Information Sciences 528 (2020) 280–293

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Recurrent neural variational model for follower-based influence maximization

Huimin Huang[a], Zaiqiao Meng[a,b,*], Shangsong Liang[a]

[a] School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China
[b] University of Glasgow, Glasgow, United Kingdom

ARTICLE INFO

Article history:
Received 22 August 2019
Revised 2 January 2020
Accepted 12 April 2020
Available online 15 April 2020

Keywords:
Recurrent neural network
Variational autoencoder
Influence diffusion
Social networks

ABSTRACT

Influence Maximization, aiming at selecting a small set of seed users in a social network to maximize the spread of influence, has attracted considerable attention recently. Most of the existing influence maximization algorithms focus on the diffusion model of one single-entity, which assumes that only one entity is propagated by users in social network. However, the diffusion situations in real world social networks often involve multiple entities, competitive or complementary, spreading through the whole network, and are more complex than the situations of single independent entity.

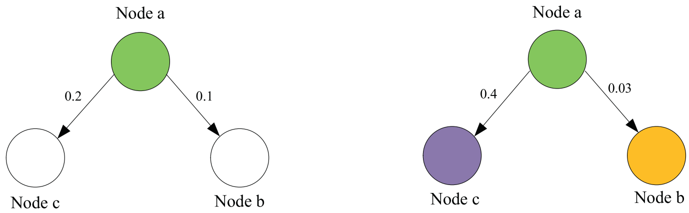
In this paper, we propose a novel optimization problem, namely, the follower-based influence maximization, which aims to promote a new product into the market by maximizing the influence of a social network where other competitive and complementary products have already been propagating. We tackle this problem by proposing a Recurrent Neural Variational model (RNV) and a follower-based greedy algorithm (RNVGA). The RNV model dynamically tracks entity correlations and cascade correlations through a deep generative model and recurrent neural variational inference, while the RNVGA algorithm applies the greedy approach for submodular maximization and efficiently computes the seed node set for the target product. Extensive experiments have been conducted to evaluate effectiveness and efficiency of our method, and the results show the superiority of our method compared with the state-of-the-art methods.

1. Introduction

Influence Maximization, aims at selecting a small set of seed users in a social network to maximize the spread of influence. As a technique to help solve social issues (e.g. preventing terrorist attacks and anticipating natural hazards) and optimize business performance (e.g. viral marketing), influence maximization has attracted considerable attention recently. It has been shown that using friend-to-friend influence to promote products are more effective than advertisements [1] and many e-commerce sites have achieve successes by exploring influence maximization [2]. Some companies attract users to share the information about their products through social media, and successively make more users to be aware of their products. Most of the previous work focus on the diffusion model of one single-entity, which assumes that only one entity

* Corresponding author.
E-mail addresses: zaiqiao.meng@glasgow.ac.uk (Z. Meng), liangshs5@mail.sysu.edu.cn (S. Liang).

https://doi.org/10.1016/j.ins.2020.04.023
0020-0255/



(a) Traditional IM. There is only one product (represented by color green) propagated in the network. Assuming node a has been activated by Product Green, the probabilities that node a activates node b and node c on Product Green are 0.1 and 0.2 respectively.

(b) IM with competitive and complementary products. Assume node a, node b, node c have activated by Product Green, Product Yellow, Product Purple separately. The relationship between Product Green and Product Yellow is competitive and the relationship between Product Green and Product Purple is complementary. The probability that node a activates node b on Product Green decreases to 0.03 and the probability that node a activates node c on Product Green increases to 0.4.

Fig. 1. A simple example to describe the difference between traditional IM and the IM with competitive and complementary entities.

(e.g. product) is propagated by users in social network. However, the diffusion situations in real world social networks often involve multiple entities which are competitive or complementary, and are more complex than the situations of single independent entity.

Fig. 1 illustrates how the Influence Maximization (IM) with competitive and complementary entities differs from traditional IM. Suppose we have a simple network and node *a* can influence node *b*. In the traditional IM (Fig. 1(a)), there is only one entity (Product Green represented by color green) propagated in the network, and node *a* can be activated on Product Green through the influence from node *a* to node *b*. In the IM with competitive and complementary entities (Fig. 1(b)), assume node *a*, node *b*, node *c* have activated by Product Green, Product Yellow (represented by color yellow), Product Purple (represented by color purple) separately. The relationship between Product Green and Product Yellow is competitive and the relationship between Product Green and Product Purple is complementary. The probability that node *a* activates node *b* on Product Green will be weakened due to the competitive relationship between Product Green and Product Yellow, while the probability that node *a* activates node *c* on Product Green will be strengthened because of the complementary relationship between Product Green and Product Purple.

Most of the previous works assume that there is only one entity introduced into the network (or there are multiple entities with pure competitive relationship) [3,4]. Recently, influence maximization problem has been extended to consider both the competition and the complementarity of different entities. Competitive and complementary influence maximization problem is studied in literature [5–7] etc., and their works focus on finding the top-k influential nodes which maximize the expected spread of all products in the network. Their models enjoyed impressive success, obtaining more accurate predictions than models of pure competition. However, these models demonstrate a number of major drawbacks: (a) most of them focus on the relationship between two propagation entities, ignoring a more complex phenomenon that the present-time-step infection depends on the entire activation sequence, (b) they fail to capture the correlations among the conditional cascades of propagation, by assuming that the conditional cascades are independent, which is usually unrealistic, (c) they cannot efficiently model the complex and dynamic interactions among entities across sequential time steps by assigning correlations of entities randomly or simply setting deterministic values to correlations of entities.

To address these issues, in this paper we study a new problem, called the follower-based influence maximization problem, that finds a set of seed users maximizing the spread influence of a given entity (e.g. product) in a network where competitive and complementary entities are already being introduced. This problem setting is more in line with the real-world situation. For example, if a company (the follower) is planing to use the viral marketing to introduce a new product to users via social networks (e.g. Twitter), the company should consider the influence of other product in the social network when selecting the seed users, since some competitive and complementary products may have already been introduced.

Modelling entity interactions and how these interactions change with time of new cascade generating, is extremely challenging. Not only is there potentially significant interactions between any two entities, but also these interactions could change over time when a user is activated by either entity through links in network. So the influence that another entity may have on a user may change considerably with each new entity activating the user [8].

To address these challenges, we utilize an entire sequence of entities activation to capture the posterior distribution of target item activation. We propose a Recurrent Neural Variational (RNV) model to capture entity correlations, cascade correlations and dynamics of these correlations. Although the standard RNN is able to predict sequential output and handle the problem of variable-length input and output, it is not applicable to model randomness or variability observed in highly structured data. This is due to the entirely deterministic transformation structure of standard RNN. By using the RNV model, we can construct highly flexible non-linear mapping from random latent state to observed output and capture the strong and complex dependencies among the output variables over time. Our RNV efficiently predicts the generation probability of target-item activation by learning hidden knowledge in entire sequence of item activation of each user, through a deep generative process. Moreover, to capture the sequential relationship between conditional cascades and threshold changes, we design a threshold updating strategy, taking advantage of the observation that threshold expresses the user's reluctance in adopting an item [9]. A simple example is that after a user buys an item, the chances that he buys another competitive item tend to decrease, and consequently, the threshold of the latter will increase.

To summarize, we make the following contributions:

(1) We propose a new problem called the follower-based influence maximization. We show that this problem is NP-hard under Multiple-entity Linear Threshold from Follower's perspective (MLTF) diffusion model. We prove the monotonicity and the submodularity of this problem and provide a greedy algorithm to solve it with approximation guarantee.

(2) We design a threshold updating strategy to capture the sequential relationship between conditional cascades and threshold changes. We propose RNV to capture the dynamics of multiple-entity interactions. It brings up new insights into the influence spread process. To the best of our knowledge, it is the first time that recurrent neural variational inference is introduced to solve influence maximization problem.

(3) We verify the effectiveness of the proposed method on three real-world and publicly available datasets, and the result demonstrate that our method significantly outperforms the state-of-the-art methods.

The rest of this paper is organized as follows. Section 2 reviews previous work. Section 3 presents our problem formulation. Section 4 details the proposed method with a threshold updating strategy, a recurrent neural variational model (RNV) and a follower-based influence maximization algorithm (RNVGA). Section 5 lists the experimental setup. Section 6 reports our experiment results and analysis. Section 7 concludes the paper.

2. Previous work

2.1. Influence maximization

Single-entity influence maximization has been studied extensively. Leskovec et al. [10] propose the CELF algorithm, which is based on the submodularity of the objective function in Linear Threshold (LT) and Independent Cascade (IC) models and employs lazy-forward optimization. Chen et al. [11] propose the NewGreedy and MixedGreedy algorithms using the idea of reducing the scale of the propagation map. Cheng et al. [12] propose the StaticGreedy algorithm, which guarantees the optimization of the submodularity of the target by repeatedly using the generated subgraph, and improves the efficiency of the original Greedy algorithm by two orders of magnitude. Another line of methods, such as Depth-based heuristic algorithm [11], PMIA [13], MIA [14], are proposed to estimate the node influence through the path function. Tang et al. proposes the TIM+ [15] and IMM [16] models, which sample nodes from the network propagation graphs to establish a reverse reachable set. Huang et al. propose a community-based and topic-aware influence maximization method [17]. Morone et al. utilize the non-backtracking matrix to study the problem of finding the minimal node set which causes the spread of information to the whole network [18]. Goyal et al. propose a new model using the credit distribution that directly leverages available propagation traces to learn how influence flows in the network and uses this to estimate expected influence spread [19]. Qiu et al. take a user's local network as the input to a graph neural network for learning her latent social representation [20]. However, these methods do not consider more complex situation of social interactions that involves multiple propagating entities, while multiple-entity diffusion is certainly more general in the real world.

2.2. Competitive and complementary viral marketing

Recent studies on influence diffusion have extended single-entity propagation to multiple-entity propagation with pure competitive relationship. Most previous works assume propagation entities are purely competitive. Carnes et al. [21], from the follower's perspective, propose two models for the propagation of two competing entities, given the fixed budget of the follower and the initial seed node set for each entity. From the perspective of social network host, Lu et al. [22] propose an algorithm which takes account of both collective expected spread maximization and fairness of seed allocation of each entity by setting optimization objective as a MinMax function. [3,23] use the game theoretic strategy to solve the competitive influence maximization problem. Zhu and Li [4], Khan et al. [24], study the problem of competitive profit maximization from the host perspective.

Recently, some work has looked into both the competition and the complementarity among entities. Ou et al. [25] study the problem of competitive and cooperative influence maximization of multiple entities with game theory. Litou et al. [7] propose a greedy algorithm for multiple-entities influence maximization, assigning correlations of entities randomly. Lu et al. [9] study the relationship between two entities, propose complementary influence maximization problem, and design an approximation algorithm, which assigns deterministic values to the correlations of entities that causes the complex correlations among entities can not be modelled efficiently. Zarezade et al. [6] use a Hawkes process to model spread of cascades in competitive and complementary networks, and prove the objective function is convex. However this method has a very high computational complexity. Myers and Leskovec [5] use Bayesian method to model the process of multiple-entity influence spread, and assume the item activation sequences are conditionally independent which is infeasible in the real world.

2.3. VAE, RNN, VRNN

In 2013, Kingma et al. [26] first proposed the Variational Auto-Encoder (VAE), which is an effective model to estimate complex multimodal distributions in the data space. Jurgen Schmidhuber [27] proposed the Recurrent Neural Network (RNN), which is a special type of neural network that learns temporal dependency from input of time series and predicts next output in a sequence. Zhang et al. [28], Zhang and Wang [29], Zhang et al. [30] studied stability criteria for the RNNs with time-varying delay. Because of the entire determinacy of internal transition structure of standard RNN, it is not applicable to model complex multimodal distributions. Recently, Variational Recurrent Neural Network (VRNN) [31–34] has been proposed, which integrates random variables into the RNN hidden state to capture dependencies and causalities between latent random variables across sequential timesteps. VRNN is more relevant to our work, and we will introduce it in detail in Section 4.

3. Problem formulation

In this section, we first briefly review the classical linear threshold diffusion model (single-entity linear threshold diffusion model), then introduce our new MLTF diffusion Model (i.e. the Multiple-entity Linear Threshold from Follower's perspective diffusion model), and then give the formulation of follower-based influence maximization problem.

In classical linear threshold diffusion model, given the social network $\mathcal{G}(\mathcal{U}, \mathcal{E})$ and the weight $w(v, u) \in [0, 1]$ of each directed edge $(v, u) \in \mathcal{E}$, where $\mathcal{U}$ denotes the set of nodes and $\mathcal{E}$ denotes the set of directed edges, node u is influenced jointly by each in-neighbor v according to weight $w_{v,u}$ such that $\Sigma_{v \in N(u)} w_{v,u} \leq 1$ ($w_{v,u} = 0$ if there is no edge (v, u) in the directed graph $\mathcal{G}$ or node u is inactive), where $N(u)$ denotes the in-neighbors of node u. Each node u has an initial activation threshold θ_u, which is chosen uniformly at random from the interval [0,1]. In step $t = 0$, with an initial set of active nodes $\mathcal{S}$ and all other nodes in the graph inactive, the diffusion process starts deterministically in discrete steps. In sept t ($t \geq 1$), the nodes activated in step $t - 1$ remain active, and the inactive nodes $u \in \mathcal{U} \setminus S_{t-1}$ can be activated if $\sum_{v \in N(u) \cap S_{t-1}} w_{v,u} \geq \theta_u$, where S_{t-1} denotes the nodes set that has been activated before step t. Otherwise, node u remains inactive in step t. The propagation process runs until there is not any new node being activated.

Definition 1 (Independent, competitive and complementary propagating entities)**.** Let $p(a_u^i)$ denote the probability that user u is activated by item i and $p(a_u^i | a_u^j)$ denote the conditional probability that user u is activated by item i on the condition that user u has been already activated by item j. For item $i, j \in \mathcal{I}$, we say that: (1) item i is independent to item j iff $\forall u \in \mathcal{U}$, $p(a_u^i | a_u^j) = p(a_u^i)$, (2) item i competes with item j iff $\forall u \in \mathcal{U}$, $p(a_u^i | a_u^j) < p(a_u^i)$, (3) item i complements item j iff $\forall u \in \mathcal{U}$, $p(a_u^i | a_u^j) > p(a_u^i)$.

The design of MLTF diffusion model not only relies on the essential elements of the classical Linear Threshold diffusion model, but also makes theory closer to the practice, where influence diffusions may occur for more than one product (including independent, competitive or complementary products). Moreover, the thresholds should be dynamic due to the dynamics of cascade interactions. Thus, MLTF diffusion model characterizes entity interactions and the dynamics of cascade interactions.

In MLTF diffusion model, given the social network $\mathcal{G}(\mathcal{U}, \mathcal{E})$ and the weight $w(v, u) \in [0, 1]$ of each directed edge $(v, u) \in \mathcal{E}$, where $\mathcal{U}$ denotes the set of nodes and $\mathcal{E}$ denotes the set of directed edges, node u is influenced on item $i \in \mathcal{I}$ jointly by each in-neighbor v according to the weight $w_{v,u}$ such that $\Sigma_{v \in N(u)} w_{v,u} \leq 1$ ($w_{v,u} = 0$ if there is no edge (v, u) in the directed graph $\mathcal{G}$ or node u is inactive to item i). For simplicity, the weights of directed edge (v, u) in promoting different items are all set as $w(v, u)$. Each node u has an initial activation threshold θ_u^i on item i, which is chosen uniformly at random from the interval [0,1]. In step $t = 0$, with an initial set of active nodes $\mathcal{S}$ and all other nodes in the graph inactive, the diffusion process starts deterministically in discrete steps. In sept t ($t \geq 1$), the nodes activated in step $t - 1$ remain active, and the inactive nodes $u \in \mathcal{U} \setminus S_{t-1}$ can be activated on item i if $\sum_{v \in N(u) \cap S_{t-1}} w_{v,u} \geq \theta_u'^i$, where S_{t-1} denotes the nodes set that has been activated before step t and $\theta_u'^i$ is the updated threshold of user u on item i caused by the dynamics of cascade interactions. Otherwise, node u remains inactive in step t. The propagation process runs until there is no any new node being activated.

Essentially, the differences between classical linear threshold model and MLTF model are for classical linear threshold model, single entity propagated, each user with an initial threshold keeping fixed once chosen, but for MLTF model, multiple entities propagated, each user with an initial threshold on each item, which is dynamic caused by the dynamics of cascade interactions happening when new entity activating the user. And the motivation of introducing our new model is that we observe in reality, there is follower (company), who wants to use viral marketing to introduce a new product into the market where competitive and complementary products are already being introduced.

Definition 2 (MLTF Influence maximization)**.** Given a social network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a log of past purchasing $\mathbb{D}$ (activation sequence of each user can be obtained from $\mathbb{D}$), the initial thresholds of each user on each item $\theta_u^{i_l}$ and the target item i^*,

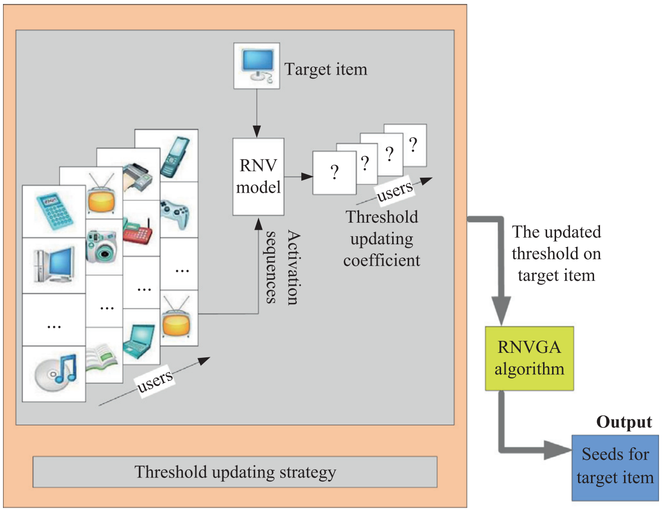


Fig. 2. Overview of the proposed method.

influence maximization under MLTF diffusion model is to find a subset of $\mathcal{V}$ for i^*, i.e., $\mathcal{S}eed_{i^*}$, such that

$$\mathcal{S}eed_{i^*} = \arg\max_{\mathcal{S}_{i^*}} \sigma(\mathcal{S}_{i^*}|\mathbb{D}), \tag{1}$$

where $\sigma(\mathcal{S}_{i^*}|\mathbb{D})$ denotes the expected spread of seed set $\mathcal{S}_{i^*}$ (the expected number of all active nodes at the end of the diffusion process, using $\mathcal{S}_{i^*}$ as seed set), given the activation sequence of each user.

4. The proposed method

Analyze the section of problem formulation, we can see that the key to solve the proposed problem is to find the updated threshold $\theta_u'^{i^*}$ of each user on target item i^*. To predict the updated threshold, we propose a method with a threshold updating strategy, a recurrent neural variational model (RNV) and a follower-based influence maximization algorithm (RNVGA). Fig. 2 presents the overview of our method. The input is a purchasing log $\mathbb{D}$ representing the interactions between users and item, a social network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and the initial thresholds of each user on target item $\boldsymbol{\theta}^{i^*}$. And the output is the seed for target item. Through RNV model, we can get threshold updating coefficient, and along with our threshold updating strategy, we further obtain the updated thresholds of each user on target item. Then using our greedy-based algorithm RNVGA, we compute and output the seed for target item. The proposed method will be detailed in the following subsections.

4.1. Threshold updating strategy

In MLTF diffusion model, nodes in social network can be activated by multiple items sequentially, which may be independent, competitive or complementary propagating entities. We observe that there could be substitute goods and complementary goods propagated through the social network. In the case of two items, buying a substitute good could set back buying another, which is typically of the same kind. Buying a complementary good could boost buying another, which tends to be adopted together. Buying a product could affect buying another to the degree, that can be characterized by a conditional probability $p(a_u^i|a_u^j)$. Moreover, the effect degree is user specific and can not be estimated as uniform and static. Specifically, for each user $u \in \mathcal{U}$, if $p(a_u^i|a_u^j) = 0$, item i and item j are mutually perfect substitute goods and the relationship between them are purely competitive.

Similarly, in the case of multiple items, adopting a product could affect adopting the others to some degree, which can be characterized by a probability $p(a_u^{i_r}|a_u^{i_1}, \dots a_u^{i_{l-1}}, a_u^{i_l})$, assuming that before activated by item i_r, user u has been sequentially activated by l items $i_1, i_2, \dots, i_l \in \mathcal{I} \setminus \{i_r\}$. Moreover, the effect degree is user specific and can not be estimated as uniform and static. We characterize this dynamics of cascade interactions as the update of threshold. Intuitively, competitive relationship will set back user to buy competitive item and accordingly the threshold will rise to a definite degree. Conversely, complementary relationship will boost user buy complementary item and accordingly the threshold will reduce to a definite degree. Thus, we design a threshold updating strategy, the goal of which is to capture user's threshold change after user has been activated by an activation sequence of items. Assuming that before activated by item i_r, user u has been sequentially activated by l items $i_1, i_2, \dots, i_l \in \mathcal{I} \setminus \{i_r\}$, the updating strategy of threshold $\theta_u^{i_r}$ is as follows:

$$
\begin{aligned}
(\theta_u^{i_r})^{i_1} &= \theta_u^{i_r} \frac{p(a_u^{i_r})}{p(a_u^{i_r}|a_u^{i_1})}, \\
(\theta_u^{i_r})^{i_2} &= (\theta_u^{i_r})^{i_1} \frac{p(a_u^{i_r}|a_u^{i_1})}{p(a_u^{i_r}|a_u^{i_1}, a_u^{i_2})}, \\
&\dots, \\
(\theta_u^{i_r})^{i_l} &= (\theta_u^{i_r})^{i_{l-1}} \frac{p(a_u^{i_r}|a_u^{i_1}, \dots a_u^{i_{l-1}})}{p(a_u^{i_r}|a_u^{i_1}, \dots, a_u^{i_{l-1}}, a_u^{i_l})},
\end{aligned} \tag{2}
$$

where $(\theta_u^{i_r})^{i_l}$ denotes the threshold of user u on item i_r after he has been activated by item sequence $i_1, i_2, \dots, i_l \in \mathcal{I} \setminus \{i_r\}$. In the above equations, for all $l \in \{2, 3, \dots, l\}$, after iteratively substituting the value of $(\theta_u^{i_r})^{i_{l-1}}$ in equation $(\theta_u^{i_r})^{i_l}$, we can get:

$$
(\theta_u^{i_r})^{i_l} = \theta_u^{i_r} \frac{p(a_u^{i_r})}{p(a_u^{i_r}|a_u^{i_1}, \dots a_u^{i_{l-1}}, a_u^{i_l})}. \tag{3}
$$

Given the initial threshold $\theta_u^{i_r}$, Eq. (3) demonstrates that the threshold update of $\theta_u^{i_r}$ under condition of activation sequence $(i_1, i_2, \dots, i_l)$, depends on the second factor of the right side of Eq. (3), i.e., $\frac{p(a_u^{i_r})}{p(a_u^{i_r}|a_u^{i_1}, \dots a_u^{i_{l-1}}, a_u^{i_l})}$, called the threshold updating coefficient, which will be predicted by our RNV model.

4.2. RNV Model

Compared with previous works on competitive and complementary influence maximization which focus on finding the top-k influential nodes maximizing the expected spread of all products in the network, the advantage of our MLTF diffusion model is that it considers and addresses the problem faced by a company that wants to use viral marketing (influence maximization) to introduce a new product into a market where competitive and complementary product are already being introduced. In other words, with MLTF diffusion model, we aim at finding the top-K influential nodes for the target product in the network where other competitive and complementary products have already been propagating. Thus, we need to build a model to capture entity correlations, cascade correlations and dynamics of these correlations.

The goal of RNV model is to predict the threshold updating coefficient on target item. Considering the dynamics and sequentiality of conditional cascades as well as the non-linear, subtle relationships in the data, we propose to utilize the framework of sequential variational autoencoder i.e., Variational Recurrent Neural Network (VRNN) [33]. VRNN can capture dependencies and causalities between latent random variables across sequential timesteps, integrating Variational Autoencoder (VAE) and Recurrent Neural Networks (RNN) to implement the stochastic realization of RNN.

Assume $u \in \mathcal{U} = 1, \dots, M$ be a user and $i \in \mathcal{I} = 1, \dots, N$ be an item by which the user can be activated. Let $\boldsymbol{X} \in \{0, 1\}^{N \times M}$ be activation matrix and x_u be the column (binary) with all the item activation status for user u. Given x_u, we let $I_u = \{i \in \mathcal{I} | x_{u,i} = 1\}$ with $N_u = |I_u|$. Similarly, $U_i = \{u \in \mathcal{U} | x_{u,i} = 1\}$ with $M_i = |U_i|$. Considering the temporal relationships within $\boldsymbol{X}$, we assume that the timing information $\boldsymbol{T} \in \mathbb{R}_+^{N \times M} \cup \{0\}$ with $t_{u,i}$ being the time when user u is activated by item i. And we assume that $t_{u,i} = 0$ if $x_{u,i} = 0$. We define $i <_u j$ to denote $t_{u,i} < t_{u,j}$. We let $x_{u(t)}$ denote the tth item in I_u in the sequence sorted by $<_u$ such that $1 \le t \le N_u$, and $x_{u(1:t)}$ denote the sequence $x_{u(1)}, \dots, x_{u(t)}$. We define the latent factors $z_{u(t)}$ $(1 \le t \le N_u)$, each of which corresponds to a $x_{u(t)}$. Obviously, for each user u, our task is to infer latent factor $z_{u(t)}$ through sequential variational autoencoder to predict temporal-dependency conditional probabilities.

In RNN framework, given the input of the user u's item activation sequence $x_{u(1:t)}$, temporal-dependency conditional probabilities that each event relies on the previous event can be learned through a recurrent neural network:

$$
\begin{aligned}
h_t &= f_\gamma(x_{(t)}, h_{t-1}), \\
p(x_{u(t)}|x_{u(1:t-1)}) &= g_\tau(h_t),
\end{aligned} \tag{4}
$$

where f is a deterministic state transition function, γ is parameters of f, g is a deterministic function mapping from the hidden state to the probability distribution over outputs, and τ is parameters of g. Thus, given a sequence $x_{u(1:T)}$, the joint sequence probability distribution can be factorized as a product of temporal-dependency conditional probabilities because of the independence of each time step:

$$
p(x_{u(1:T)}) = \prod_{t=0}^{T} p(x_{u(t)}|x_{u(1:t-1)}) \tag{5}
$$

Although RNN is a type of neural network which is suitable to predict next output in a sequence and can handle the problem of variable-length input and output, it is not applicable to model randomness or variability observed in highly structured data, using conditional output probability model in standard RNN. This is due to the entirely deterministic transformation structure of standard RNN. To construct highly flexible non-linear mapping from random laten state to observed output, VRNN has been proposed recently [33], which extends VAE into a recurrent framework and jointly model high-dimensional

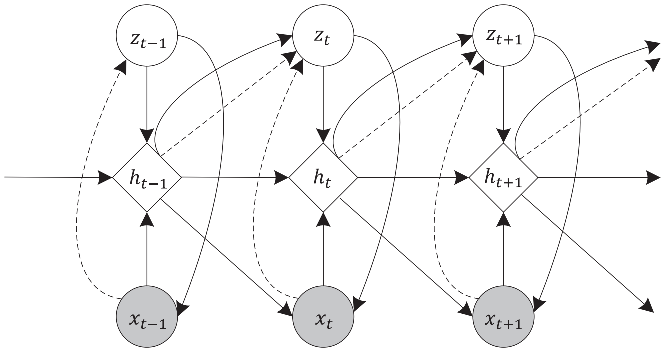


Fig. 3. Graphical model for RNV. Solid and dashed lines represent generative and inference process, respectively. Gray and white nodes represent observed variables and latent variables, respectively.

sequences as well as approximate infer of posterior distribution. Inspired by VRNN, we approximate temporal-dependency posterior of latent variable related to $z_{u(t)}$ by introducing generation networks and inference networks. The graphical representation of our RNV is shown in Fig. 3. According to the graphical representation, given a sequence $x_{u(1:T)}$, the joint likelihood of RNV can be factorized as:

$$p(x_{u(1:T)}, z_{u(1:T)}) = \prod_{t=0}^{T} p(x_{u(t)}|z_{u(t)})p(z_{u(t)}). \tag{6}$$

Instead of inferring the joint distribution, i.e., Eq. (6), we are more interested in inferring approximate posterior distribution over the next item activating user u with the previous item sequence that activated u. To do this, we first consider the prior on the latent random variable $z_{u(t)}$:

$$\begin{aligned} &z_{u(t)} \sim \mathcal{N}(\boldsymbol{\mu}_{0,t}, \mathrm{diag}(\boldsymbol{\sigma}_{0,t}^2)), \\ &[\boldsymbol{\mu}_{0,t}, \boldsymbol{\sigma}_{0,t}] = \varphi_\lambda^{\mathrm{pri}}(\boldsymbol{h}_{t-1}), \end{aligned} \tag{7}$$

where $\boldsymbol{\mu}_{0,t}$ and $\boldsymbol{\sigma}_{0,t}$ denote the mean and covariance of the prior distribution respectively, and $\varphi_\lambda^{\mathrm{pri}}$ is the neural network modeling the prior distribution. The latent random variable $z_{u(t)}$ can capture temporal-dependency relation using $\boldsymbol{h}_{t-1}$ in generating network $\varphi_\lambda^{\mathrm{pri}}$.

Then, let us consider the generating distribution, which is not only depend upon $z_{u(t)}$ but also upon $\boldsymbol{h}_{t-1}$:

$$\begin{aligned} &x_{u(t)}|z_{u(t)}\mathcal{N}(\boldsymbol{\mu}_{x,t}, \mathrm{diag}(\boldsymbol{\sigma}_{x,t}^2)), \\ &[\boldsymbol{\mu}_{x,t}, \boldsymbol{\sigma}_{x,t}] = \varphi_\lambda^{\mathrm{dec}}(\varphi_\lambda^{z}(z_{(t)}), \boldsymbol{h}_{t-1}), \end{aligned} \tag{8}$$

where $\boldsymbol{\mu}_{x,t}$ and $\boldsymbol{\sigma}_{x,t}$ denote the mean and covariance of the generating distribution respectively, $\varphi_\lambda^{\mathrm{dec}}$ is the neural network modeling the generating distribution and φ_λ^{z} is the embedding network of $z_{(t)}$. According to Fig. 1, hidden state variable $\boldsymbol{h}_t$ depend upon previous state variable $\boldsymbol{h}_{t-1}$, $x_{(t)}$ and $z_{(t)}$. We use the following recurrent neural function to update hidden state.

$$\boldsymbol{h}_t = \varphi_\phi^{\mathrm{rec}}(\varphi_\lambda^{x}(x_{(t)}), \varphi_\lambda^{z}(z_{(t)}), h_{t-1}) \tag{9}$$

where $\varphi_\phi^{\mathrm{rec}}$ is a highly flexible state transition function, ϕ is parameter set of φ^{rec}, φ_λ^{x} is the embedding network of $x_{(t)}$.

Unlike the standard VAE, the posterior distribution in our RNV is not a standard Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ but a Gaussian distribution whose mean $\boldsymbol{\mu}_\lambda(t)$ and covariance $\boldsymbol{\sigma}_\lambda(t)$ depend on both x_t and $\boldsymbol{h}_{t-1}$:

$$\begin{aligned} &z_{u(t)}|x_{u(1:t-1)} \sim \mathcal{N}(\boldsymbol{\mu}_{z,t}, \mathrm{diag}(\boldsymbol{\sigma}_{z,t}^2)), \\ &[\boldsymbol{\mu}_{z,t}, \boldsymbol{\sigma}_{z,t}] = \varphi_\lambda^{\mathrm{enc}}(\varphi_\lambda^{x}(x_{(t)}), \boldsymbol{h}_{t-1}), \end{aligned} \tag{10}$$

where $\boldsymbol{\mu}_{z,t}$ and $\boldsymbol{\sigma}_{z,t}$ denote the mean and covariance of the generating distribution respectively, and $\varphi_\lambda^{\mathrm{enc}}$ is the neural network capturing the approximate posterior distribution.

Following VAE, we can compute the tractable standard Evidence Lower Bound (ELBO) for the inference as follows.

$$\mathcal{L}(q) = \mathbb{E}_{q(\mathbf{z}_{\le N_u}|\mathbf{x}_{\le N_u})}[-\log q(\mathcal{Z}|\mathcal{X}) + \log p(\mathcal{X}, \mathcal{Z})] \tag{11}$$

where $\mathcal{X}$ and $\mathcal{Z}$ are the set of observable variables and latent variables respectively. By assuming the probability independence between users, we can get:

$$\mathcal{L}(\theta, \lambda; \mathbf{x}) = \sum_{u}\sum_{t=1}^{N_u}\{\mathbb{E}_{q(\mathbf{z}_{\le N_u}|\mathbf{x}_{\le N_u})}[\log p(\mathbf{x}_t|\mathbf{z}_{\le t}, \mathbf{x}_{<t})] \\ - \mathrm{KL}(q(\mathbf{z}_t|\mathbf{x}_{\le t}, \mathbf{z}_{<t})||p(\mathbf{z}_t|\mathbf{x}_{<t}, \mathbf{z}_{<t}))\} \tag{12}$$

The posteriors are assumed to be Gaussian distribution, thus analytical forms of KL terms of Eq. (12) can be integrated. But the expectation terms can not be integrated analytically. To solve this problem, we use Stochastic Gradient Variational Bayes (SGVB) estimator [26] to estimate the expectation w.r.t. $q(\mathbf{z}_{\le N_u}|\mathbf{x}_{\le N_u})$ by applying a reparametrization trick [26], which samples temporal-dependency auxiliary noise variables $\boldsymbol{\epsilon}$ according to a fixed distribution $p(\boldsymbol{\epsilon})$, and obtains $\mathbf{z}$ by means of a differentiable transformation depending on $\boldsymbol{\lambda}, \boldsymbol{\epsilon}, \mathbf{x}$. Thus, the ELBO can be yielded as:

$$\mathcal{L}(\phi, \lambda; \mathbf{x}) \simeq \widetilde{\mathcal{L}}(\phi, \lambda; \mathbf{x}) = \sum_{u}\sum_{t=1}^{N_u}\left\{\frac{1}{B}\sum_{b=1}^{B}(\log p(\mathbf{x}_t|\mathbf{z}_{\le t}^{b}, \mathbf{x}_{<t})) - \mathrm{KL}(q(\mathbf{z}_t|\mathbf{x}_{\le t}, \mathbf{z}_{<t})||p(\mathbf{z}_t|\mathbf{x}_{<t}, \mathbf{z}_{<t}))\right\} \tag{13}$$

where B denotes the size of samplings, k represents the k-th sample and $\mathbf{z}_{\le t}^{b} = \boldsymbol{\mu}_{z,t} + \mathrm{diag}(\boldsymbol{\sigma}_{z,t}^{2})) \cdot \boldsymbol{\epsilon}_{z,t}^{b}$ with $\boldsymbol{\epsilon}_{z,t}^{b}$ being an auxiliary noise variable sampled from $\mathcal{N}(0, I)$. Then, an estimator of the ELBO of the full dataset can be constructed using minibatch technique:

$$\mathcal{L}(\phi, \lambda; \mathbf{x}) \simeq \widetilde{\mathcal{L}}^{M}(\phi, \lambda; \mathbf{x}) = \frac{1}{E}\sum_{i=1}^{E}\widetilde{\mathcal{L}}(\phi, \lambda; \mathbf{x}^{(i)}) \tag{14}$$

Following [26], we can use single datapoints to make parameter updates as long as the minibatch size E is large enough, e.g., E=128. Then, we use Stochastic Gradient Descent (SGD) to update the parameters. After the convergence of parameters, we can infer $p(\mathbf{z}_t)$ with prior network, and infer $p(\mathbf{z}_t|\mathbf{x}_{(1:t-1)})$ with encoder network.

RNV model is outlined in Algorithm 1. The input of the algorithm is a log of past purchasing $\mathbb{D}$ (the previous item activation sequences can be extracted from $\mathbb{D}$) and the initial thresholds of each user on target item $\boldsymbol{\theta}^{i^*}$, and the output of it is the vector of updated thresholds on target item $\boldsymbol{\theta}'^{i^*}$ with $\theta_u'^{i^*}$ being the updated threshold of user u on target item i^*. The algorithm starts with a initialization of parameters $\boldsymbol{\phi}, \boldsymbol{\lambda}$. Then it executes the process of recurrent neural variational inference to estimate the parameters $\boldsymbol{\phi}, \boldsymbol{\lambda}$. After that, it infers $p(\mathbf{z}_t)$ with prior network, and infer $p(\mathbf{z}_t|\mathbf{x}_{(1:t-1)})$ with encoder network. Next, it update the thresholds of each user on target item according to the following equation:

$$(\theta_u^{i^*})^{x_{u(1:t-1)}} = \theta_u^{i^*}\frac{p(z_{u(t)})}{p(z_{u(t)}|x_{u(1:t-1)})} \tag{15}$$

where $(\theta_u^{i^*})^{x_{u(1:t-1)}}$ denotes the update of threshold $\theta_u^{i^*}$ under the condition of sequence $x_{u(1)}, \ldots, x_{u(t-1)}$. Finally, the algorithm return the vector of updated thresholds on target item $\boldsymbol{\theta}'^{i^*}$.

Algorithm 1 Algorithm to outline RNV.

Input: a log of past purchasing $\mathbb{D}$, the initial thresholds of each user on target item $\boldsymbol{\theta}^{i^*}$
Output: threshold updating matrix $\boldsymbol{\theta}'^{M \times N}$.

$\boldsymbol{\phi}, \boldsymbol{\lambda} \leftarrow$ Initialize parameters;
repeat
    $\mathbf{X}^M \leftarrow$ Random minibatch;
    $\boldsymbol{\sigma} \leftarrow$ Random samples form noise distribution $p(\boldsymbol{\sigma})$;
    $\mathbf{g} \leftarrow \nabla_{\boldsymbol{\theta},\boldsymbol{\lambda}}\widetilde{\mathcal{L}}^M(\theta, \lambda; \mathbf{X}^M; \boldsymbol{\sigma})$ //Stochastic gradient descent for minibatch estimator;
    $\boldsymbol{\phi}, \boldsymbol{\lambda} \leftarrow$ Update parameters using gradients $\mathbf{g}$;
until convergence of parameters $(\boldsymbol{\phi}, \boldsymbol{\lambda})$
$p(\mathbf{z}_t) \leftarrow$ infer $\boldsymbol{z}_t$ with prior network;
$p(\mathbf{z}_t|\mathbf{x}_{(1:t-1)}) \leftarrow$ infer posterior distribution with encoder network;
$//\mathbf{z}_t = z_{u_1(t)}, z_{u_2(t)}, \ldots, z_{u_m(t)}$
for each user $u = 1, 2, \ldots, m$ **do**
    $(\theta_u^{i^*})^{x_{u(1:t-1)}} = \theta_u^{i^*}\frac{p(z_{u(t)})}{p(z_{u(t)}|x_{u(1:t-1)})}$;
    //update the thresholds of each user on target item
end for
return $\boldsymbol{\theta}'^{i^*}$;

4.3. RNVGA: Follower-based influence maximization algorithm

Theorem 1. *Follower-based influence maximization problem is NP-hard.*

Proof. Supposing we have the updated thresholds of each user on target item in hand, Follower-based Influence Maximization problem is simplified to the classical influence maximization under Linear Threshold model. Therefore, the theorem holds. □

Theorem 2. *For MLTF Model, the objective function $\sigma(\mathcal{S}_{i^*}|\mathbb{D})$ satisfies monotonicity and submodularity.*

Proof. (1) **Monotonicity.** After the diffusion process of the existing items in $\mathcal{I} - i^*$, for item i^*, $\forall\, \mathcal{S}_{i^*} \subseteq \mathcal{V}$, adding a new seed node $u \in \mathcal{V} \setminus \mathcal{S}_{i^*}$ into $\mathcal{S}_{i^*}$, will not cause the influence spread decrease, i.e., $\mathcal{I}(\mathcal{S}_{i^*} \cup \{u\}|\mathbb{D}) \geq \mathcal{I}(\mathcal{S}_{i^*}|\mathbb{D})$.

(2) **Submodularity.** After the diffusion process of the existing items in $\mathcal{I} - i^*$, the thresholds of item i^* is updated. For $\mathcal{R} \subseteq \mathcal{T} \subseteq \mathcal{V}$ and $v \in \mathcal{V} \setminus \mathcal{T}$, it is easy to verify $\mathcal{I}(\mathcal{R} \cup \{v\}|\mathbb{D}) - \mathcal{I}(\mathcal{R}|\mathbb{D}) \geq \mathcal{I}(\mathcal{T} \cup \{v\}|\mathbb{D}) - \mathcal{I}(\mathcal{T}|\mathbb{D})$ with live-edge path following [35]. Therefore, the theorem holds. □

We have proved that objective function $\sigma(\mathcal{S}_{i^*}|\mathbb{D})$ is monotone and submodular, and thus we propose a greedy algorithm, RNVGA, to compute the top-K influential nodes for the target item. The proposed algorithm is outlined in Algorithm 2,

Algorithm 2 RNVGA Algorithm.

Input: social network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, seed set size K, the updated thresholds of each user on target item, $\boldsymbol{\theta}'^{i^*}$
Output: Seed node set $\mathcal{S}_{i^*}$.

```
for j = 1...K do
    u* ← argmax_{u∈V} I(S_{i*} ∪ u|D) − I(S_{i*}|D);
    S_{i*} = S_{i*} ∪ u*;
end for
return S_{i*};
```

and it has an approximation ratio of $(1 - \frac{1}{e})$ from the result of [35], where e is the base of the natural logarithm. With the updated thresholds, the algorithm selects K influential nodes greedily that maximize the marginal expected spread, and finally returns the seed node set $\mathcal{S}_{i^*}$.

5. Experimental setup

In this section, we detail our experimental setup, including research questions, datasets, baselines, evaluation metrics and experimental settings.

5.1. Research questions

We seek to answer the following research questions that guide the remainder of the paper:

(RQ1) Can our RNVGA algorithm predict accurately compared with the state-of-art models?
(RQ2) Does our proposed RNVGA algorithm outperform the state-of-the-art competitive and complementary influence maximization algorithms w.r.t. overall performance?
(RQ3) How do the hyper-parameters (neg-ratio and embedding size D) of RNVGA affect the performance of influence spread?
(RQ4) Does our proposed RNVGA algorithm have scalability when increasing the network size?
(RQ5) Does the RNV model enhance the performance of expected spread?

5.2. Datasets

We use three real-world and publicly available datasets: Yelp challenge 2013[1], Digg[2] and Flixster[3] data sets, each of which contains a social graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a log of past purchasing $\mathbb{D}$ and items features. Specifically, Yelp data set contains 70,816 users, 622,873 links, 15,584 items, Digg contains 30,358 users, 99,846 directed links, 7100 items, and Flixster contains 11,258 users, 84,606 directed links, 14,296 items.

For Yelp dataset challenge 2013, the relationship network composed of links is undirected graph, so we construct the directed relationship graph by turning all undirected edges into double-directed edges. We do not have data directly reflecting

[1] https://www.yelp.com/.
[2] https://www.digg.com/.
[3] https://www.flixster.com/.

purchase time, so we assume the time at which the user reviews the item is the time when the item is purchased. Besides, we disregard the unfrequent behaviors of repeated review for the same item.

5.3. Baselines

We make comparisons between our RNVGA algorithm and the following state-of-the-art baseline algorithms:

IMM [16]: It is an influence maximization algorithm that uses the classical statistical tool martingale, which can provides accurate results with approximate guarantee.

Clash [5]: This model allows for the competition as well as the cooperation of different contagions in information diffusion.

CorrelatedC [6]: It uses the Hawkes process to model spread of cascades in social networks full with competitive and complementary entities.

CCDLT [7]: It utilizes a greedy algorithm to solve the problem of multiple-entities influence maximization, assigning the correlations of entities randomly.

5.4. Evaluation metrics and experimental settings

To evaluate our models performance on prediction accuracy, we use three common evaluation metrics: **precision, recall** and **F1-score**.

Precision: It measures the ratio of the truly activated nodes in all the activated nodes that are searched by the algorithm. The calculation formula is as follows.

$$Precision = \frac{TP}{TP + FP} \tag{16}$$

where TP denotes the number of the actually activated nodes in the activated nodes that are detected by the algorithm, and FP denotes the number of the nodes that are not actually activated in the activated nodes that are detected by the algorithm.

Recall: It indicates how many of the nodes that are actually activated are detected by the algorithm. The calculation formula is as follows.

$$Recall = \frac{TP}{TP + FN} \tag{17}$$

where FN denotes the number of the actually activated nodes that are not detected by the algorithm.

F1-score: It is the harmonic mean of precision and recall. The calculation formula is as follows.

$$F1 - score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{18}$$

To evaluate the overall performance of our algorithm, we set **expected spread** and **running time** as evaluation metrics, which are two extensively adopted metrics on influence maximization problem [15,16].

To evaluate the effectiveness of the proposed RNV model, we obtain the purchase actions in the following cases as the ground truth: social influence - a user buys an item after at least one of his friends bought the item. The considered period of social influence is set according to [36].

We use 90% purchasing log as train set and 5% purchasing log as validation set, and set 5% purchasing log and all links in friend relationship graph as test set. For the baselines, the optimal parameters are set according to the literatures. For our RNV, we set negative sampling ratio, i.e., $Neg - ratio = 5,$ and set the size of embedding dimension, i.e., $D = 50$. The minibatch is set to 128. The prior network is two latent layers with Relu activation; the decoder network is two latent layers with Relu activation; the encoder network is two latent layers with Relu activation and the last layer of encoder network is sigmod activation; The two embedding networks are both two latent layers with Relu activation; the RNN is realized as a GRU.

6. Results and analysis

In this section, we answer the research questions listed in Section 7.1, analyze the experimental results.

6.1. Accuracy of prediction

RQ1: Table 1 presents the **precision, recall** and **F1-Score** of our RNVGA algorithm and the baselines on Yelp data set challenge 2013, Digg and Flixster. From Table 1, we can see that all the algorithms predict most accurately on Yelp data set challenge 2013 due to the large size of this dataset. RNVGA significantly outperforms all the other algorithms on all three datasets because it takes into account not only the dependency between the present-time-step activation and the previous sequence of all activation products, but also the correlation among the probabilities of each product activation in a previous activation sequence. However, the baseline algorithms either assume that cascades of multiple-entity propagation

Table 1
Comparison of precision, recall and F1-Score for all models on three datasets.

		Precision	Recall	F1-Score
Digg	IMM	0.1905	0.5850	0.2874
	Clash	0.2053	0.7304	0.3205
	CCDLT	0.2194	0.7516	0.3367
	CorrelatedC	0.2332	0.7701	0.3580
	RNVGA	**0.5215**	**0.7816**	**0.6256**
Flixster	IMM	0.2160	0.4604	0.2940
	Clash	0.2255	0.5185	0.3138
	CCDLT	0.2803	0.5592	0.3734
	CorrelatedC	0.2902	0.5761	0.3860
	RNVGA	**0.5743**	**0.7746**	**0.6596**
Yelp 2013	IMM	0.4205	0.5015	0.4574
	Clash	0.4403	0.5204	0.4770
	CCDLT	0.4621	0.5316	0.4944
	CorrelatedC	0.4732	0.6017	0.5298
	RNVGA	**0.8964**	**0.6917**	**0.7809**

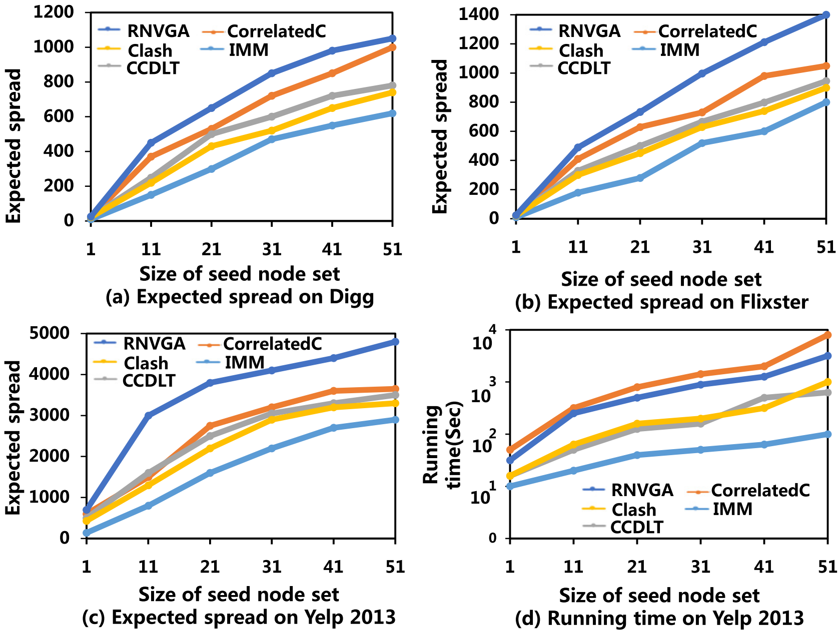


Fig. 4. Expected spread and running time with different size of seed node set K.

are independent which is usually not true in the real world, or do not find a more efficient framework to characterize the non-linear and undeterministic relationships among propagation cascades. The performance of all algorithms on Flixster is higher than Digg due to that Flixster contains more items and the its average degree of network is bigger. Thus, algorithms can capture more influence propagation information on Flixster than Digg, leading to better performance. We also note that IMM algorithm has lowest performance on all three datasets due to IMM being designed to run under the condition of single propagation entity, and thus it exhibits its inadaptability and inefficiency in the network of multiple propagation entities.

6.2. Algorithm overall performance

RQ2: We evaluate the proposed RNVGA, by varying the number of seeds K from 1 to 50, in comparison with the baseline algorithms. Fig. 4(a)–(c) report the expected spread of all algorithms. The following findings can be observed from Fig. 4(a)-(c): (1) Our RNVGA achieved all the best performance in terms of three datasets and expected spread, which confirms the effectiveness of our RNVGA to the multiple-entity propagation task. (2) All competitive and complementary influence maximization algorithms outperform single-entity influence maximization algorithm, i.e. IMM, in terms of three datasets and expected spread, which indicates that under the condition of multiple propagation entities, taking into account the relationships (competition and complementation) among entities does help to capture a more complex form of correlation between

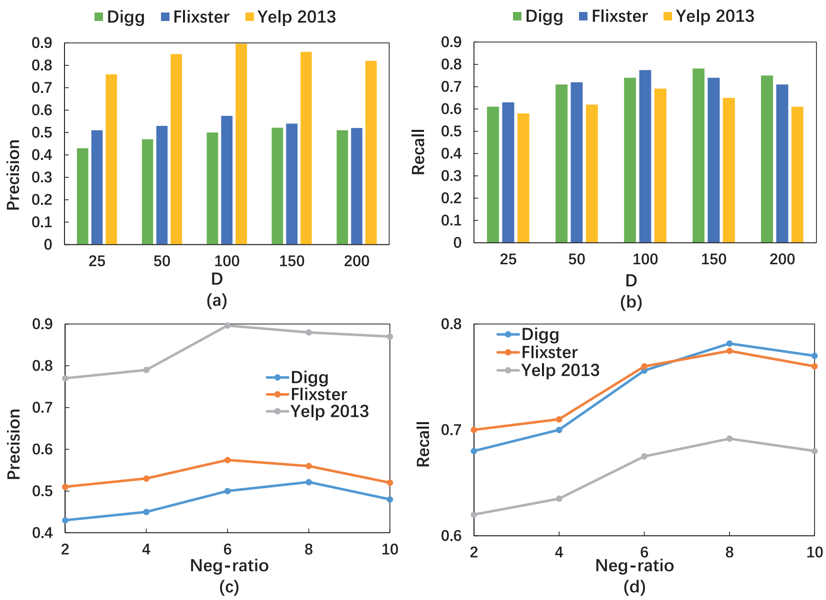


Fig. 5. Performance on Precision and Recall with different embedding sizes D (subfigures (a) and (b)) and negative sampling ratio Neg-ratio (subfigures (c) and (d)), respectively, on the three datasets, Digg, Fixster and Yelp 2013.

different entities and discover how these correlations affect the propagation of a given entity. (3) We also observe that the Clash algorithm obtains the lowest performance of expected spread among all competitive and complementary influence maximization algorithms, which demonstrates that considering the dependencies between cascades in multiple-entity propagation does help to characterize the interaction of different propagation entities. (4) The performances of RNVGA and CorrelatedC outperform that of CCDLT, which indicates that not assigning the correlations of entities randomly but modeling the relationship between the given entity and previous activation sequence can help better capture the different degrees of competitiveness or complementarity among the entities. (5) Our RNVGA outperforms CorrelatedC, which demonstrates that our way of incorporating deep learning can efficiently capture non-linear and subtle dependencies among complex data.

Fig. 4 (d) reports the running time of all algorithms. Due to space constraints, we only report the results of running time on Yelp 2013. Similar trends are found upon Digg and Flixster. We can see that our RNVGA obtains a better solution by spending more time. Clash and CCDLT are the best two algorithms (excluding IMM) w.r.t. running time, because when modeling, they simplify the propagation process to some extend by ways of assigning the correlations of entities randomly or assuming independencies between cascades.

6.3. Sensitivity analysis

RQ3: In this part, we study the impact of the hyper-parameters on our RNVGA. To evaluate the effects of the dimension of latent space, we compare the performance of Precision and Recall by setting the size of embeddings' dimension, D, as 25, 50, 100, 150 and 200, with fixing Neg-ratio=5 on the three datasets, respectively. The experimental results are reported in Fig. 5(a)–(b). It can be observed from Fig. 5(a)–(b) that larger dimension leads to better performance. Specifically, the optimal embedding size of RNVGA for Digg is 150, and for Flixster and Yelp 2013 it is 100.

To understand the impact of the negative sampling ratio (Neg-ratio $\in$ [2, 4, 6, 8, 10]) on RNVGA, we compare the performance of Precision and Recall for different Neg-ratio on the three datasets. Fig. 5(c)–(d) show the performance with different negative sampling ratios. We can observe from Fig. 5(c)–(d) that: (1) In general, sampling more negative samples will lead to better performance. (2) For the three datasets, the optimal ratio for our RNVGA is between 6 and 8, which means we can tune the Neg-ratio to achieve best performance.

6.4. Scalability evaluation

RQ4: We evaluate the scalability of RNVGA and the baseline algorithms by increasing the size of the network, with the seed set size K fixed as 20. Fig. 6(a)–(b) show the results of expected spread and running time, respectively, when varying the network size M from 10K to 50K. We use the directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ in Yelp 2013 to generate subnetworks of different sizes: we first randomly select a node, then starting from this node execute the bread-first traversal.

Fig. 6 (a) reports the expected spread of all algorithms appears relatively stable across the subnetworks of different sizes. It is consistent with the results of experiments w.r.t. **RQ2** that the expected spread of RNVGA outperforms the baselines

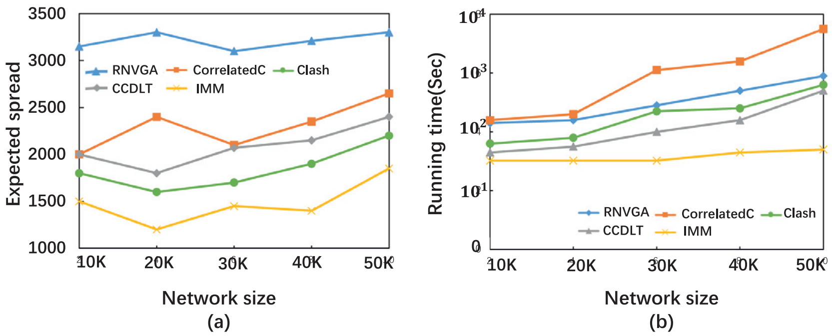


Fig. 6. scalability on Yelp 2013.

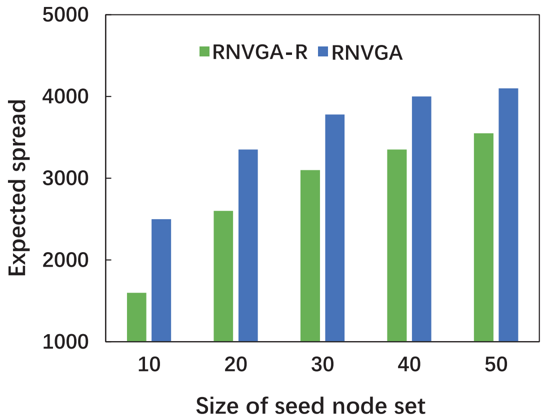


Fig. 7. Performance comparison of RNVGA and RNVGA-R on Yelp 2013.

significantly. Fig. 6(b) reports that the running time of all the algorithms increase when the network size enlarging. It is remarkable in Fig. 6(b), RNVGA scales better than CorrelatedC.

6.5. Contribution of RNV model

Finally, we turn to answer **RQ5** for understanding the effect of our RNV model. We consider the variant of our RNVGA, RNVGA-R, which randomly sets the probability that a given entity is activated under the condition of the item activation sequence. This strategy of random assigning is similar to that in literature [7], which assigns correlations of entities randomly. In Fig. 7, we can find RNVGA significantly outperforms RNVGA-R on the performance of expected spread for the dataset Yelp 2013, setting the size of seed set as 10, 20, 30, 40 and 50. This demonstrates effectively modeling the dependencies between the given entity and previous activation sequence through our RNV model can get better performance than randomly assigning the correlations between them.

7. Conclusions

We have studied the competitive and complementary Linear Threshold Model and influence maximization problem from the perspective of the follower, who intends to promote new product into the network that is occupied by multiple products. To tackle the problem, we have proposed a threshold updating, a recurrent neural variational model (RNV) and a follower-based algorithm (RNVGA). RNV dynamically tracks entity correlations and cascade correlations through a deep generative model and recurrent neural variational inference. RNVGA is greedy-based and efficiently mines seed node set for the target product. Experimental results validate the effectiveness of the proposed model and algorithm.

There are several directions for future research. First, since our method is currently only based on the linear threshold propagation model, one issue that arises with our model is how to apply it to the independent cascade propagation model. Second, we only adopted the offline evaluation method in this article, we'd like to deploy our model into a real system platform and to evaluate our model in the online situation.

Declaration of Competing Interest

We wish to draw the attention of the Editor to the following facts which may be considered as potential conflicts of interest and to significant financial contributions to this work.

We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in this manuscript has been approved by all of us.

We confirm that we have given due consideration to the protection of intellectual property association with this work and that there are no impediments to publication, including the timing of publication, with respect to intellectual property. In so doing we confirm that we have followed the regulations of our institutions concerning intellectual property.

We understand that the Corresponding Author is the sole contact for the Editorial process (including Editorial Manager and direct communication with the office). He is responsible for communicating with the other authors about progress, submissions of revisions and final approval of proofs. We confirm that we have provided a current, correct email address which is accessible by the Corresponding Author and which has been configured to accept email from INFORMATION SCIENCES.

CRediT authorship contribution statement

Huimin Huang: Conceptualization, Methodology, Software, Data curation, Writing - original draft. **Zaiqiao Meng:** Validation, Formal analysis, Writing - review & editing, Supervision. **Shangsong Liang:** Visualization, Writing - review & editing.

References

[1] H. Huang, H. Shen, Z. Meng, Item diversified recommendation based on influence diffusion, Inform. Process. Manage. 56 (3) (2019) 939–954.
[2] H. Hung, H. Shuai, D. Yang, L. Huang, W. Lee, J. Pei, M. Chen, When social influence meets item inference, in: SIGKDD, ACM, 2016, pp. 915–924.
[3] H. Li, S.S. Bhowmick, J. Cui, Y. Gao, J. Ma, GetReal: towards realistic selection of influence maximization strategies in competitive networks, in: SIGMOD, ACM, 2015, pp. 1525–1537.
[4] Y. Zhu, D. Li, Host profit maximization for competitive viral marketing in billion-scale networks, in: INFOCOM, IEEE, 2018, pp. 1160–1168.
[5] S.A. Myers, J. Leskovec, Clash of the contagions: cooperation and competition in information diffusion, in: ICDM, IEEE, 2012, pp. 539–548.
[6] A. Zarezade, A. Khodadadi, M. Farajtabar, H.R. Rabiee, H. Zha, Correlated cascades: compete or cooperate., in: AAAI, 2017, pp. 238–244.
[7] I. Litou, V. Kalogeraki, D. Gunopulos, Influence maximization in a many cascades world, in: ICDCS, IEEE, 2017, pp. 911–921.
[8] F. Wu, B.A. Huberman, Novelty and collective attention, Proc. Natl. Acad. Sci. U.S.A. 104 (45) (2007) 17599–17601.
[9] W. Lu, W. Chen, L.V. Lakshmanan, From Competition to Complementarity: Comparative Influence Diffusion and Maximization, vol. 9, VLDB Endow., 2015, pp. 60–71.
[10] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J.M. Vanbriesen, N.S. Glance, Cost-effective outbreak detection in networks, in: SIGKDD, ACM, 2007, pp. 420–429.
[11] W. Chen, Y. Wang, S. Yang, Efficient influence maximization in social networks, in: SIGKDD, ACM, 2009, pp. 199–208.
[12] S. Cheng, H. Shen, J. Huang, G. Zhang, X. Cheng, StaticGreedy: solving the scalability-accuracy dilemma in influence maximization, in: CIKM, ACM, 2013, pp. 509–518.
[13] S. Galhotra, A. Arora, S. Roy, Holistic influence maximization: combining scalability and efficiency with opinion-aware models, in: SIGMOD, ACM, 2016, pp. 743–758.
[14] W. Chen, C. Wang, Y. Wang, Scalable influence maximization for prevalent viral marketing in large-scale social networks, in: SIGKDD, ACM, 2010, pp. 1029–1038.
[15] Y. Tang, X. Xiao, Y. Shi, Influence maximization: near-optimal time complexity meets practical efficiency, in: SIGMOD, ACM, 2014, pp. 75–86.
[16] Y. Tang, Y. Shi, X. Xiao, Influence maximization in near-linear time: a martingale approach, in: SIGMOD, ACM, 2015, pp. 1539–1554.
[17] H. Huang, H. Shen, Z. Meng, H. Chang, H. He, Community-based influence maximization for viral marketing, Appl. Intell. (2019) 1–14.
[18] F. Morone, H.A. Makse, Influence maximization in complex networks through optimal percolation, Nature 524 (7563) (2015) 65–68.
[19] A. Goyal, F. Bonchi, L.V.S. Lakshmanan, A data-based approach to social influence maximization, VLDB Endow. 5 (1) (2011) 73–84.
[20] J. Qiu, J. Tang, H. Ma, Y. Dong, K. Wang, J. Tang, DeepInf: social influence prediction with deep learning, in: SIGKDD, ACM, 2018, pp. 2110–2119.
[21] T. Carnes, C. Nagarajan, S.M. Wild, A. Van Zuylen, Maximizing influence in a competitive social network: a follower's perspective, in: ICEC, ACM, 2007, pp. 351–360.
[22] W. Lu, F. Bonchi, A. Goyal, L.V. Lakshmanan, The bang for the buck: fair competitive viral marketing from the host perspective, in: SIGKDD, ACM, 2013, pp. 928–936.
[23] A. Borodin, M. Braverman, B. Lucier, Strategyproof mechanisms for competitive influence in networks, in: WWW, ACM, 2013, pp. 141–151.
[24] A. Khan, B. Zehnder, D. Kossmann, Revenue maximization by viral marketing: a social network host's perspective, in: ICDE, IEEE, 2016, pp. 37–48.
[25] H.-C. Ou, C.-K. Chou, M.-S. Chen, Influence maximization for complementary goods: why parties fail to cooperate? in: CIKM, ACM, 2016, pp. 1713–1722.
[26] D.P. Kingma, M. Welling, Auto-encoding variational bayes, ICLR, 2014.
[27] J. Schmidhuber, On learning to think: algorithmic information theory for novel combinations of reinforcement learning controllers and recurrent neural world models, arXiv:1511.09249 (2015).
[28] H. Zhang, Z. Wang, D. Liu, Global asymptotic stability of recurrent neural networks with multiple time-varying delays, IEEE Trans. Neural Netw. 19 (5) (2008) 855–873.
[29] H.G. Zhang, Z.S. Wang, New delay-dependent criterion for the stability of recurrent neural networks with time-varying delay, Sci. China Ser. F 52 (6) (2009) 942–948.
[30] H. Zhang, Z. Liu, G. Huang, Z. Wang, Novel weighting-delay-based stability criteria for recurrent neural networks with time-varying delay, IEEE Trans. Neural Netw. 21 (1) (2010) 91–106.
[31] J. Bayer, C. Osendorfer, Learning stochastic recurrent networks, arXiv:1411.7610 (2014).
[32] N. Boulangerlewandowski, Y. Bengio, P. Vincent, Modeling temporal dependencies in high-dimensional sequences: application to polyphonic music generation and transcription, in: ICML, ACM, 2012, pp. 1881–1888.
[33] O. Fabius, J.R. Van Amersfoort, D.P. Kingma, Variational recurrent auto-encoders., arXiv:1412.6581 (2014).
[34] K. Gregor, I. Danihelka, A. Graves, D.J. Rezende, D. Wierstra, Draw: a recurrent neural network for image generation, in: ICML, ACM, 2015, pp. 1462–1471.
[35] D. Kempe, J.M. Kleinberg, E. Tardos, Maximizing the spread of influence through a social network, in: SIGKDD, ACM, 2003, pp. 137–146.
[36] J. Yang, J. Leskovec, Modeling information diffusion in implicit networks, ICDM, IEEE, 2010.