed distribution and reparameterizing this distribution using π, we are able to perform back-propagation. In detail, $G_i \sim Gumbel(0,1)$ are i.i.d. samples drawn from standard Gumbel distribution. It can be obtained as follows:

$$G_i = -\log(-\log(u_i)) \qquad u_i \sim Uniform(0,1) \tag{6}$$

The $argmax(\cdot)$ operation in Eq. (5) is non-differentiable, so we use Softmax as a differentiable approximation to it:

$$\begin{aligned} z_i &= Softmax[\log \pi_i + G_i] \\ &= \frac{exp((\log \pi_i + G_i)/\tau)}{\sum_{j=1}^{K} exp((\log \pi_j + G_i)/\tau)} \qquad for \quad i = 1, 2, ..., K \end{aligned} \tag{7}$$

The Softmax operation here has a temperature parameter $\tau \in (0, \infty)$. For $\tau \to 0$, samples will be one-hot. For $\tau \to \infty$, both the expectation and the individual samples become uniform. Then Eq. (3) can be written as:

$$\beta_{ik} = \frac{exp[(\frac{1}{\sqrt{D}} d(\mathbf{z_i}, \mathbf{p_k}) + G_k)/\tau]}{\sum_{k'=1}^{K} exp[(\frac{1}{\sqrt{D}} d(\mathbf{z_i}, \mathbf{p_{k'}}) + G_{k'})/\tau]} \tag{8}$$

We will show the effectiveness of Gumbel-Softmax in Section 4.4.2.

3.3. Optimization Objective

We can get K disentangled representations at each timestamp t using Eq. (4). Different hidden state vectors can complement each other and provide more comprehensive information to predict the next infected users. Therefore, we compute the similarity, *i.e.*, dot product, between these K disentangled representations $\mathbf{y_t^{(k)}}$ and node embeddings $\mathbf{X}$ to predict the next activated node. Hence, the loss function can be defined as follows:

$$\begin{aligned} L(\theta, t) &= -\log p_\theta(\mathbf{x_{t+1}} | \mathbf{y_t}) \\ &= -\log \frac{\max_{k \in \{1,2,...,K\}} exp(\frac{1}{\sqrt{D}} \mathbf{x_{t+1}} \cdot \mathbf{y_t^{(k)}})}{\sum_{v' \in V} \max_{k \in \{1,2,...,K\}} exp(\frac{1}{\sqrt{D}} \mathbf{x_{v'}} \cdot \mathbf{y_t^{(k)}})} \end{aligned} \tag{9}$$

We scale the similarity scores by a factor $\frac{1}{\sqrt{D}}$ because there is a layer-normalization layer at the end of the model, and the scaling factor encourages convergence.

It is noteworthy that in Eq. (9), the Softmax function is applied to all of the K disentangled factors. The model is forced to preserve different information in $\mathbf{y_t^{(k)}}(1 \leq k \leq K)$. Therefore, we do not need to use regularization terms to restrict the distribution of $\mathbf{y_t^{(k)}}$ in our model.

4. Experiments

In this section, we will introduce the datasets used in our experiments, and the state-of-the-art baselines which will be compared with our proposed method. We further introduce the evaluation metrics used to evaluate the performance of the baselines and our method. We will design comparative experiments and ablation experiments to show the superiority of our method.

4.1. Datasets

Twitter [54] dataset collects tweets published in October 2010, containing URLs in the message body. It also collects the follower and friend information for all tweeting users, resulting in a social graph. Retweet behavior happens when a user tweeted a URL that has previously appeared in her Twitter feed. The cascades are constructed according to timestamps in tweet metadata. Each URL is considered to be a unique marker of information, spreading among users.

Douban [55] is a Chinese social networking service network where users can share content about movies, music, and books. In our experiment, we consider each book as an information item, and a user is activated if the user reads the book. The network is constructed using users' co-occurrence in social gatherings, where users physically meet.

Memetracker [56, 57] contains millions of online mainstream social media activity, tracking the most frequent phrases, *i.e.*, memes. Memes were information items being shared among websites. One cascade records the diffusion process of one meme according to timestamps. This dataset doesn't have an underlying social graph.

We follow the experiment setting in [21], randomly selecting 80% of cascades for training, 10% for validation, and 10% for test. We only keep the cascades

Table 2: Statistics of datasets.

Dataset	# Cascades	# Nodes	# Links	Average Length
Twitter	3,442	12,627	309,631	32.60
Douban	10,602	23,123	348,280	27.14
Memetracker	12,661	4,709	-	16.24

with lengths shorter than 500. The statistics of datasets are listed in Table 2. Twitter and Douban datasets have an underlying social graph, which will be used by some of the baseline models.

4.2. Baselines and Experimental Settings

We compared our method with several state-of-the-art baselines.

DeepDiffuse [32] is an LSTM based model with an attention mechanism, utilizing node sequence and their corresponding activated time. This model doesn't need granular information, *i.e.*, underlying social network. We replace timestamps with integer activated steps in the cascade because we ignore exact timestamps in our datasets.

Topo-LSTM [15] is a topological recurrent network, which can model diffusion topology as dynamic directed acyclic graphs (DAGs). In our experiment, the cascade structure is extracted from the underlying social graph.

NDM [4] employs convolutional network and attention mechanism for cascade modeling. It makes relax assumptions on the datasets and doesn't need a diffusion graph.

SNIDSA [16] is a novel sequential neural network with structure attention. It utilizes both the sequential nature of a cascade and structural characteristics of the underlying social graph with the help of a gating mechanism.

FOREST [21] is a novel GRU-based information diffusion model. It extracts underlying social graph information using DeepWalk [58] and uses reinforcement learning to incorporate macroscopic prediction ability into itself.

4.3. Evaluation Metrics and Experiment Setting

As pointed out by [15], there can be an arbitrary number of potential candidates, information diffusion prediction can be seen as a retrieval task. We use two popular information retrieval evaluation method:

- **hits**@N This is the rate of top N candidates containing the ground truth, also known as accuracy at top N ($A@N$).
- **map**@N: This is the mean average precision.

The same evaluation metrics are also used [21, 14, 32, 36]. We set $N = 10, 50, 100$ for evaluation.

Since SNIDSA and TopoLSTM need an underlying social graph in the dataset, we exclude them for Memetracker.

We implement our method in PyTorch [59]. We use Adam [60] optimizer for mini-batch gradient descent and use dynamic learning rate [61]. The dropout [62] rate is set to $1e-4$. The temperature parameter of Gumbel-Softmax $\tau = 1.0$. For the dimension of node representations D and the number of disentangled factors K, please refer to Section 4.4.2 about parameters analysis.

4.4. Experimental Results

We design the comparative experiments, ablation experiments, and analyze the parameter sensitivity in this section.

4.4.1. Overall Evaluation

Fig. 11 shows the performance of different methods on three datasets. We select the results when $K = 4$ for our method. For the analysis of parameters, please refer to Section 4.4.2. We have the following observations:

(1) SIDDA outperforms all the state-of-the-art baseline methods consistently and significantly on *hits*@N and *map*@N. The results show that our method can predict the next activated user successfully.

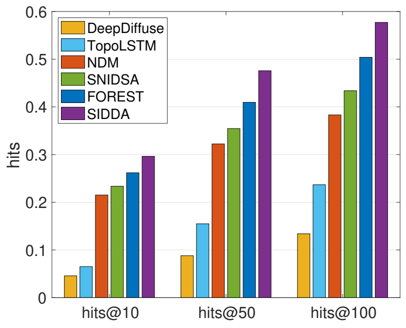

Figure 5: $hits@N$ of Twitter

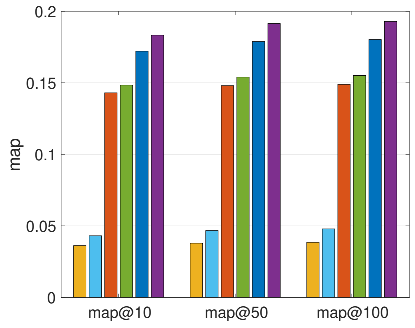

Figure 6: $map@N$ of Twitter

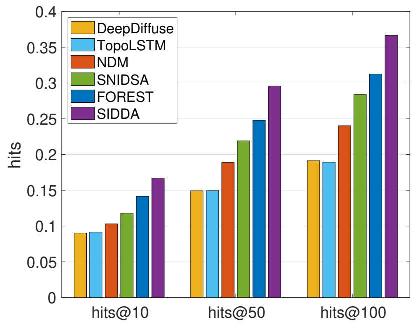

Figure 7: $hits@N$ of Douban

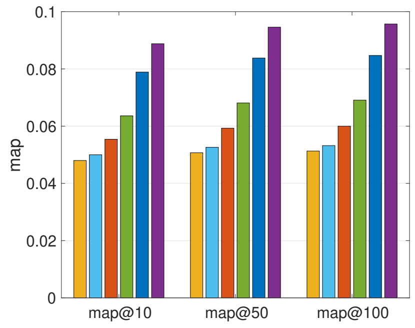

Figure 8: $map@N$ of Douban

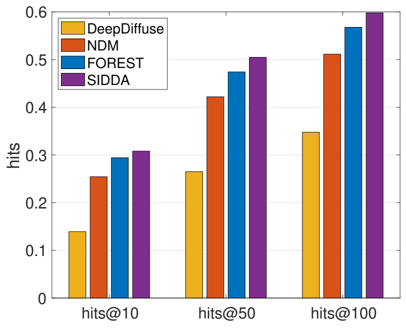

Figure 9: $hits@N$ of Memetracker

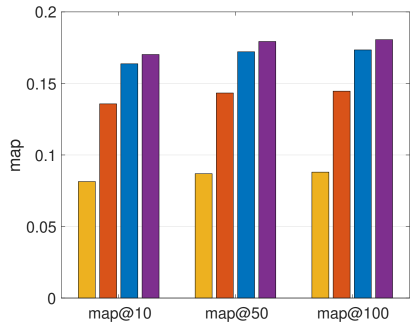

Figure 10: $map@N$ of Memetracker

Figure 11: The overall results, scores are the higher the better.

(2) The performance gain is especially large on datasets with longer cascades. Our method outperforms the best baseline method by more than 13% for the Twitter dataset in terms of $hits@N$. It also gains a relative improvement of more than 17% for $hits@N$ of Douban dataset. We attribute this behavior to the fact that the average length of Twitter (32.60) and Douban dataset (27.14) is longer than that of Memetracker (16.24) dataset.

4.4.2. Ablation Study

***The effect of the number of disentangled factors.*.** We study how the number of disentangled factors K can affect the performance of our method. As shown in Fig.12, our method has the worst performance when $K = 1$. Because when $K = 1$, our model is a GRU model with an attention mechanism, the latent information is not disentangled thoroughly. The performance on Douban increases with K getting larger, and start to decrease when $K > 4$. The results of the Memetracker dataset also have a similar trend. While performance on the Twitter dataset is much better when $K > 4$, showing an increasing trend. This is because Twitter has more diverse topics and may have more factors to disentangle. The reason is also that the average length of Twitter cascades is longer than that of the Douban and Memetracker dataset.

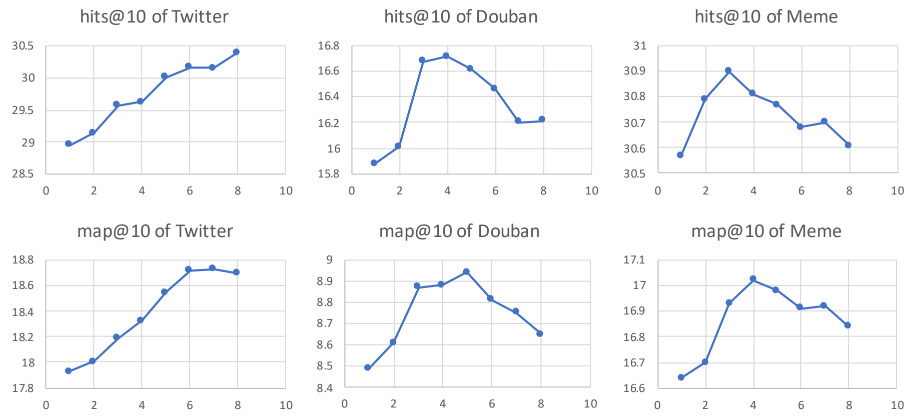

Figure 12: Performance with different numbers of disentangled factors K.

The effect of the number of model dimension.. We also study how the dimension of node representations D can affect the performance. We evaluate our method when $K = 4$ and $D \in \{16, 32, 64, 128\}$. As illustrated in Fig. 13, on Twitter and Memetracker the performance doesn't converge until $D = 128$, possibly because they have larger training sets. Douban dataset converges when $D = 64$ and shows a decreasing trend when D gets larger.

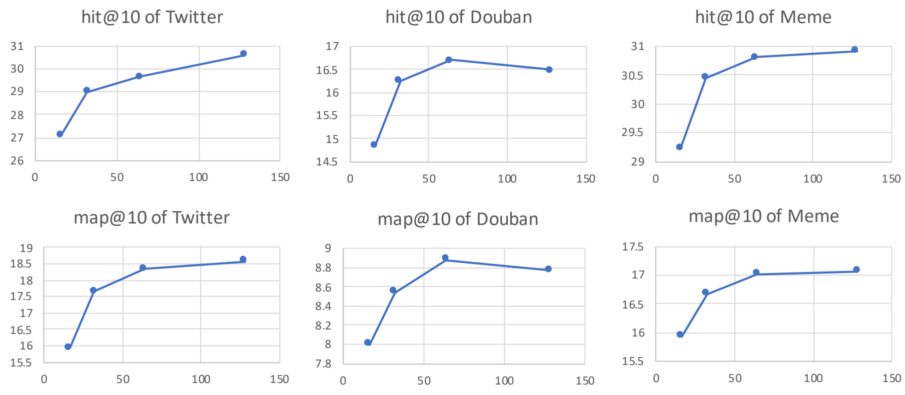

Figure 13: Performance with different dimensions of node representations D.

The effect of Gumbel-Softmax.. We also evaluate the benefit of Gumbel-Softmax proposed in Section 3.2.4. The experiments are conducted when $K = 4$, and the results are shown in Table 3, 4 and 5. The results show that Gumbel-Softmax performs better than the traditional Softmax function.

The effect of sequential attention module.. We evaluate the effect of the sequential attention module by replacing it by an average pooling operation. The results are shown in Table 6, 7 and 8, which demonstrates the superiority of sequential attention module. SIDDA without sequential attention is denoted as SIDDA w/o SA.

Table 3: Benefit of Gumbel-Softmax on Twitter dataset when $K = 4$

	$hits@N(\%)$			$map@N(\%)$		
$N =$	10	50	100	10	50	100
Softmax	29.17	45.99	55.41	18.02	18.80	18.94
Gumbel-Softmax	**29.63**	**47.58**	**57.71**	**18.33**	**19.14**	**19.29**
Improvement	1.58%	3.46%	4.15%	1.72%	1.81%	1.85%

Table 4: Benefit of Gumbel-Softmax on Douban dataset when $K = 4$

	$hits@N(\%)$			$map@N(\%)$		
$N =$	10	50	100	10	50	100
Softmax	16.41	29.38	36.58	8.65	9.23	9.33
Gumbel-Softmax	**16.71**	**29.58**	**36.67**	**8.88**	**9.46**	**9.57**
Improvement	1.83%	0.68%	0.25%	2.66%	1.50%	2.57%

Table 5: Benefit of Gumbel-Softmax on Memetracker dataset when $K = 4$

	$hits@N(\%)$			$map@N(\%)$		
$N =$	10	50	100	10	50	100
Softmax	30.49	49.57	59.38	16.64	17.54	17.68
Gumbel-Softmax	**30.81**	**50.47**	**59.79**	**17.02**	**17.93**	**18.06**
Improvement	1.05%	1.82%	0.69%	2.28%	2.22%	2.15%

Table 6: Benefit of sequential attention on Twitter dataset

	$hits@N(\%)$			$map@N(\%)$		
$N =$	10	50	100	10	50	100
SIDDA w/o SA	25.04	44.36	55.36	13.25	14.11	14.26
SIDDA	**29.63**	**47.58**	**57.71**	**18.33**	**19.14**	**19.29**
Improvement	18.33%	7.26%	4.24%	35.58%	35.65%	35.27%

Table 7: Benefit of sequential attention on Douban dataset

	hits@N(%)			*map*@N(%)		
$N =$	10	50	100	10	50	100
SIDDA w/o SA	15.82	28.80	35.94	8.36	8.95	9.05
SIDDA	**16.71**	**29.58**	**36.67**	**8.88**	**9.46**	**9.57**
Improvement	5.63%	2.71%	1.05%	6.22%	5.80%	5.75%

Table 8: Benefit of sequential attention on Memetracker dataset

	hits@N(%)			*map*@N(%)		
$N =$	10	50	100	10	50	100
SIDDA w/o SA	28.49	47.83	58.00	15.08	15.98	16.12
SIDDA	**30.81**	**50.47**	**59.79**	**17.02**	**17.93**	**18.06**
Improvement	8.14%	5.51%	3.09%	12.86%	12.20%	12.03%

4.4.3. Understanding of Disentangled Factors

To understand the disentangled representations and verify our motivation as illustrated in Fig. 1, we will present additional experiments in this subsection. We feed the embedding of every user into our model, and compute the similarity between the first hidden state and K prototype embeddings, getting each user's most correlated latent factor. Note that in Twitter dataset, we have an underlying social graph. Our goal is to figure out whether users sharing the same main latent factor have higher probability to get connected in the social graph than those users who don't.

Therefore, we compute the connection probabilities for (1) two random users with the same main latent factor and (2) two fully random users, respectively. As shown in Fig. 14, the connection probability for two fully random users is about 0.38%. When using disentangled attention, users having the same main factor are more likely to be connected in the social graph. The probability will increase when K gets larger, which verifies our motivation.

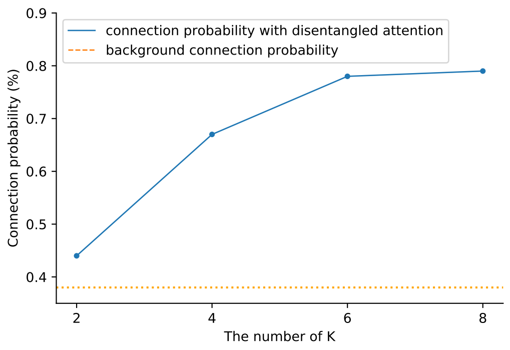

Figure 14: The connection probabilities for (1) two random users with the same main latent factor (blue) and (2) two fully random users (orange).

5. Conclusion

In this paper, we propose a novel diffusion prediction model, SIDDA, which can extract latent factors by learning multiple hidden states at each timestamp. Specifically, we propose sequential attention to capture sequential relations of information cascades. Then we adopt a disentangled attention to learn multiple hidden states representing different latent factors. Additionally, we employ Gumbel-Softmax to further enhance the disentangled effect. The experiments on three real-world datasets show the superiority of our method when compared with state-of-the-art diffusion prediction models.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (No. 62002029) and the Fundamental Research Funds for the Central Universities (No. 2020RC23).

References

[1] C. Li, J. Ma, X. Guo, Q. Mei, Deepcas: An end-to-end predictor of information cascades, in: Proceedings of the 26th international conference on World Wide Web, 2017, pp. 577–586.

[2] X. Chen, F. Zhou, K. Zhang, G. Trajcevski, T. Zhong, F. Zhang, Information diffusion prediction via recurrent cascades convolution, in: 2019 IEEE 35th International Conference on Data Engineering (ICDE), IEEE, 2019, pp. 770–781.

[3] Q. Cao, H. Shen, J. Gao, B. Wei, X. Cheng, Popularity prediction on social platforms with coupled graph neural networks, in: Proceedings of the 13th International Conference on Web Search and Data Mining, 2020, pp. 70–78.

[4] C. Yang, M. Sun, H. Liu, S. Han, Z. Liu, H. Luan, Neural diffusion model for microscopic cascade study, IEEE Transactions on Knowledge and Data Engineering.

[5] Y. Zhao, N. Yang, T. Lin, S. Y. Philip, Deep collaborative embedding for information cascade prediction, Knowledge-Based Systems 193 (2020) 105502.

[6] A. Sankar, X. Zhang, A. Krishnan, J. Han, Inf-vae: A variational autoencoder framework to integrate homophily and influence in diffusion prediction, in: Proceedings of the 13th International Conference on Web Search and Data Mining, 2020, pp. 510–518.

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

[8] J. Leskovec, L. A. Adamic, B. A. Huberman, The dynamics of viral marketing, ACM Transactions on the Web (TWEB) 1 (1) (2007) 5.

[9] H. Li, X. Ma, F. Wang, J. Liu, K. Xu, On popularity prediction of videos shared in online social networks, in: Proceedings of the 22nd ACM international conference on Information & Knowledge Management, 2013, pp. 169–178.

[10] J. Leskovec, L. Backstrom, J. Kleinberg, Meme-tracking and the dynamics of the news cycle, in: Proceedings of SIGKDD, ACM, 2009.

[11] S. Vosoughi, D. Roy, S. Aral, The spread of true and false news online, Science 359 (6380) (2018) 1146–1151.

[12] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, Communications of the ACM 60 (6) (2017) 84–90.

[13] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805.

[14] Y. Wang, H. Shen, S. Liu, J. Gao, X. Cheng, Cascade dynamics modeling with attention-based recurrent neural network., in: IJCAI, 2017, pp. 2985–2991.

[15] J. Wang, V. W. Zheng, Z. Liu, K. C.-C. Chang, Topological recurrent neural network for diffusion prediction, in: 2017 IEEE International Conference on Data Mining (ICDM), IEEE, 2017, pp. 475–484.

[16] Z. Wang, C. Chen, W. Li, A sequential neural information diffusion model with structure attention, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2018, pp. 1795–1798.

[17] T. Che, Y. Li, A. P. Jacob, Y. Bengio, W. Li, Mode regularized generative adversarial networks, arXiv preprint arXiv:1612.02136.

[18] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, P. Abbeel, Infogan: Interpretable representation learning by information maximizing generative adversarial nets, in: Advances in neural information processing systems, 2016, pp. 2172–2180.

[19] C. Yang, M. Sun, X. Yi, W. Li, Stylistic chinese poetry generation via unsupervised style disentanglement, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 3960–3969.

[20] E. Jang, S. Gu, B. Poole, Categorical reparameterization with gumbel-softmax, arXiv preprint arXiv:1611.01144.

[21] C. Yang, J. Tang, M. Sun, G. Cui, Z. Liu, Multi-scale information diffusion prediction with reinforced recurrent networks., in: IJCAI, 2019, pp. 4033–4039.

[22] F. Zhou, X. Xu, G. Trajcevski, K. Zhang, A survey of information cascade analysis: Models, predictions and recent advances, arXiv preprint arXiv:2005.11041.

[23] Q. Zhao, M. A. Erdogdu, H. Y. He, A. Rajaraman, J. Leskovec, 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, 2015, pp. 1513–1522.

[24] J. Cheng, L. Adamic, P. A. Dow, J. M. Kleinberg, J. Leskovec, Can cascades be predicted?, in: Proceedings of the 23rd international conference on World wide web, 2014, pp. 925–936.

[25] S. Gao, J. Ma, Z. Chen, Modeling and predicting retweeting dynamics on microblogging platforms, in: Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, 2015, pp. 107–116.

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

[27] F. Zhou, X. Xu, C. Li, G. Trajcevski, T. Zhong, K. Zhang, A heterogeneous dynamical graph neural networks approach to quantify scientific impact, arXiv preprint arXiv:2003.12042.

[28] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint arXiv:1609.02907.

[29] K. Saito, R. Nakano, M. Kimura, Prediction of information diffusion probabilities for independent cascade model, in: International conference on knowledge-based and intelligent information and engineering systems, Springer, 2008, pp. 67–75.

[30] K. Saito, M. Kimura, K. Ohara, H. Motoda, Learning continuous-time information diffusion model for social behavioral data analysis, in: Asian Conference on Machine Learning, Springer, 2009, pp. 322–337.

[31] S. Bourigault, S. Lamprier, P. Gallinari, Representation learning for information diffusion through social networks: an embedded cascade model, in: Proceedings of the Ninth ACM international conference on Web Search and Data Mining, 2016, pp. 573–582.

[32] M. R. Islam, S. Muthiah, B. Adhikari, B. A. Prakash, N. Ramakrishnan, Deepdiffuse: Predicting the'who'and'when'in cascades, in: 2018 IEEE International Conference on Data Mining (ICDM), IEEE, 2018, pp. 1055–1060.

[33] C. Yuan, J. Li, W. Zhou, Y. Lu, X. Zhang, S. Hu, Dyhgcn: A dynamic heterogeneous graph convolutional network to learn users' dynamic preferences for information diffusion prediction, arXiv preprint arXiv:2006.05169.

[34] S. Molaei, H. Zare, H. Veisi, Deep learning approach on information diffusion in heterogeneous networks, Knowledge-Based Systems 189 (2020) 105153.

[35] Z. Wang, C. Chen, W. Li, Attention network for information diffusion prediction, in: Companion Proceedings of the The Web Conference 2018, 2018, pp. 65–66.

[36] Z. Wang, W. Li, Hierarchical diffusion attention network., in: IJCAI, 2019, pp. 3828–3834.

[37] X. Chen, K. Zhang, F. Zhou, G. Trajcevski, T. Zhong, F. Zhang, Information cascades modeling via deep multi-task learning, in: Proceedings of the

42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 885–888.

[38] Y. Bengio, A. Courville, P. Vincent, Representation learning: A review and new perspectives, IEEE transactions on pattern analysis and machine intelligence 35 (8) (2013) 1798–1828.

[39] J. Ma, C. Zhou, P. Cui, H. Yang, W. Zhu, Learning disentangled representations for recommendation, in: Advances in Neural Information Processing Systems, 2019, pp. 5711–5722.

[40] J. Ma, C. Zhou, H. Yang, P. Cui, X. Wang, W. Zhu, Disentangled self-supervision in sequential recommenders, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 483–491.

[41] L. Hu, S. Xu, C. Li, C. Yang, C. Shi, N. Duan, X. Xie, M. Zhou, Graph neural news recommendation with unsupervised preference disentanglement, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 4255–4264.

[42] J. Ma, P. Cui, K. Kuang, X. Wang, W. Zhu, Disentangled graph convolutional networks, in: International Conference on Machine Learning, 2019, pp. 4212–4221.

[43] H. Kim, A. Mnih, Disentangling by factorising, arXiv preprint arXiv:1802.05983.

[44] S. Gidaris, P. Singh, N. Komodakis, Unsupervised representation learning by predicting image rotations, arXiv preprint arXiv:1803.07728.

[45] J.-T. Hsieh, B. Liu, D.-A. Huang, L. F. Fei-Fei, J. C. Niebles, Learning to decompose and disentangle representations for video prediction, in: Advances in Neural Information Processing Systems, 2018, pp. 517–526.

[46] H. Yu, C. Park, C. Yang, Q. Zhu, D. Kim, J. Han, Unsupervised differentiable multi-aspect network embedding, in: 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD2020), KDD, 2020.

[47] Y. Liu, X. Wang, S. Wu, Z. Xiao, Independence promoted graph disentangled networks., in: AAAI, 2020, pp. 4916–4923.

[48] S. Molaei, H. Zare, H. Veisi, Deep learning approach on information diffusion in heterogeneous networks, Knowledge-Based Systems 189 (2020) 105153.

[49] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, arXiv preprint arXiv:1406.1078.

[50] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.

[51] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv preprint arXiv:1409.0473.

[52] J. L. Ba, J. R. Kiros, G. E. Hinton, Layer normalization, arXiv preprint arXiv:1607.06450.

[53] W. Wong, What is gumbel-softmax?, `https://towardsdatascience.com/what-is-gumbel-softmax-7f6d9cdcb90e`.

[54] N. O. Hodas, K. Lerman, The simple rules of social contagion, Scientific reports 4 (2014) 4343.

[55] E. Zhong, W. Fan, J. Wang, L. Xiao, Y. Li, Comsoc: adaptive transfer of user behaviors over composite social network, in: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, 2012, pp. 696–704.

[56] J. Leskovec, L. Backstrom, J. Kleinberg, Meme-tracking and the dynamics of the news cycle, in: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, 2009, pp. 497–506.

[57] J. Leskovec, A. Krevl, SNAP Datasets: Stanford large network dataset collection, `http://snap.stanford.edu/data` (Jun. 2014).

[58] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, Association for Computing Machinery, New York, NY, USA, 2014, p. 701–710. `doi:10.1145/2623330.2623732`.

[59] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in pytorch.

[60] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015.

[61] X.-H. Yu, G.-A. Chen, S.-X. Cheng, Dynamic learning rate optimization of the backpropagation algorithm, Trans. Neur. Netw. 6 (3) (1995) 669–677. `doi:10.1109/72.377972`.

[62] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.