Sequential Information Diffusion Model with Disentangled Attention

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

[a]*Beijing University of Posts and Telecommunications, China*

Abstract

This is abstract.

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

1. Introduction

The phenomenon of information diffusion, dubbed as information cascade, is ubiquitous in our daily lives, *e.g.*, the spread of a 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

*Corresponding author

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

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. For example, as illustrated in Fig.1

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 prediction.

To be more specific, we propose Sequential Information Diffusion model with Disentangled Attention (SIDDA) by applying a sequential attention module balabala how to do. Consequently, we are able to learn multiple hidden state vectors characterizing different latent factors for future predictions at each step of cascade modeling. The multiple hidden state vectors can complement each other and provide more comprehensive information to predict 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 innovatively use Gumbel-Softmax to realize the sampling process in a differentiable way, and enhance the disentanglement effect.
- The experiments on three real-world datasets demonstrate that our proposed method outperforms state-of-the-art baselines significantly.

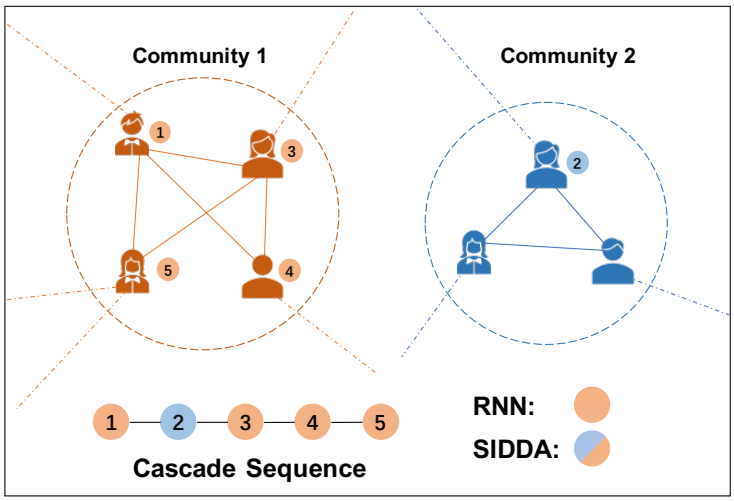

Figure 1: Illustration of mode collapse problem.

2. Related Work

2.1. Information Diffusion Prediction

There are two kinds of tasks associated with information diffusion prediction: (1) Predict the next activated node in the cascade. (2) Study the growth of cascades. In some papers [2, 4, 20, 21], these tasks are classified as microscopic and macroscopic tasks.

2.1.1. Macroscopic Information Diffusion Prediction

Macroscopic information diffusion prediction is also known as popularity prediction, predicting cascades size in the future, [22] or predicting whether some information diffusion will become viral. Therefore, this task can be seen as a classification problem or a regression problem. Previously, related works are based on feature-based approaches [23] and generative approaches [24]. In recent years, with the success of deep learning, methods based on Recurrent Neural Network (RNN) are proposed, *e.g.*, DeepCas [1], Deephawkes [25]. Some researchers also introduce Graph Neural Networks (GNN) to model social graph

and diffusion paths, *e.g.*, CoupledGNN [3]. CasCN [2] combines Graph Convolutional Network [26] and RNN together, modeling diffusion path as a dynamic graph.

2.1.2. Microscopic Information Diffusion Prediction

Our work focuses on microscopic information diffusion prediction, forecasting the next activated node given the information cascades. We classify previous works into two categories: IC-based methods and deep-learning-based methods.

IC model [27] is one of the most widely used models, which assumed independent probability between nodes. This kind of model can not only predict the next activated node but also generate information cascades. Various extensions like continuous-time IC [28], Embedded-IC [29], are proposed by incorporating timestamps and representation learning.

With the development of deep learning, various kinds of neural networks are proposed to solve micro information diffusion prediction. Generally, RNN, GNN, and attention mechanism are widely used, and get promising results. Topo-LSTM [15] proposes a novel LSTM to model tree-structured cascades. CYAN-RNN [14] and Deep-Diffuse [30] apply RNN and design complex attention mechanisms. They also take timestamps into consideration with the help of temporal point processes. There are also some models totally based on attention mechanism to avoid the non-sequential problem, *e.g.*, DAN [31], Hi-DAN [32]. Some works [33, 34, 35, 36] introduced the concept of heterogeneous information networks [37] into information diffusion, constructing heterogeneous information networks based on the social graph. Both of the user and information items can be seen as nodes in this kind of graphs. FOREST [20] is an RNN-based method, which extracts nodes feature using DeepWalk [38] on the underlying social graph.

In addition to RNN, GNN, and attention mechanism, there are also some novel model structures are proposed in recent years. HID [39] integrates node representation learning and multi-scale modeling together. It is a universal framework, which can be layered on top of all existing information diffusion

methods with a node representation learning component. Semi-supervised learning can also be used to capture node features in the cascades, *e.g.*, DCE [5] and Inf-VAE [6].

Besides, some works focus on both macroscopic and microscopic information diffusion prediction. FOREST [20] makes use of reinforcement learning to incorporate macroscopic prediction ability into itself. DMT-LIC [40] is a multi-task learning framework with a shared-representation layer. It designs two different task layers to handle these two tasks.

2.2. Disentangled Representation Learning

Disentangled representation learning has been successfully used in various fields, aiming to learn and disentangle different latent features hidden in the observed data [41]. In the field of recommendation system, a few works are proposed to learn disentangled item representations [42]. [43] performs self-supervision in the latent space, getting disentangled intentions from item sequences for the 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. GNUD [44], making use of DisenGCN [45], is the first try to explore disentanglement in news recommendation. Besides, disentangled representation learning is also successfully used in the field of computer vision [46, 47, 48] and the modeling of graph-structured data [49, 50].

As far as we know, no one has used the disentangled representation learning method in information diffusion prediction.

3. Method

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

3.1. Preliminary

Given a node set V and a cascade set C, we have $V = \{v_1, v_2, ..., v_N\}$ and $C = \{c_1, c_2, ..., c_M\}$. N is the size of the node set, and M is the size of cascade. Each cascade $c_i \in C$ is a sequence of node $\{v_1^i, v_2^i, ..., v_{|c_i|}^i\}$, ordered by their activated time. Following the same setting in [4, 15, 16, 20], we only keep the order of nodes and ignore their activated timestamps, leaving the use of timestamp for future work.

Information diffusion prediction can be formulated as: given previously activated nodes sequence $\{v_1^i, v_2^i, ..., v_k^i\}$ in cascade c_i for $k = 1, 2, ..., |c_i| - 1$, predict the next activated node v_{k+1}^i. The task is to build a model that is able to learn the conditional probability function $p(v_{k+1}^i | \{v_1^i, v_2^i, ..., v_k^i\})(1 \leq i \leq M)$. We will omit the superscript for simplification in the rest of the paper.

All of the notations mentioned above can be seen in Table1.

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, $C = \{c_1, c_2, ..., c_M\}$, $c_i = \{v_1^i, v_2^i, ..., v_{\|c_i\|}^i\}$
M	the size of cascade set
K	the number of disentangled preferences
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 Structure

The framework of our method is proposed in Fig.2 Initially, we embed every node into node embedding. 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 at each time step. Then we set K prototype embeddings to represent latent labels of nodes, *i.e.*, communities in social networks, research topics in citing networks. Then a disentangled attention mechanism is applied to generate K disentangled representations for each hidden state. Moreover, Gumbel-Softmax is proposed to realize the sampling process and enhance the disentanglement effect.

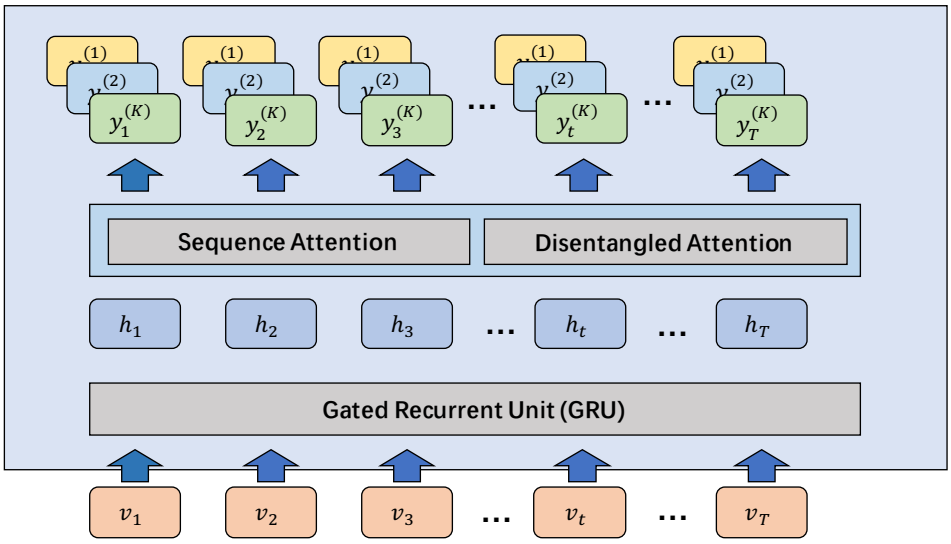

Figure 2: Feed node embeddings in to a Gated Recurrent Unit(GRU) model to get hidden state at time t, then use proposed disentangled attention to derive K disentangled representations $y_t^{(k)}(1 \leq k \leq K)$ with the help of K prototype embeddings.

3.2.1. Recurrent Network

Recurrent neural networks(RNNs) have shown great potentials in modeling sequence data. Previous works [15, 16, 20, 14, 30, 51] used RNNs as their basis to model information diffusion cascades. At first, 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, it can output a hidden state $\mathbf{h_t}$ capturing the dynamic information of all of the previously activated nodes.

Gated Recurrent Unit (GRU) [52] can solve the vanishing gradient problem to some extent compared with a standard RNN. It can be considered as a variation of the LSTM [53] but is computationally cheaper. Therefore, we select GRU as our basic model.

Essentially, the reset gate decides how much of the past hidden state to forget, using current node embedding $\mathbf{x_t}$ and last hidden state $\mathbf{h_{t-1}}$.

$$\mathbf{r} = \sigma(\mathbf{W_{ir}x_t} + \mathbf{b_{ir}} + \mathbf{W_{hr}h_{t-1}} + \mathbf{b_{hr}}) \tag{1}$$

The update gate determines how much of the previous information needs to be taken into account. This formula is the same as that of the reset gate, but its usage and the parameters are different.

$$\mathbf{z} = \sigma(\mathbf{W_{iz}x_t} + \mathbf{b_{iz}} + \mathbf{W_{hz}h_{t-1}} + \mathbf{b_{hz}}) \tag{1a}$$

The memory content will use the reset gate to store relevant information. Then a non-linear activation function tanh is applied.

$$\mathbf{n} = \tanh\left(\mathbf{W_{in}x_t} + \mathbf{b_{in}} + \mathbf{r} \odot (\mathbf{W_{hn}h_{t-1}} + \mathbf{b_{hn}})\right), \tag{1b}$$

Finally, the update gate determines what to output and pass down to the network.

$$\mathbf{h_t} = (1 - \mathbf{z}) \odot \mathbf{n} + \mathbf{z} \odot \mathbf{h_{t-1}} \tag{1c}$$

In our model, $\odot$ is the Hadamard(element-wise) product between vectors. $\mathbf{W}_* \in \mathbb{R}^{D \times D}$, $\mathbf{b}_* \in \mathbb{R}^D$, D is the dimension of node embeddings and hidden states.

3.2.2. Sequence Attention

The attention mechanism is originally used in neural machine translation [54], and widely used in various kinds of the machine learning tasks. Previous works [31, 32] used self-attention [55] as their basis to model information cascades and got promising results. Every hidden state $\mathbf{h_t}$ contains information

about the input node sequence with a focus around position t. Attention mechanism can capture the sequence relations automatically, and assign different attention weight to hidden state $\mathbf{h_i}(i < t)$.

The sequence attention α_{ij} is computed by:

$$\alpha_{ij} = \frac{exp(\frac{1}{\sqrt{D}} e_{ij})}{\sum_{k=1}^{t} exp(\frac{1}{\sqrt{D}} e_{ik})} \tag{2}$$
$$e_{ij} = d(h_i, h_j)$$

where $\frac{1}{\sqrt{D}}$ is a scaling coefficient, $h_i \in \mathbb{R}^D$. $d(\cdot, \cdot)$ is a distance function, and we use dot product here.

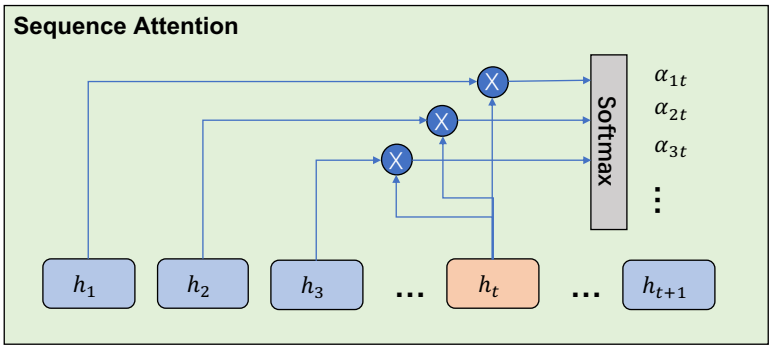

Figure 3: Use self-attention here to model the effect of previous hidden state.

3.2.3. Disentanglement Attention

The model only has the node sequence as the input, but do not know the message item being transmitted. If we can explore the underlying content of the message from node sequences by observing the cascades, we can improve our model's prediction ability. Above mentioned sequence attention can capture the relation between hidden states at different timestamps. In this section, we propose our disentangled attention to learn disentangled representations.

We virtually classify users' preferences into K topics. Take Twitter, for example, one user may have interests in sports, movies, games, etc, tweets also have these topics respectively. We want to disentangle these diverse user preferences from the hidden state $\mathbf{h_t}$ to get the disentangled hidden representations.

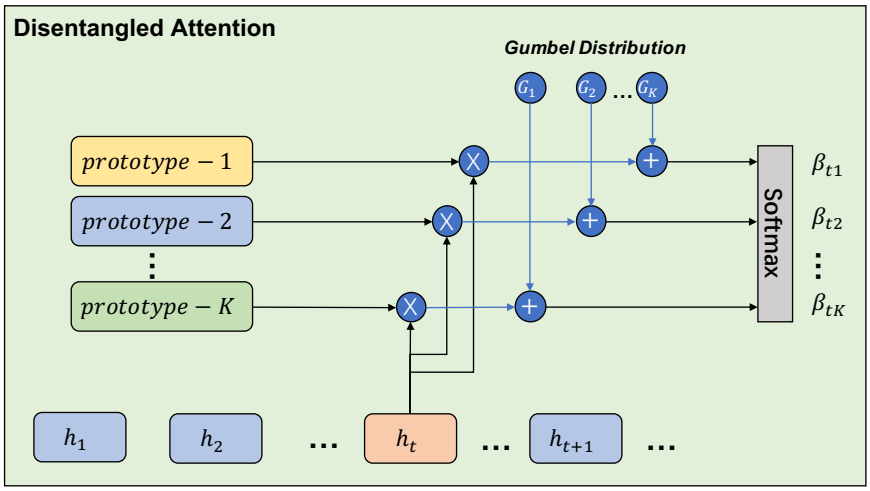

Figure 4: Compute the cosine-similarity between the hidden state h_t and K prototype embeddings, getting the disentanglement attention weights $p_{k|t}(1 \leq k \leq K)$. To enhance the disentanglement effect, Gumbel-Softmax is proposed in section 3.2.4 to approximate samples from a categorical distribution.

Firstly, we set K preference embeddings $\mathbf{p_k}(1 \leq k \leq K)$ to represent the virtual topics of hidden states. We expect these preference embeddings can capture diverse properties after training. Then at each timestamp i, we can cluster the hidden state h_i into K classes by computing the distance between $\mathbf{h_i}$ and $\mathbf{P}$. These virtual topics can be seen as the latent labels of hidden state. The equations are as follows:

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

$$d(\mathbf{h_i}, \mathbf{h_j}) = \frac{\mathbf{h_i} \cdot \mathbf{h_j}}{\|\mathbf{h_i}\| \|\mathbf{h_j}\|}$$

where $k = 1, 2, ..., K$ and $i = 1, 2, ..., t$. $\{\mathbf{p_k} \in \mathbb{R}^D : 1 \leq k \leq K\}$ are preference embeddings under K latent categories. $d(\cdot, \cdot)$ 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 [42]. Specifically, the results of dot products are more favorable to those latent classes with higher vector norms. We apply a Layer-Normalization [56] layer on every embedding before computing

their distance to avoid Internal Covariate Shift(ICS) and encourage convergence.

These disentangled attention weights are computed as a classification task, classifying the hidden state into K virtual topics. Finally, 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 $\beta_{ik}(1 \leq k \leq K)$ tend to have a similar result. The hidden state h_i are disentangled to K latent topics equally, which means users tend to have similar preferences over all of these K topics. To solve this drawback, we propose Gumbel-Softmax in Section.3.2.4, enhancing the disentanglement effect.

Then 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 user's k_{th} preference:

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

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

We use $Softmax$ in Eq.3.2.3 to compute attention weights of different preferences. The $Softmax$ function is applied to the cosine-similarity results, which are in $[-1, 1]$. The result of $Softmax$ tends to be uniform, which means that the hidden state belongs to those latent preferences equally. As pointed out by [49], a user may simultaneously have multiple preferences and interests, like a Twitter user may have interests in movies, sports, and games. In one cascade, however, there must be some main reasons this user gets activated. The tweet one user retweeted must have one primary topic that interests her.

Therefore, preference classification should be done in a discrete 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 is not differentiable anymore if we use a hard-selection

operation. Reparameterizing tricks can help us. The Gumbel-Softmax distribution is a continuous distribution that approximates samples from a categorical distribution and also works with back-prorogation optimization.

Gumbel-Softmax trick [57] 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 avoid the stochastic node during back-propagation. $G_i \sim Gumbel(0, 1)$ 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 are perfectly discrete, *i.e.*, one-hot. For $\tau \to \infty$, both the expectation and the individual samples become uniform. Then Eq.3.2.3 can be written as:

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

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

3.3. Optimization Objective

We can get K disentangled representations with different preferences at each timestamp t using Eq.4. We compute the similarity, *i.e.*, dot product, between these K preference embeddings $\mathbf{y_t^{(k)}}$ and node embeddings $\mathbf{X}$ to predict the next

activated node. So the loss function can be defined as follows:

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

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 preferences. The model is forced to preserve different information in $y_t^{(k)}(1 \leq k \leq K)$. Therefore, we do not need to use regularization terms to restrict the distribution of $y_t^{(k)}$ in our model.

4. Experiment

In this section, we will introduce the datasets used in our experiment, 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 the comparative experiments, ablation experiments to show the superiority of our method with graph visualization.

4.1. Datasets

Twitter [58] 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 [59] 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 [60, 61] 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.

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

We follow the experiment setting in [20], randomly selecting 80% of cascades for training, 10% for validation, and 10% for test. We only keep the cascades 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

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

DeepDiffuse [30] 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 [20] is a novel GRU-based information diffusion model. It extracts underlying social graph information using DeepWalk [38] and uses reinforcement learning to incorporate macroscopic prediction ability into itself.

4.3. Evaluation Metrics and Experiment Setting

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 [20, 14, 30, 32]. We set $N = 10, 50, 100$ for evaluation.

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

We implement our method in PyTorch [62]. We use Adam [63] optimizer for mini-batch gradient descent and use dynamic learning rate [64]. The dropout [65] 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 preference K, please refer to Sec.4.4.2 about parameters analysis.

4.4. Experiment 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 result when $K = 4$ for our method. For the analysis of parameters, please refer to Sec.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.

(2) The performance gain is especially large on datasets with longer cascades. Our method outperforms the best baseline method by more than 13% for Twitter dataset in term of $hits@N$. It also gains the 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 preferences.. We study how the number of disentangled preferences 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 Twitter users have much more preferences to disentangle. The reason is also that the average length of Twitter cascades is longer than that of the Douban and Memetracker dataset.

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,

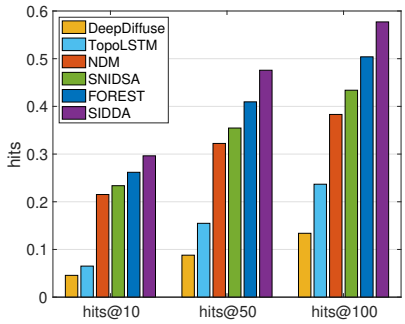

Figure 5: *hits*@N of Twitter

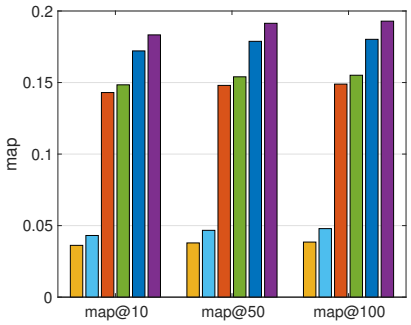

Figure 6: *map*@N of Twitter

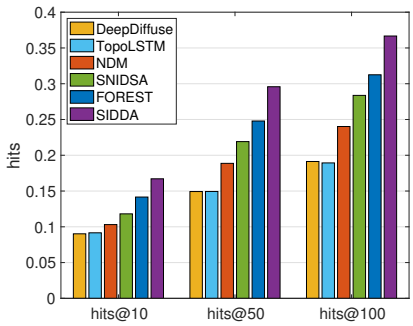

Figure 7: *hits*@N of Douban

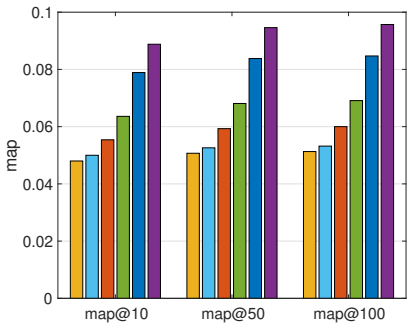

Figure 8: *map*@N of Douban

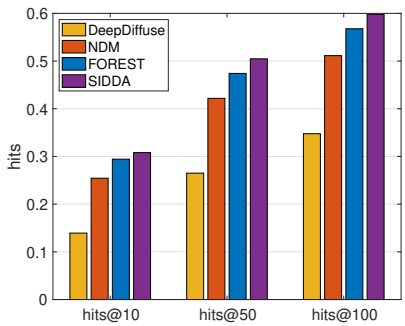

Figure 9: *hits*@N of Memetracker

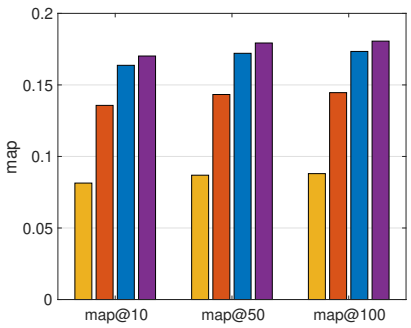

Figure 10: *map*@N of Memetracker

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

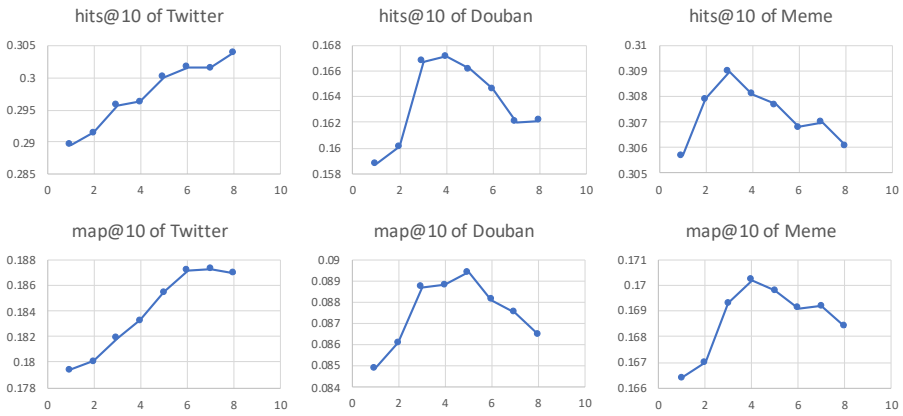

Figure 12: Performance with different numbers of disentangled preferences K.

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 get larger.

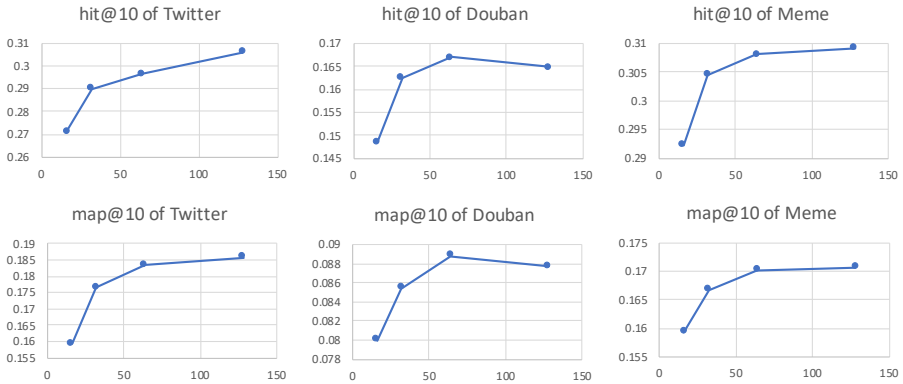

Figure 13: Performance with different dimensions of node representations D.

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

performs better than the traditional Softmax function.

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%

The effect of attention mechanisms.. We evaluate the effect of above proposed attention mechanisms. Without sequential attention and disentangled attention, our model degenerate into a GRU diffusion model. The results are shown in Table.6,7,8, which demonstrates the superiority of our attention mechanisms. SIDDA without attention mechanisms is noted as SIDDA w/o att.

Table 6: Benefit of attention mechanisms on Twitter dataset

	$hits@N(\%)$			$map@N(\%)$		
$N =$	10	50	100	10	50	100
SIDDA w/o att	28.36	44.76	54.36	17.82	18.58	18.70
SIDDA	**29.63**	**47.58**	**57.71**	**18.33**	**19.14**	**19.29**
Improvement	4.48%	6.30%	6.16%	2.86%	3.01%	3.16%

Table 7: Benefit of attention mechanisms on Douban dataset

	$hits@N(\%)$			$map@N(\%)$		
$N =$	10	50	100	10	50	100
SIDDA w/o att	15.32	27.45	34.78	8.22	8.77	8.82
SIDDA	**16.71**	**29.58**	**36.67**	**8.88**	**9.46**	**9.57**
Improvement	9.07%	7.76%	5.43%	8.03%	7.87%	8.50%

Table 8: Benefit of attention mechanisms on Memetracker dataset

	$hits@N(\%)$			$map@N(\%)$		
$N =$	10	50	100	10	50	100
SIDDA w/o att	29.65	48.51	57.96	16.27	17.15	17.29
SIDDA	**30.81**	**50.47**	**59.79**	**17.02**	**17.93**	**18.06**
Improvement	3.91%	4.04%	3.16%	4.61%	4.55%	4.45%

4.4.3. Case Study

To further show the disentangled effect, we examine how the user preferences are classified. We assume that users connected in the social graph tend to have similar preferences. For example, in Twitter dataset, we have an underlying social graph constructed by following and friend information. We expect that users sharing the same main preference have higher probability to get connected in the social graph than those users who don't. Therefore, we feed all of the user embeddings into our model, and compute the similarity between the first hidden state and K preferences embeddings, getting one user's main preference. We sample 100,000 pairs of users in the Twitter dataset, and compute the connected probability of these two users.

As illustrated in Fig.14, we draw the connected probability of user pairs. If disentangled attention is not used, the connected rate is about 0.38%. When using disentangled attention, users having the same main preferences are more likely to be connected in the social graph. The trend in Fig.14 is consistent with that in Fig.13.

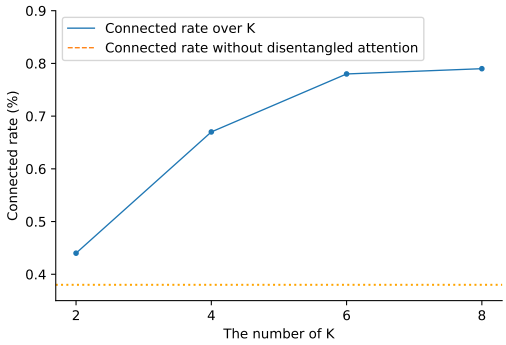

Figure 14: The connected rate of users in the same class. If there is no disentangled attention, connected rate is 0.38%.

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 a sequence attention to capture sequential relations of information cascades. Then we adopt a disentangled attention to learn multiple hidden states under different user preferences. Additionally, to further enhance the disentangled effect, we realize preferences classification in a sampling way using Gumbel-Softmax. The experiments on three real-world datasets show the superiority of our method when compared with state-of-the-art diffusion prediction models.

Acknowledgement

References

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] 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.

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

[22] 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.

[23] 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.

[24] 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.

[25] 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.

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

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

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

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

[34] 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.

[35] Y. Su, X. Zhang, S. Wang, B. Fang, T. Zhang, S. Y. Philip, Understanding information diffusion via heterogeneous information network embeddings, in: International Conference on Database Systems for Advanced Applications, Springer, 2019, pp. 501–516.

[36] 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.

[37] C. Shi, S. Y. Philip, Heterogeneous information network analysis and applications, Springer, 2017.

[38] 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.

[39] H. Zhou, S. Xu, Z. Fu, G. de Melo, Y. Zhang, M. Kapadia, Hid: Hierarchical multiscale representation learning for information diffusion, Corpus 3 (2020) U1.

[40] 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.

[41] 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.

[42] 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.

[43] 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.

[44] 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.

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

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

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

[48] 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.

[49] 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.

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

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

[52] 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.

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

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

[55] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in neural information processing systems, 2017, pp. 5998–6008.

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

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

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

[59] 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.

[60] 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.

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

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

[63] 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.

[64] 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`.

[65] 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.