Modeling Information Diffusion With Sequential Interactive Hypergraphs

Hai Jin, *Fellow, IEEE*, Yao Wu, Hong Huang, Yu Song, Haohui Wei, and Xuanhua Shi

Abstract—In online social networks, numerous users generate and distribute tremendous content simultaneously, and how information spreads in online society has attracted critical attention. Recent studies introduce deep learning to help mine the complex diffusion pattern and forecast the diffusion trend. However, there are two limitations of the state-of-art methods. On the one hand, existing models only focus on the internal influence within the diffusion flow but ignore the external influence from the dissemination of other contents. On the other hand, the dynamics of user interest are barely considered, while the changes of user preference for contents also have a considerable impact on future diffusion. To address these issues, we introduce the hypergraph structure and a sequential framework to model complex interactions in social networks. Then, we propose dual-channel hypergraph neural networks to tackle the diffusion prediction problem, denoted by HyperINF. Specifically, in the user channel, we build sequential user interactive hypergraphs and learn the dynamic user representation, and in the diffusion channel, we construct a diffusion interactive graph to capture the cross-diffusion relation. At last, we consider the social relation to help make the prediction. Experimental results on three datasets suggest the effectiveness and practicability of the proposed framework.

Index Terms—Social networks, information diffusion prediction, hypergraph

1 Introduction

The emergence of social networks enriches the real-time communication of individuals, and a large number of contents are spreading in the social network. Hence, understanding information diffusion is critical for social marketing and even misinformation recognition. Researchers explore the factors of information diffusion and propose various models to extract the diffusion patterns for future prediction.

Previous studies have shown that the factors of information diffusion are diverse, such as topics [1], social relationship [2], and time sensitivity [3]. The influence of different factors is mixed and complex. Recent models settle the diffusion prediction problem with the help of deep learning techniques. Instead of manually extracting the features, representation learning is widely used in diffusion models [4], [5], [6], [7]. Furthermore, researchers get to adopt *Recurrent Neural Networks* (RNNs) to model the diffusion cascade and improve the prediction accuracy. For example, some models [8] directly adopt RNNs to process the diffusion sequence in time order, while other works incorporate structural features of social networks by innovating basic RNNs [9] or adding customized structural module[10], [11].

Even though the state-of-art models achieve promising results in diffusion prediction, these models have two limitations. First, the external influence from the dissemination of other contents is ignored. The existing models only focus on the internal influence within the diffusion sequence, such as temporal and social relations. As shown in Fig. 1, multiple information diffusion flows coexist in the social network. There are complex interactions not only within the diffusion but also cross various contents dissemination. For example, in the dissemination of p_1 and p_2, it is observed that the similar diffusion pattern occurs among u_3, u_2, and u_4 at t_5. Meanwhile, u_5 retweets p_2. In this case, if we utilize previous models to predict who will retweet p_1 at t_6, the most likely user is u_5. While it has not reckoned with the spreading of p_4 at the same time, the fact remains that u_5 retweets p_4 not p_1 at t_6. It shows that the simultaneous diffusion flows may interfere with each other, and the external influence from the dissemination of other posts needs to be considered. So it is necessary to model the diffusion by considering the complex interactions in the global view.

Second, the dynamics of user interest are barely considered in existing methods. In the social network, the interests of users are changing over time, and the diffusion patterns may also change. As Fig. 1 shows, p_3 is retweeted by u_5 at t_1 and u_6 at t_3, while p_4 is retweeted by u_5 at t_5 and u_6 at t_6. It is also observed that u_3 retweets p_4 at t_4. With a similar pattern of p_3 and p_4, it is more likely to predict p_3 will be retweeted by u_3 in the future. But if u_5 and u_6 have changed the preference of contents at t_4, it is of a high probability that the contents of p_3 and p_4 are unrelated. In that case, u_3 is probably not interested in p_3 and unlikely to retweet p_3. Hence, it is significant to consider the dynamics of user interest and adopt dynamic user modeling for more

- *The authors are with the National Engineering Research Center for Big Data Technology and System, Service Computing Technology and System Lab, Cluster and Grid Computing Lab, Huazhong University of Science and Technology, Wuhan, Hubei 430074, China. E-mail: hjin@mail.hust.edu.cn, {yaowu, honghuang, yusonghust, weihh77, xhshi}@hust.edu.cn.*

Manuscript received 25 June 2021; revised 29 Nov. 2021; accepted 13 Jan. 2022. Date of publication 23 Feb. 2022; date of current version 8 Sept. 2022.
This work was supported in part by the National Key Research and Development Program of China under Grant 2020AAA0108501 and in part by the National Natural Science Foundation of China under Grant 62172174.
(Corresponding author: Hong Huang.)
Recommended for acceptance by J. Chen.
Digital Object Identifier no. 10.1109/TSUSC.2022.3152366

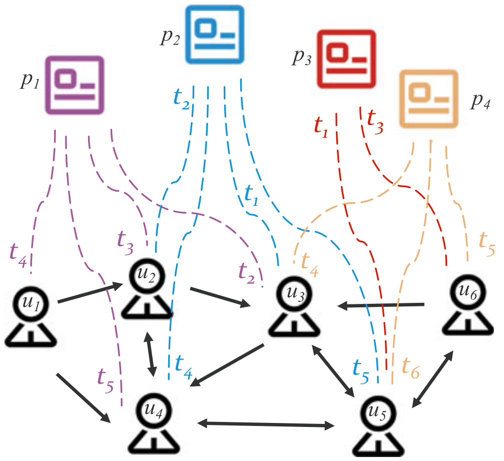
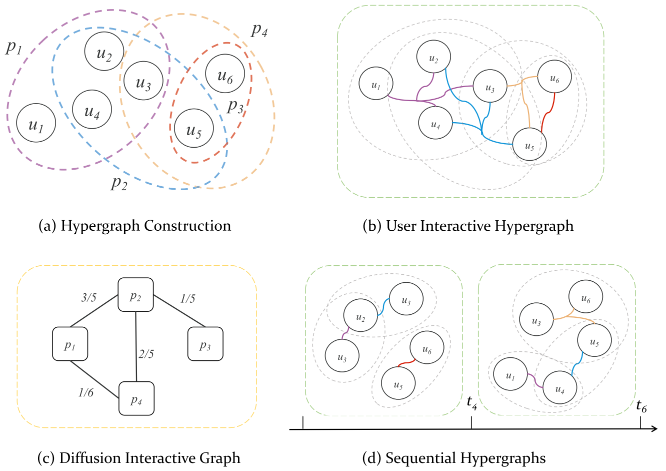


Fig. 1. The information diffusion flows in social networks.



(a) Hypergraph Construction

(b) User Interactive Hypergraph

(c) Diffusion Interactive Graph

(d) Sequential Hypergraphs

Fig. 2. The construction of the interactive graphs.

accurate prediction. Even though some works[12], [13] have paid attention to the dynamics of user preference, they still fail to model the diffusion flow in more realistic scenarios.

To address the above limitations, we introduce a hypergraph structure and dynamic user modeling to model the complex interactions. As shown in Fig. 1, the global interactions include various many-to-many connections between users and posts. Hence, modeling the complex interactions among users coincides with the concept of the hypergraph, where an edge can link any number of nodes [14]. In this way, we formulate each diffusion flow as a hyperedge and construct a user interactive hypergraph. As shown in Figs. 2a and 2b, each hyperedge links users who spread the same post, and the hypergraph preserves the high-order relations among users. Meanwhile, to further capture the connections between different diffusion flows, we construct a diffusion interactive graph. As Fig. 2c shows, each vertex represents a hyperedge in the user interactive hypergraph. The linkage exists if at least one common user shares two related posts, and the corresponding weight indicates the concurrence of users. Furthermore, to model the dynamics of user preference, we construct sequential user interactive hypergraphs by dividing the period into sequential time windows, shown in Fig. 2d.

Based on that, we propose dual-channel hypergraph neural networks to tackle the information diffusion prediction problem, denoted by **HyperINF**. As Fig. 3 shows, in the user channel, we adopt hypergraph convolution networks on each user interactive hypergraph to utilize the high-order relations. Then, we establish a sequential framework with a residual layer between two continuous periods to model the dynamics of user interest. In the diffusion channel, the weighted graph neural networks are applied to the diffusion interactive graph to capture the cross-diffusion relations. Then we design a fusion layer to combine the static user embedding, dynamic user embedding from the user channel, and diffusion embedding from the diffusion channel. Then we consider social relations and adopt diffusion graph neural networks to forecast the subsequent activated users in the near future. The evaluation results suggest the validity of HyperINF. Additionally, we make comparisons of time and memory efficiency between the proposed method with the state-of-art methods in Section 4.6. The result shows that the proposed model is a practical framework running with much less GPU memory consumption and a little more time. In summary, the main contributions of the paper lie in three aspects:

- We study the external influence from the dissemination of other contents and further consider the dynamics of user interest when modeling information diffusion.
- We propose the dual-channel hypergraph neural networks and establish a sequential framework for information diffusion prediction, denoted by *HyperINF*.
- We evaluate HyperINF on three real-world datasets. The experimental results suggest the effectiveness and practicability of the proposed method.

The rest of the paper is organized as follows. The second part gives a brief review of related works. The third section introduces the proposed model in detail. The next section describes the experimental settings and analyzes the evaluation results. The final section summarises the proposed model and includes a discussion of future works.

2 Related Works

2.1 Information Diffusion Prediction

Many studies have investigated information diffusion in online social media and aimed at discovering the latent diffusion patterns for future diffusion prediction. Generally, the information diffusion prediction falls into two categories: macro-prediction and micro-prediction. The macro-prediction seeks to forecast the future popularity while the other focuses on predicting the following activated users at the micro-level.

Recently, researchers have attempted to adopt emerging deep learning techniques and proposed many diffusion models for prediction. Representation learning is widely used to learn the user embedding for further diffusion, such as EmbeddingIC [4], inf2Vec [5], and HID [7]. Some works [6] also consider the content feature and project the users and contents into the same vector space. But these works fail to incorporate the representation learning into the diffusion model for prediction. Thus, there is a large number of diffusion models that build up an end-to-end framework for prediction. DeepCas [15] and DeepHawkes [16] are the earlier RNN-based models to settle the macro-prediction problem and consider the time features at the same time. In addition, [17] introduces the uncertainty in the diffusion process with a variational autoencoder. Moreover, the models in [18], [19] also introduce the content features when modeling the temporal process. Meanwhile, the graph neural networks are utilized in recent studies [20], [21], [22], [23], [24].

For the diffusion prediction at the micro-level, some models [8], [25] consider the temporal characteristics and enrich the basic RNNs-based framework with the attention mechanism. NDM [26] adopts the convolutional neural network and attention mechanism to make a prediction. Other models [9], [10], [27] introduce the structure information in social networks. For example, TopoLSTM [9] improves the basic LSTM with social topology while SNIDSA [10] adds a structure attention model before the recurrent neural networks. Some diffusion models utilize both time and structure features, such as infVAE [28]. Moreover, FOREST [11] proposes a multi-scale diffusion model to solve the micro-prediction problem and further realizes macro-prediction with reinforcement learning.

By contrast, previous studies pay more attention to the internal influence within the diffusion flow and ignore the mutual influence between simultaneous information dissemination. Furthermore, the dynamics of user interest are barely considered. To address these issues, we propose the model HyperINF and mainly focus on the diffusion prediction at the micro-level in the paper.

2.2 Hypergraph Neural Networks

The hypergraph is a generalization of the concept of the graph, where the edge could link even more than two nodes [14]. Therefore, a hypergraph can represent the complex relations among nodes. There are a lot of applications in various fields, such as image recognition [29], sentiment classification [30], and knowledge representation [31]. With the emergence of graph neural networks, an increasing number of scholars get to explore graph neural networks on a hypergraph. Gao *et al.* first propose hypergraph neural networks [32] and consider the dynamic modifications of graph structure in [33]. Besides, HyperGCN [34] extends graph convolutional networks on a hypergraph. The study [35] proposes hypergraph neural networks with the self-attention mechanism. Similarly, the model in [36] leverages an attention module after performing convolution on a hypergraph. Meanwhile, there are many personalized hypergraph neural networks for various tasks. For example, HGC-RNN [37] is proposed to settle the time series prediction problems, and the works [38], [39] apply hypergraph neural networks to recommendation tasks.

In this paper, we attempt to tackle the diffusion prediction problem with hypergraph neural networks. Inspired by previous works, we propose a dual-channel hypergraph convolution module and establish a sequential framework to model the dynamics of user preference for the content on a social network.

3 Methods

In this section, we present the overall framework, as shown in Fig. 3. We first give the problem definition and details of the proposed methods. To model the complex interactions in social networks, we construct the user interactive hypergraph and the diffusion interactive graph. Then we propose sequential hypergraph neural networks with dual-channel, including user channel and diffusion channel. In the user channel, we construct a sequential hypergraphs convolution framework and model the dynamics of user preference. In the diffusion channel, we adopt graph neural networks to extract the cross-diffusion relations. At last, a prediction module introduces the social structure for final diffusion prediction. To facilitate reading, we summarize some symbols in the paper in Table 1.

3.1 Problem Definition

We study the information diffusion on a social network $\mathcal{G} = (\mathcal{U}, \mathcal{E})$, where $\mathcal{U}$ denotes the users and $\mathcal{E}$ represents the following relationships. Generally, we can collect the users' action logs $L = \{(u, p, t) | u \in \mathcal{U}, p \in \mathcal{P}\}$, where (u, p, t) denotes that user u takes action with post $p \in \mathcal{P}$ at time t. In the paper, we focus on the diffusion prediction problem at the micro-level. For instance, we aim to predict who will like, retweet, or comment on the post p on Twitter. Hence, the overall objective is to predict the following activated users, and the diffusion prediction problem can be formulated as follows:

Problem Definition. Given the users' action logs L in $\mathcal{G}$, we can get the diffusion sequences $\mathcal{S} = \{s_j | p_j \in \mathcal{P}\}$. For each $p \in \mathcal{P}$, $s = \{(u_1, t_1), ..., (u_n, t_n)\}$, where $(u_k, p, t_k) \in L$ and $t_{k-1} < t_k$. The objective is to predict the following activated users at t_{n+1} in s. Hence, the prediction problem is formulated as $u_{n+1} = \arg\max_{u \in \mathcal{U}} \mathrm{P}(\mathcal{U} | s, \mathcal{G}, L)$.

3.2 Modeling the Interaction in Social Networks

Generally, numerous contents are spreading in the social network simultaneously. The users take actions on various posts, and complex interactions exist between users and posts. It is observed that the global interactions include various many-to-many relations, as shown in Fig. 1. Each diffusion flow consists of the participation of many users, and a single user could spread multiple posts. In this case, the simple graph structure cannot meet the requirements of modeling many-to-many relations among users triggered by information diffusion. Thus, we introduce the hypergraph structure to model the complex interactions. We formulate each diffusion flow as a hyperedge and construct the user interactive hypergraph, as shown in Fig. 2b. By doing this, the connections between different diffusion flows are implied in the user interactive hypergraph, and yet we still want to build the cross-diffusion relations directly to further address the connections between different diffusion flows. Hence, we construct the diffusion interactive graph, as shown in Fig. 2c. Meanwhile, it is also important to model the dynamics of user preferences as mentioned before. Thus, we attempt to implement dynamic user modeling. First, we model the interactions among users in multiple time windows to extract current user preference. So we get the sequential user interactive hypergraphs in chronological order, as shown in Fig. 2d. Since it is not irrelevant to the user preference in continuous time windows, we design a sequential framework with the residual layer

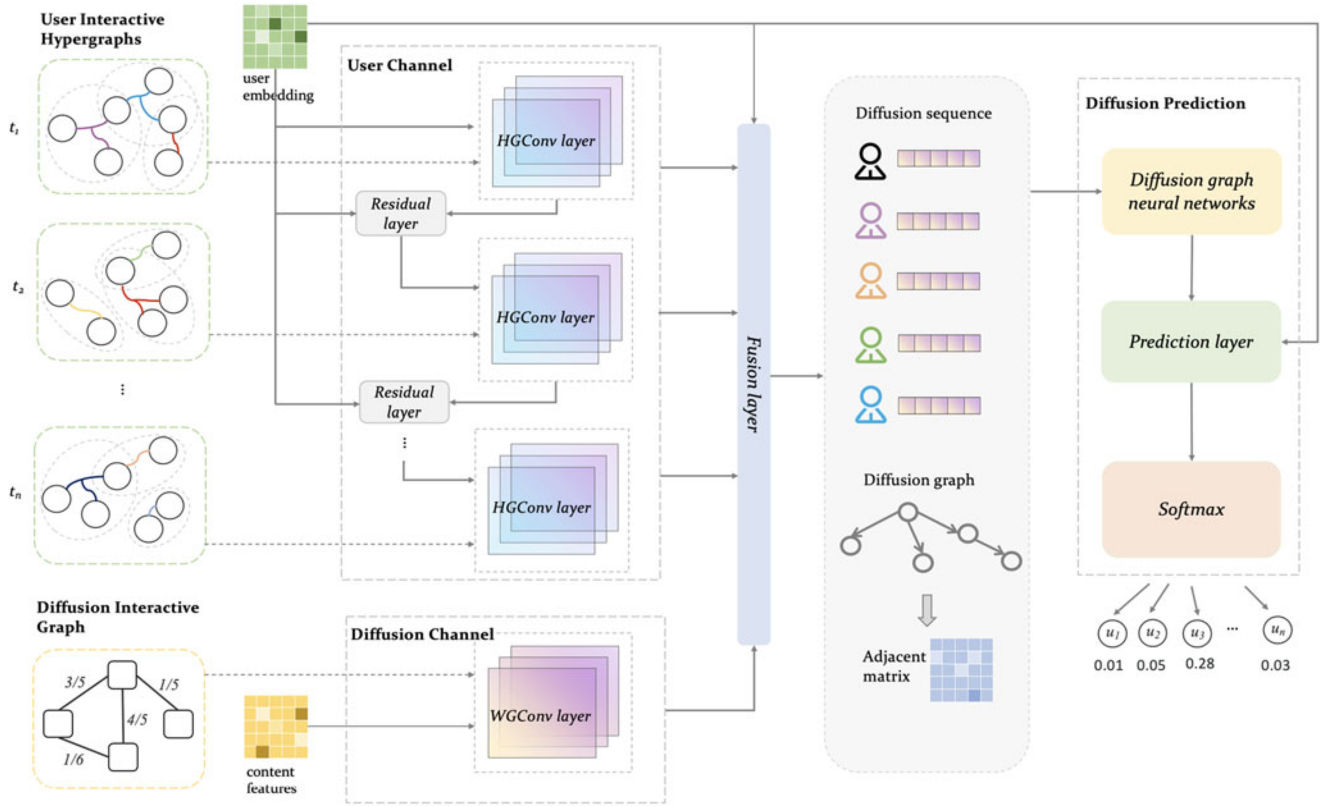


Fig. 3. The overall framework of the proposed model HyperINF.

between two continuous-time windows to model the dynamic changes of the user interest.

We illustrate the construction of the graphs mentioned above in Fig. 2. Specifically, Fig. 2a shows the hypergraph construction, and Fig. 2b presents an example of the user interactive hypergraph, defined by:

Definition 1. User Interactive Hypergraph. *Let* $\mathcal{H} = (U, E, W)$ *denote the user interactive hypergraph, where* $E = \{\epsilon_i | \epsilon_i = \{u_q, u_k, \ldots\}, \text{where } (u_q, p_i, t_q), (u_k, p_i, t_k), \ldots \in L\}$ *and* $W \in \mathbb{R}^{|E|\times|E|}$ *is the weight matrix for hyperedges.*

$W_{\epsilon\epsilon}$ is the positive weight assigned to each edge ϵ, and we define $W_{\epsilon\epsilon} = 1$ in the paper. The connection between users and hyperedges is denoted by $H \in \mathbb{R}^{|U|\times|E|}$. H is the incidence matrix, where $H_{u\epsilon} = 1$ if $u \in \epsilon$, otherwise $H_{u\epsilon} = 0$. Moreover, we can get the degree matrix of users and hyperedges, denoted by $D_u \in \mathbb{N}^{|U|\times|U|}$ and $D_\epsilon \in \mathbb{N}^{|E|\times|E|}$, in which

$$\begin{aligned} d(u) &= \sum_{\epsilon=1}^{|E|} W_{\epsilon\epsilon} H_{u\epsilon} \\ d(\epsilon) &= \sum_{u=1}^{|U|} H_{u\epsilon} \end{aligned} \tag{1}$$

Meanwhile, we notice that the derived line graph of a hypergraph presents connections among hyperedges. In the line graph, each vertex denotes a hyperedge, and two vertices are adjacent if corresponding hyperedges link at least one common node in the hypergraph [40]. For the user interactive hypergraph, the hyperedge presents the diffusion flow. Then the connections among different diffusion flows can be represented by the line graph derived from the user interactive hypergraph. In this way, we construct the diffusion interactive graph to capture the cross-diffusion relations shown in Fig. 2c. For each $p \in \mathcal{P}$, we use p to represent the diffusion of the post p for simplicity. The definition is as follows:

Definition 2. Diffusion Interactive Graph. *Let* $\mathcal{G}^{\mathcal{D}} = \{\mathcal{P}, E^d, W^d\}$ *denote the diffusion interactive graph, where* $\mathcal{P}$ *is the set of diffusion flows.* E^d *is the edge set and* $E^d = \{(p_i, p_j) | (p_i, p_j) = 1 \text{ if } |\epsilon_i \cap \epsilon_j| >= 1\}$. W^d *is the weight matrix and* $w^d_{i,j} = |\epsilon_i \cap \epsilon_j| / |\epsilon_i \cup \epsilon_j|$.

Moreover, the interests of users are not static and will change over time. It is also important to capture the dynamics of user preference for more accurate predictions. To model the users at different times, we construct the sequential user interactive hypergraphs in chronological order, as shown in Fig. 2d. Specifically, we divide the period into continuous-time windows and generate user interactive hypergraphs for each time window. We give the formulation in Definition 3.

Definition 3. Sequential User Interactive Hypergraphs. *Let* $\mathcal{H}^{t_k} = (U^{t_k}, E^{t_k}, W^{t_k})$ *denote the user interactive hypergraph at kth time window, and* $\mathcal{H}^{\mathcal{T}} = \{\mathcal{H}^{t_1}, \mathcal{H}^{t_2}, \ldots, \mathcal{H}^{t_n}\}$ *is the sequential user interactive hypergraphs.*

Based on this, we propose the dual-channel hypergraph neural networks, including the user channel and the diffusion channel. The user channel is to extract the high-order relations between users and learn dynamic user embedding, while the diffusion channel utilizes the connections between diffusion flows to embed the cross-diffusion relations into diffusion embedding.

TABLE 1
The Overall Descriptions of Symbols

Symbol	Description
$\mathcal{G}$	the social network
$\mathcal{U}$	the users in $\mathcal{G}$
$\mathcal{P}$	the posts in $\mathcal{G}$
L	the users' action logs
$\mathcal{S}$	the diffusion sequences of $\mathcal{P}$
$\mathcal{H}$	the user interactive hypergraph
E	the interaction hyperedges in $\mathcal{H}$
W	the weight matrix of hyperedges E
H	the incidence matrix of $\mathcal{H}$
$\mathcal{G}^{\mathcal{D}}$	the diffusion interactive graph
$\mathcal{H}^{t_k}$	the user interactive hypergraph at kth time window
$\mathcal{H}^{\mathcal{T}}$	the sequential user interactive hypergraphs
$\mathcal{G}^{p}$	the diffusion graph of the post p

3.3 The User Channel

In the user channel, the objective is to utilize the high-order relations in user interactive hypergraph and learn robust users' representation for the prediction task. Hence, we expect the representations of users, who share more common posts, to be closer to each other. Recently, the study of *graph neural networks* (GNNs) on the hypergraph has obtained promising results, such as [32], [36]. The hypergraph convolution networks update the hidden state of the user by aggregating the related hyperedges' features, and the hyperedges' features are generated by gathering the linked users' features in a previous step. Similarly, if we apply the convolution operation on user interactive hypergraph, the user could receive more information from neighborhoods linked by more common hyperedges. Since sequential user interactive hypergraphs are constructed in continuous time windows, we apply hypergraph convolution networks on each hypergraph.

Let $\mathbf{X}$ denote the static user embedding matrix and $\mathbf{X}^{t_k,0} = [\mathbf{x}_1^{t_k,0}, ..., \mathbf{x}_{|\mathcal{U}^{t_k}|}^{t_k,0}]$ denote the initial user embedding matrix at kth time window. Then the updated users' representations $\mathbf{X}^{t_k,1}$ are derived as

$$\mathbf{X}^{t_k,1} = f(\mathbf{X}^{t_k,0}, H^{t_k}, W^{t_k}, \Theta^0) \tag{2}$$

where $f(\cdot)$ denotes the hypergraph convolution operation. H^{t_k} and W^{t_k} are the incidence matrix and the weight matrix of $\mathcal{H}^{t_k}$. Θ^0 is the transform matrix to extract the user feature for propagation in the convolution layer. For each user on $\mathcal{H}^{t_k}$, the updating operation can be formulated by:

$$\mathbf{x}_i^{t_k,1} = \sum_{u=1}^{|U^{t_k}|} \sum_{\epsilon=1}^{|E^{t_k}|} H_{i\epsilon}^{t_k} H_{u\epsilon}^{t_k} W_{\epsilon\epsilon}^{t_k} \mathbf{x}_u^{t_k,0} \Theta^0 \tag{3}$$

For all users of $\mathcal{H}^{t_k}$, we can get the matrix form as follow:

$$\mathbf{X}^{t_k,1} = H^{t_k} W^{t_k} H^{t_k T} \mathbf{X}^{t_k,0} \Theta^0 \tag{4}$$

While stacking l layers in the model, the scale of $\mathbf{X}^{t_k,l}$ would be changed by the computation in Eq.(4), which could make the gradients explode or disappear during the feature propagation. Thus we add the symmetrical normalization to keep the convolution layer numerically stable. Moreover, we add an activation function $\sigma(\cdot)$ before the output. Overall, the hypergraph convolution layer is derived as

$$\mathbf{X}^{t_k,1} = \sigma(D_u^{t_k-1/2} H^{t_k} W^{t_k} D_\epsilon^{t_k-1} H^{t_k T} D_u^{t_k-1/2} \mathbf{X}^{t_k,0} \Theta^0) \tag{5}$$

where $D_u^{t_k}$ and $D_\epsilon^{t_k}$ are the degree matrix of users and hyperedges in $\mathcal{H}^{t_k}$. Then, with lth layers, the user representation is computed by:

$$\begin{aligned} \mathbf{X}^{t_k,l} &= f(\mathbf{X}^{t_k,l-1}, H_k^t, W^{t_k}, \Theta^{l-1}) \\ &= \sigma(D_u^{t_k-1/2} H^{t_k} W^{t_k} D_\epsilon^{t_k-1} H^{t_k T} D_u^{t_k-1/2} \mathbf{X}^{t_k,l-1} \Theta^{l-1}) \end{aligned} \tag{6}$$

Residual Layer. Since the dynamic user embedding is independently learned in each time window, it extracts the user preference in the present time but still fails to model the dynamics of user interest. Hence, we add a residual layer between two continuous-time windows. It is expected to keep the static information of users and model the dynamics of user interest. To do that, the layer is designed as the linear weighted sum of user static embedding and user dynamic embedding from last time windows. Therefore, for each user $u \in U^{t_k}$, the initial user embedding $\mathbf{X}^{t_k,0}$ of kth hypergraph convolution module is derived as

$$\mathbf{x}_u^{t_k,0} = \begin{cases} \mathbf{x_u}, & t = 1 \\ \alpha \mathbf{x}_u^{t<k,l} + (1-\alpha)\mathbf{x}_u, & t > 1 \end{cases} \tag{7}$$

where x_u is the static user embedding and $\mathbf{x}_u^{t<k,l}$ is the latest user embedding for u before kth time windows. If $u \in U^{t_{k-1}}$, $\mathbf{x}_u^{t<k} = \mathbf{x}_u^{t_{k-1}}$; otherwise, $\mathbf{x}_u^{t<k} = \mathbf{x}_u^{t_q}$, where $u \in U^{t_q}$ and $q = \arg\min_q(t_k - t_q)$. The momentum term α controls the proportion of the feature from the last time window. It is noticed that the longer the time interval, the smaller the impact on the current time window. Meanwhile, the residual layer can guarantee that the gradient will not disappear especially when there are many time windows. Because there always exists a constant $1-\alpha$ after taking derivative, the loss can still be effectively propagated back even if the $\frac{\partial \mathcal{L}}{\partial \mathbf{x}}$ is extremely small. In this way, we learn the dynamic user embedding in each time window.

3.4 The Diffusion Channel

In the diffusion channel, we extract the cross-diffusion relations and learn the diffusion embedding based on the diffusion interactive graph. In the graph, the linkage exists if at least one common user share both two posts. The more common users are, the bigger the related edge weight is. Thus, we expect the representations of the diffusion, with the connection of higher weight, to get closer to each other. For this to happen, we adopt the weighted graph neural networks, and the diffusion embedding matrix $\mathbf{X}_d^l$ at lth layer is updated by:

$$\mathbf{X}_d^l = \hat{\mathbf{D}}_d^{-1} \hat{\mathbf{A}}_d \mathbf{X}_d^{l-1} \Theta_d^l \tag{8}$$

where Θ_d^l is the transformation matrix for the lth layer. Let $\hat{\mathbf{A}}_d = \mathbf{A} + \mathbf{I}$ and $\mathbf{I}$ be the identity matrix. $\mathbf{A}_d \in \mathbb{R}^{|\mathcal{P}|\times|\mathcal{P}|}$ is the weight matrix of $\mathcal{G}^{\mathcal{D}}$, where $\mathbf{A}_{d,(i,j)} = w_{i,j}^d$. Thus we can get the diagonal degree matrix $\hat{\mathbf{D}}_d \in \mathbb{R}^{|\mathcal{P}|\times|\mathcal{P}|}$, where $\hat{\mathbf{D}}_{d,(i,i)} = \sum_{j=1}^{|\mathcal{P}|} \hat{\mathbf{A}}_{d,(i,j)}$. In this way, the weighted graph neural networks realize to aggregate information from the neighbors in line with the proportion of edge weight.

Fusion Layer. To integrate the information embedded in the learned embeddings, we combine the static user embedding, dynamic user embedding from the user channel, and diffusion embedding from the diffusion channel for further diffusion prediction. Thus, we propose a fusion operation $g(\cdot)$ to combine the learned embeddings. In the paper, we choose concatenation to generate the embedding for each user. Then, the embedding for user u in the diffusion of p in k-th time window is computed by:

$$\begin{aligned}\mathbf{e}_{p,u} &= g(\mathbf{x}_u, \mathbf{x}_u^{t_k}, \mathbf{x}_{d,p}) \\ &= \mathbf{W}_F[\mathbf{x}_u, \mathbf{x}_u^{t_k}, \mathbf{x}_{d,p}] \end{aligned} \tag{9}$$

where $\mathbf{x}_u$ is the static user embedding and $\mathbf{x}_u^{t_k}$ denotes the latest dynamic user embedding at t_k. $\mathbf{x}_{d,p}$ is the embedding of the diffusion flow with post p. $\mathbf{W}_F$ is the learnable transformation matrix in the fusion layer.

3.5 Information Diffusion Prediction

After getting the user representation through the fusion layer, we next settle the diffusion prediction problem. To predict the future flow of a specific post, we need to consider the internal influence within the diffusion cascade. For example, we can adopt the GRU to model the diffusion flow in time order.

Considering that the social relations between users are important factors of triggering the information diffusion, we further introduce the social structure in diffusion prediction. Thus, we adopt the diffusion graph neural networks with a gated mechanism to introduce the influence of social relations. We formulate each diffusion flow as a diffusion graph. For $p \in \mathcal{P}$, the diffusion sequence $s = \{(u_1, t_1), \ldots, (u_n, t_n)\}$, where $(u_k, p, t_k) \in L$ and $t_{k-1} < t_k$. Let $\mathcal{G}^p = \{\mathcal{U}^p, \mathcal{E}^p\}$ denote the diffusion graph of p, where $\mathcal{U}^p = \{u_k | (u_k, t_k) \in s\}$ and $\mathcal{E}^p = \{(u_a, u_b) | (u_b, u_a) \in \mathcal{E} \text{ and } t_a < t_b,$ where $u_a, u_b \in \mathcal{U}^p\}$.

Similar to the model in [27], we expect to learn each user embedding on the diffusion graph by aggregating the neighborhood users' embedding and the previous related users' embedding alternately. Then, the initial user embedding $\mathbf{x}_{p,u}^{(0)} = \mathbf{e}_{p,u}$, and the update function $\psi(\cdot)$ is derived as

$$\mathbf{x}_{p,u}^{(t)} = \psi(\mathbf{X}_p^{(t-1)}, \mathbf{A}_p) \tag{10}$$

where $\mathbf{x}_{p,u}^{(t)}$ is the hidden representation of user $u \in \mathcal{U}^p$ at time step (t) in the dissemination of p and $\mathbf{X}_p^{(t-1)} = [\mathbf{x}_{p,u_1}^{(t-1)}, \ldots, \mathbf{x}_{p,u_n}^{(t-1)}]$ is the user embedding matrix at previous time step $(t-1)$. Moreover, since the diffusion graph is a directed graph, the adjacent matrix $\mathbf{A}_p$ is the combination of the incoming and outgoing adjacent matrix of diffusion graph $\mathcal{G}^p$, denoted by $\mathbf{A}_p = [\mathbf{A}_p^{\mathbf{In}}, \mathbf{A}_p^{\mathbf{Out}}]$. To be specific, we first gather the features from the spatial neighborhood users in the diffusion graph, denoted by:

$$\mathbf{b}_{p,u}^{(t)} = [\phi(\mathbf{A}_p^{\mathbf{In}} \mathbf{X}_p^{(t-1)}), \phi(\mathbf{A}_p^{\mathbf{Out}} \mathbf{X}_p^{(t-1)})] \tag{11}$$

where $\phi(\cdot)$ is the linear transformation function. Then we use a gated mechanism of GRU to model the diffusion flow in the temporal dimension. The related users' states at previous time steps are aggregated by:

$$\begin{aligned}\mathbf{z}_{p,u}^{(t)} &= \sigma(\mathbf{W}^z \mathbf{b}_{p,u}^{(t)} + \mathbf{U}^z \mathbf{x}_{p,u}^{(t-1)}) \\ \mathbf{r}_{p,u}^{(t)} &= \sigma(\mathbf{W}^r \mathbf{b}_{p,u}^{(t)} + \mathbf{U}^r \mathbf{x}_{p,u}^{(t-1)}) \\ \mathbf{h}_{p,u}^{(t)} &= \tanh(\mathbf{W}^h \mathbf{b}_{p,u}^{(t)} + \mathbf{U}^h (\mathbf{r}_{p,u}^{(t)} \odot \mathbf{x}_{p,u}^{(t-1)})) \\ \mathbf{x}_{p,u}^{(t)} &= (1 - \mathbf{z}_{p,u}^{(t)}) \odot \mathbf{x}_{p,u}^{(t-1)} + \mathbf{z}_{p,u}^{(t)} \odot \mathbf{h}_{p,u}^{(t)} \end{aligned} \tag{12}$$

where $\mathbf{z}_{p,u}^{(t)}$ and $\mathbf{r}_{p,u}^{(t)}$ are update gate and reset gate. $\sigma(\cdot)$ is activation function. Then, we learn the user hidden representation $\mathbf{X}_p = [\mathbf{x}_{p,u_1}, \ldots, \mathbf{x}_{p,u_n}]$ on the diffusion graph $\mathcal{G}^p$.

In some cases, social relations are not available. Thus, we also explore the proposed framework with recurrent neural networks and self-attention mechanisms to make the prediction in Section 4.4.

Diffusion Prediction. For the final prediction, we fuse the final users' embedding to generate the diffusion graph representation. In the paper, we use the soft attention mechanism to calculate the graph representation $\mathbf{e}_p$ as follows:

$$\begin{aligned}\mathbf{e}_p &= \tanh\left(\sum_{u_i \in \mathcal{U}^p} \alpha_i \mathbf{x}_{p,u_i}\right), \\ \alpha_i &= \sum_{j=1}^{|\mathcal{U}^p|} \mathbf{w}_a^T \sigma(\mathbf{W}_1 \mathbf{x}_{p,u_i} + \mathbf{W}_2 \mathbf{x}_{p,u_j} + \mathbf{w}_b) \end{aligned} \tag{13}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are the transformation matrix. $\mathbf{w}_a$ is the vector to learn the attention score of the user in each time step, and $\mathbf{w}_b$ is the bias vector. Since the latest activated users have a significant influence on the future diffusion, we combine the diffusion graph representation and the most recently activated users' representations $\mathbf{x}_{p,u_n}$ for final prediction, denoted by:

$$\hat{\mathbf{y}}_p = softmax(\mathbf{X}[\mathbf{e}_p, \mathbf{x}_{p,u_n}]) \tag{14}$$

where $\hat{\mathbf{y}}_p \in \mathbb{R}^{|\mathcal{U}|}$ is the next activation probability distribution.

Model Optimization. The objective of diffusion prediction is to predict the following activated users in the diffusion flow. Hence, we choose the cross-entropy as the loss functions, derived as

$$\mathcal{L} = -\frac{1}{|\mathcal{P}_t|} \sum_{p \in \mathcal{P}_t} \sum_{i}^{|\mathcal{U}|} \mathbf{y}_{p,i} \log(\hat{\mathbf{y}}_{p,i}) \tag{15}$$

where $\mathcal{P}_t$ is the training samples and $\mathbf{y}_p \in \mathbb{N}^{|\mathcal{U}|}$ is the ground truth. To train the proposed model, we adopt Adam optimizer to find the optimal model in the paper.

4 Evaluations

In this section, we conduct experiments to evaluate the proposed model, denoted by HyperINF. First, we test the model on three datasets and explore the function of different modules. Then we analyze different settings of the hyper-parameters. At last, we compare the time and memory efficiency between HyperINF and baselines.

4.1 Experimental Settings

4.1.1 Datasets

For evaluations, we choose three datasets: Digg [41], Flixster [42], and Weibo [43]. The detailed statistics are shown in Table 2.

TABLE 2
The Statistics of Three Datasets: Digg, Flixster, and Weibo

Dataset	Digg	Flixster	Weibo
Users	14, 381	19, 036	32, 595
Edges	268, 521	275, 186	178, 169
Cascades	3, 553	10, 000	13, 808
Avg. Cascade length	30.0	16.3	8.4

- Digg is a news-sharing platform where users vote and elect the most valuable articles. The dataset includes 3553 reports and related users' activity logs during one month in 2009.
- Flixster is a social movie site where users share or comment on movies. We filter the diffusion cascades where the participants are less than five from the original dataset. For evaluation, we randomly select 10000 movies from 2006 to 2009.
- Weibo is a social networking site based on a microblogging service. We select the cascades where there are at least five participants and more than half of the users take action at least three times. Then we collect the 13, 808 posts and the retweet logs between 2009 to 2012.

4.1.2 Baselines

To compare HyperINF with the state-of-art models, we choose several baselines as follows:

- Deepdiffuse [8] is proposed to model the diffusion with RNNs and predicts the following users with timestamps. In the experiment, the predicted timestamp is neglected.
- TopoLSTM [9] aims to improve the diffusion model by considering the social relations between users and proposes a novel LSTM-based framework with the social structure.
- SNIDSA [10] defines the structure attention module to introduce the structure features and then adopts the RNNs to model the diffusion for prediction.
- FOREST [11] incorporates the user's neighbor information into the GRU-based diffusion model and deals with the macroscopic diffusion prediction problem with reinforcement learning. We only utilize FOREST for diffusion prediction at the micro-level in the experiments.
- infGNN [27] proposes personalized graph neural networks to model the diffusion in spatial and temporal dimensions alternately for prediction.
- DyHGCN [12] builds a heterogeneous graph including following relations and reposting relations between users and considers the evaluation of the heterogeneous graph while modeling the diffusion with an attention mechanism.

4.1.3 Experimental Settings

For Digg and Flixster datasets, we divide the period into 20 time windows and generate user interactive hypergraphs in each time window. For Weibo dataset, we divide the time into 16 intervals. In the experiment, we distribute the dataset into train/valid/test datasets in a ratio of 0.8/0.1/0.1. We tune the hyperparameters for each dataset. Specifically, the settings of batch size are {64, 128, 256, 512}; the embedding size is chosen from {32, 64, 128}; the options of learning rate are {0.005, 0.001, 0.0005}. We stack two layers with 64 or 128 hidden units for all convolution modules. The maximum iterations are 100 to train the models. Besides, the other parameters of the baselines are defined as the default in the original paper. The evaluation metrics are *hit ratio* (Hits) and *mean reciprocal rank* (MRR) with the top-k predictions, denoted by Hits@k and MRR@k. Hits@k is the accuracy among the top-k candidate users predicted by the evaluated model. The higher accuracy means the higher probability of selecting the right activated users among the top-k candidate users. MRR@k is the average of the reciprocal ranks of the next activated users within the top-k candidate users. The larger value indicates the higher ranking of the right activated users in the top-k candidate users. For two metrics, the bigger value addresses the better performance of the evaluated model. The ranges of k are $\{1, 10, 20, 50\}$. Since Hits@ 1 is equal to MRR@1, we add MRR@5 for MRR. We repeat each test multiple times for all models on each dataset and present the average results. All experiments are conducted on a machine with one Tesla V100 GPU.

4.2 Overall Results

We evaluate the proposed model on three datasets and compare it with the baselines mentioned above. The results are presented in Figs. 4 and 5. In general, HyperINF performs the best in most cases. It is observed that there is a significant improvement in the accuracy of the Top-k predictions when k is small, such as Hits@1 and MRR@5 on three datasets. HyperINF excels all baselines when k is smaller than 10. For example about Hits@1, the improvements are nearly double on Digg and more than double on Flixster. The performance on Weibo also improves up by half in most cases. It shows that the accuracy of the prediction is greatly promoted by HyperINF.

It is noticeable that the performance gains on Flixster dataset are more than 10% about Hits and over 20% about MRR. Since Flixster is a social movie site, users gather due to the same interest in films, and the preference is relatively stable in a certain period. Then, the diffusion pattern is highly correlated with the type of movie. Thus, the effect of the cross-diffusion relationships in the diffusion interactive graph is at work, and the proposed method performs much better. For Digg dataset, even if FOREST and DyHGCN perform slightly better than HyperINF in terms of certain indicators, the values of MRR are always the highest. For Weibo, all baselines and HyperINF perform relatively worse than Digg and Flixster. It is noticed that there are more users and the average length of cascades is also shorter in Weibo dataset. In this case, HyperINF still has a relatively higher performance. It shows that modeling the diffusion from a global view can extract more information of users' preferences and offer enough information for prediction even in the early stages of diffusion.

As for the baselines, the models incorporating the social structure, such as FOREST and infGNN, are better than

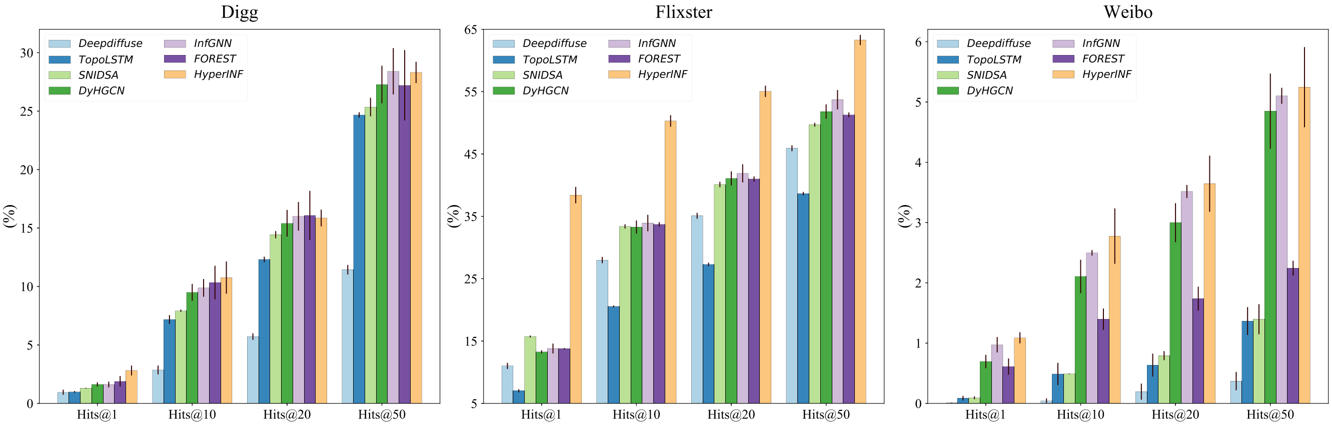


Fig. 4. Experimental results of diffusion prediction on Digg, Flixster, and Weibo datasets about Hits@k (%).

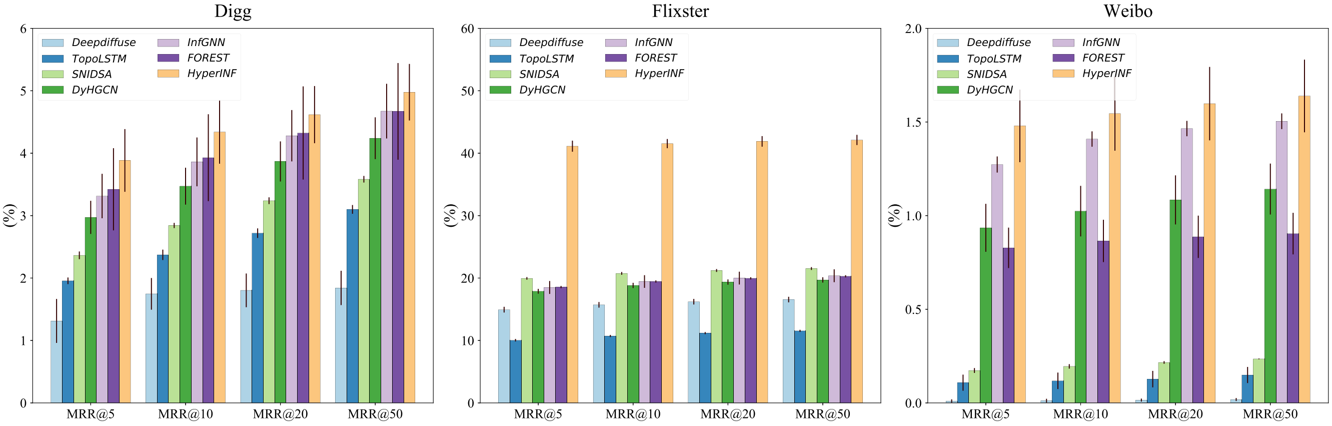


Fig. 5. Experimental results of diffusion prediction on Digg, Flixster, and Weibo datasets about MRR@k (%).

those that sequentially model the diffusion in the temporal dimension, such as Deepdiffuse. In addition, the results illustrate that the dynamic user modeling has a positive effect on the accuracy of diffusion prediction, which is introduced in DyHGCN and HyperINF. By contrast, our model performs better than DyHGCN in most cases. Overall, our proposed method is proved to be effective, and the sequential hypergraph neural networks framework works in the case of modeling the diffusion from a global view.

4.3 The Analysis of the Dual-Channel Module

To further explore the function of two channels in HyperINF, we conduct an ablation study. Therefore, we establish several variants of the proposed model. To be specific, HyperINF-U gets rid of the whole modules in the user channel, while HyperINF-D is the variant without the diffusion channel. HyperINF-S is the model without the sequential framework in the user channel, and the user interactive hypergraph is constructed in the whole period. HyperINF-R only takes off the residual layer between two hypergraph convolution modules in continuous time windows. In addition, we also study the effect of the static user preference for prediction. Thus, we only preserve the dynamic user embedding from the user channel for further prediction in HyperINF-d. The evaluation results on Digg and Flixster datasets are presented in Table 3.

The Analysis of User and Diffusion Channel. HyperINF-U and HyperINF-D take off the user channel and the diffusion channel, respectively. Overall, the two variants suffer performance degradation compared to HyperINF, except for HyperINF-U on Digg about Hits@50. While HyperINF-U still has clear performance degradation on Hit@1 and the performances of HyperINF-U are degraded by around 0.7% about MRR. Besides, even though HyperINF-D has a higher performance than HyperINF-U about Hit@1 on Digg, HyperINF-D still performs worse than HyperINF-U about the remaining metrics. It shows that the diffusion channel and the user channel play different roles in the case. The user channel can help to improve the prediction accuracy with the fewer top-k candidate users and the diffusion channel plays an important role in accurately selecting effective users on the whole. As for the Flixster, HyperINF-D, without the diffusion channel, has relatively worse performance than HyperINF-U. The performance of HyperINF-D drops by over 10% about Hit@1 and 6% about MRR@5, while the performances of HyperINF-U only fall by around 1%. It also indicates that cross-diffusion relations have a more profound impact. Because Flixster is a content-sensitive community and users share the movies depending largely on individual interests. Thus, the post embedding learned by the diffusion channel embeds the feature of the content and plays an important role in the prediction. Furthermore, it is found that HyperINF-S, without the

TABLE 3
The Performance of the Variants of HyperINF on Digg and Flixster Datasets (%)

Metric	Digg				Flixster			
Method	Hits@1	Hits@50	MRR@5	MRR@50	Hits@1	Hits@50	MRR@5	MRR@50
HyperINF-U	1.972	28.451	3.225	4.363	37.603	62.207	40.453	41.519
HyperINF-D	2.254	26.197	3.159	4.260	27.275	59.288	35.327	35.725
HyperINF-S	2.028	27.606	3.235	4.428	37.867	62.750	40.486	41.610
HyperINF-R	2.141	**28.732**	3.370	4.434	38.050	62.820	**41.246**	41.774
HyperINF-d	1.690	27.169	2.897	4.165	25.010	55.460	32.162	32.558
HyperINF	**2.817**	28.310	**3.884**	**4.977**	**38.401**	**63.283**	41.126	**42.124**

TABLE 4
The Performance of HyperINF With LSTM, GRU, and Self-Attention on Digg and Flixster Datasets (%)

Metric	Digg				Flixster			
Method	Hits@1	Hits@50	MRR@5	MRR@50	Hits@1	Hits@50	MRR@5	MRR@50
HyperINF-LSTM	2.394	28.151	3.279	4.289	38.767	63.133	41.441	42.559
HyperINF-GRU	2.535	28.169	3.526	4.560	**39.452**	**63.417**	**42.449**	**42.996**
HyperINF-SF	1.690	23.662	2.350	3.368	37.452	59.529	39.665	40.662
HyperINF	**2.817**	**28.310**	**3.884**	**4.977**	38.401	63.283	41.126	42.124

sequential framework, also performs better than HyperINF-U in some cases. It demonstrates the function of the user channel in HyperINF again. Thus, the proposed dual-channel model benefits from each channel, and the combination has a good effect on improving performance.

The Analysis of Sequential Framework. We further explore the function of the sequential hypergraphs framework in the user channel. As shown in Table 3, it is found that HyperINF-R, without the residual layer, has better performance than HyperINF-S, without the sequential framework. It illustrates that the sequential hypergraph neural networks in the user channel do work. The dynamic user modeling with the sequential framework has a beneficial effect. One interesting finding is that the model without the residual layer performs better than HyperINF in some cases. For Digg, the residual layers lead to a slight performance degradation about the Hits@50. The possible reason could be there is a palpable shift in some users' interests, and the residual layer may introduce nasty noises into next time windows in that case. We also observed that HyperINF performs worse than HyperINF-R in terms of MRR@5 on Flixster. Since Flixster datasets include the cascades during 3 years, the instant interests of some users may not be able to last. Then the noise will be introduced into the following periods and has a bad impact on the performance. While HyperINF-R performs worse than HyperINF in terms of the remaining metrics. Overall, the rank of right prediction by HyperINF is higher than HyperINF-R.

At last, we further remove the static user embedding within the fusion layer in the absence of the diffusion channel. The results show that HyperINF-d suffers degraded performance. So the static user feature embedded in the user representation is crucial and useful. It is essential to preserve the static user characteristics while modeling the dynamics of user preference.

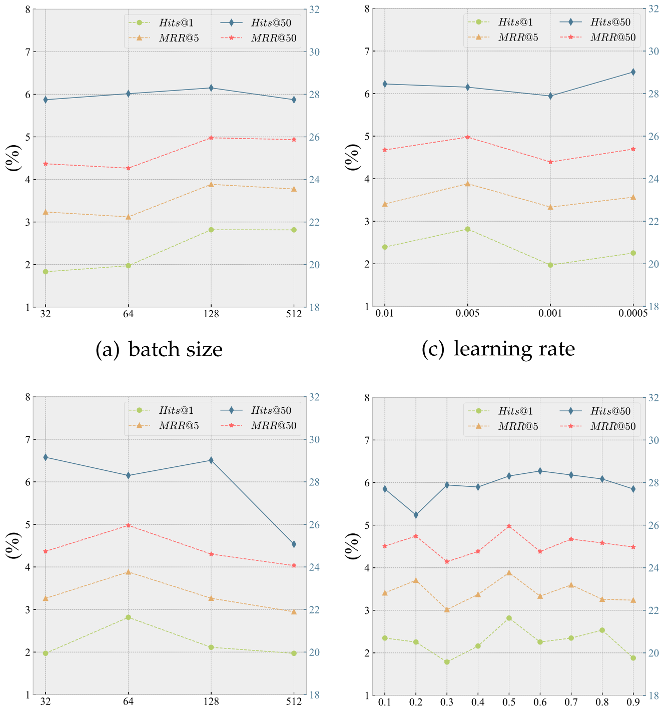
In summary, the high-order relations between users and cross-diffusion relations, extracted by the dual-channel modules, have a beneficial effect in improving the prediction ability of HyperINF. Moreover, the sequential framework is useful and necessary for capturing the dynamics of user interest.

4.4 The Analysis of the Prediction Module

In this part, we want to explore the function of the social structure in diffusion prediction. We test the proposed framework without considering the social relation. Specifically, HyperINF-LSTM and HyperINF-GRU are the variants replacing the diffusion graph neural networks with LSTM and GRU. While HyperINF-SF adopts the self-attention mechanism as the sequence model without the pooling layer and uses the last outputs for further prediction. The results are shown in Table 4. It is observed that HyperINF achieves the best performance on Digg dataset, while HyperINF-GRU and HyperINF-LSTM perform better than HyperINF on Flixster. It shows that introducing social structural information does not always work. For example, the influence of social relations is not significant for Flixster, since it is a content-sensitive community. By contrast, the topic of the content is the dominant factor of information diffusion. Hence, it is not necessary to introduce the structure information in that case. In addition, it is noticed that HyperINF-SF performs the worst by only utilizing the attention layer. In comparison, the RNNs-based diffusion sequence model works in helping capture the diffusion pattern. In general, HyperINF is a flexible and practical framework with different methods to model the diffusion flow. In some cases, it is beneficial to introduce social relations for further prediction. In other cases, we can use RNNs to model diffusion flow in time order.

4.5 Hyper-Parameters Sensitivity Analysis

This section explores the hyper-parameters sensitivity of batch size, embedding size, learning rate, and α in the



(a) batch size

(c) learning rate

(b) embedding size

(d) α of residual layer

Fig. 6. The performance of HyperINF with different batch size, embedding size, learning rate, and α of residual layer on Digg dataset.

residual layer. The basic settings are (batch size, embedding size, learning rate, α) = (128, 64, 0.005, 0.5). We test the sensitivity of each hyper-parameter, with other parameters remaining the same. Specifically, we set batch size as well as embedding size from {32, 64, 128, 512} and learning rate from {0.01, 0.005, 0.001, 0.0005} in turns. The scope of α in the residual layer ranges from 0.1 to 0.9 with the step of 0.1. We present the evaluation results on Digg in Fig. 6.

Batch Size. The proposed model performs better with bigger batch size. When the batch size is too small, such as 32, the model is more likely to be overfitted to the training data.

Embedding Size. Our model has better performance with a smaller embedding size on Digg. HyperINF reaches the best performance about Hit@50 when the embedding size is 32. In comparison, HyperINF performs almost the same or even worse about Hits@1, MRR@5, and MRR@50, with embedding size greater than 64.

Learning Rate. For learning rate, the model performs the best about Hit@50 when the learning rate is small enough, like 0.0005. Similarly, HyperINF has better performances from the metrics of Hits@1, MRR@5, and MRR@50, when the learning rate is 0.005. In this case, the performance only slightly declines about Hits@50.

α of Residual Layer. As α gets larger, the performance of HyperINF shows a trend of rising first and then becoming stable with slight fluctuation on the whole. As shown in Fig. 6d, it performs better with α varying from 0.5 to 0.8, and the best performance is reached when α equals 0.5 on Digg dataset. Based on these observations, we set batch size, embedding size, learning rate, and α as 128, 64, 0.005, and 0.5 for training HyperINF on Digg dataset. A suitable combination of hyper-parameters can help the model have better performance and generalization ability.

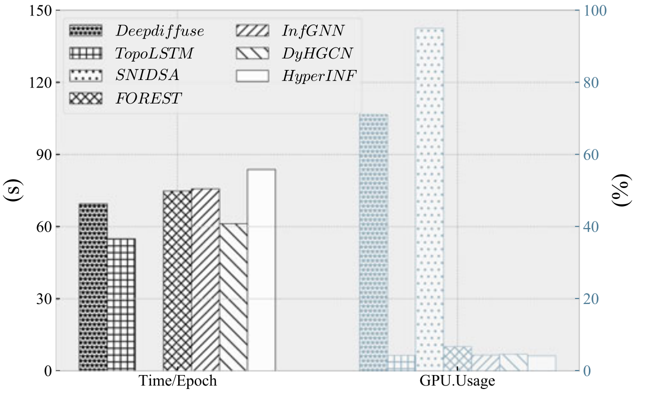


Fig. 7. The comparison of average running time (s) and GPU usage (%) on Digg dataset.

4.6 The Comparison of Time and Memory Efficiency

To study time and memory efficiency, we compare the average running time per epoch and GPU usage of HyperINF with other baselines. To be specific, we test the average running time with a batch size of 128 and embedding size of 64 during 50 epochs. The GPU usage is tested with batch and embedding sizes of 32 and 64 since SNIDSA runs *out of memory* (OOM) with the batch size of 128. The results on Digg dataset are presented in Fig. 7. It is observed that HyperINF costs relatively more time per epoch while the GPU utilization is comparatively low. By contrast, DyHGCN costs less time than the proposed model since the dynamic model is still constructed within the diffusion cascade. Since HyperINF considers the dynamics of user preference in the global view, the interactive graph is relatively bigger than the heterogeneous graph built by DyHGCN. So the training time is a bit longer. It is also noticed that Deepdiffuse and SNIDSA have a higher demand for GPU, and SNIDSA even runs OOM with the batch size larger than 64. The rest models have less GPU usage, and HyperINF consumes the least memory of GPU. Overall, HyperINF costs a little more time with low GPU usage and achieves high performance.

5 Conclusion and Future Works

For information diffusion prediction, we propose dual-channel hypergraph neural networks with a sequential framework, denoted by HyperINF. We construct the user interactive hypergraph and diffusion interactive graph to model the users' interactions in a more realistic and complex scenario. In the user channel, we build sequential user interactive hypergraphs and apply the hypergraph convolution networks on each hypergraph to learn the dynamic user representation. We capture the cross-diffusion relations in the diffusion channel by utilizing the weighted graph neural networks on the diffusion interactive graph. Then, we combine the user static user embedding, dynamic user embedding from the user channel, and diffusion embedding from the diffusion channel with a fusion layer. For further prediction, we introduce the social structure information with the diffusion graph neural networks. In the end, the model outputs the forecast result through a softmax layer. We conduct extensive experiments, and the evaluation results show the effectiveness of our proposed model. In addition, we evaluate the time and memory efficiency. The results show that the proposed model is a practical framework with much less memory consumption and a little more computation time.

In the paper, we mainly focus on diffusion prediction at the micro-level. The extension of the proposed framework into popularity prediction at the macro-level is also worthy of further study. Meanwhile, if content features of the post are available, it can be incorporated as the input of the diffusion channel and further improve the model's generalization ability. Besides, it is the most challenging and significant to study a more explainable diffusion prediction model. Last but not least, since there are many complicated application scenarios for information diffusion models, the exploration of the energy-efficient framework also needs to be considered seriously.

References

[1] O. Tsur and A. Rappoport, "What's in a hashtag? Content based prediction of the spread of ideas in microblogging communities," in *Proc. Int. Conf. Web Search Data Mining*, 2012, pp. 643–652.

[2] B. Suh, L. Hong, P. Pirolli, and E. H. Chi, "Want to be retweeted? Large scale analytics on factors impacting retweet in twitter network," in *Proc. IEEE Int. Conf. Social Comput. Netw.*, 2010, pp. 177–184.

[3] H. Peng, J. Zhu, D. Piao, R. Yan, and Y. Zhang, "Retweet modeling using conditional random fields," in *Proc. Int. Conf. Des. Mater. Workshops*, 2011, pp. 336–343.

[4] S. Bourigault, S. Lamprier, and P. Gallinari, "Representation learning for information diffusion through social networks: An embedded cascade model," in *Proc. Int. Conf. Web Search Data Mining*, 2016, pp. 573–582.

[5] S. Feng, G. Cong, A. Khan, X. Li, Y. Liu, and Y. M. Chee, "Inf2vec: Latent representation model for social influence embedding," in *Proc. Int. Conf. Data Eng.*, 2018, pp. 941–952.

[6] Y. Su, X. Zhang, S. Wang, B. Fang, T. Zhang, and P. S. Yu, "Understanding information diffusion via heterogeneous information network embeddings," in *Proc. Int. Conf. Database Syst. Adv. Appl.*, 2019, pp. 501–516.

[7] H. Zhou, S. Xu, Z. Fu, G. de Melo, Y. Zhang, and M. Kapadia, "HID: Hierarchical multiscale representation learning for information diffusion," in *Proc. Int. Joint Conf. Artif. Intell.*, 2020, pp. 3385–3391.

[8] M. R. Islam, S. Muthiah, B. Adhikari, B. A. Prakash, and N. Ramakrishnan, "DeepDiffuse: Predicting the 'who' and 'when' in cascades," in *Proc. Int. Conf. Des. Mater.*, 2018, pp. 1055–1060.

[9] J. Wang, V. W. Zheng, Z. Liu, and K. C. C. Chang, "Topological recurrent neural network for diffusion prediction," in *Proc. Int. Conf. Des. Mater.*, 2017, pp. 475–484.

[10] Z. Wang, C. Chen, and W. Li, "A sequential neural information diffusion model with structure attention," in *Proc. Conf. Inf. Knowl. Manage.*, 2018, pp. 1795–1798.

[11] C. Yang, J. Tang, M. Sun, G. Cui, and Z. Liu, "Multi-scale information diffusion prediction with reinforced recurrent networks," in *Proc. Int. Joint Conf. Artif. Intell.*, 2019, pp. 4033–4039.

[12] C. Yuan, J. Li, W. Zhou, Y. Lu, X. Zhang, and S. Hu, "DyHGCN: A dynamic heterogeneous graph convolutional network to learn users' dynamic preferences for information diffusion prediction," in *Proc. Eur. Conf. Mach. Learn.*, 2021, pp. 347–363.

[13] H. Miyazawa and T. Murata, "Graph convolutional network with time-based mini-batch for information diffusion prediction," in *Proc. Complex Netw. Appl. IX*, 2021, pp. 53–65.

[14] C. Berge, *Hypergraphs: Combinatorics of Finite Sets*, vol. 45. Amsterdam, The Netherlands: Elsevier, 1984.

[15] C. Li, J. Ma, X. Guo, and Q. Mei, "DeepCas: An end-to-end predictor of information cascades," in *Proc. Int. World Wide Web Conf.*, 2017, pp. 577–586.

[16] Q. Cao, H. Shen, K. Cen, W. Ouyang, and X. Cheng, "DeepHawkes: Bridging the gap between prediction and understanding of information cascades," in *Proc. Conf. Inf. Knowl. Manage.*, 2017, pp. 1149–1158.

[17] F. Zhou, X. Xu, K. Zhang, G. Trajcevski, and T. Zhong, "Variational information diffusion for probabilistic cascades prediction," in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 1618–1627.

[18] D. Liao, J. Xu, G. Li, W. Huang, W. Liu, and J. Li, "Popularity prediction on online articles with deep fusion of temporal process and content features," in *Proc. Conf. Assoc. Advance. Artif. Intell.*, 2019, pp. 200–207.

[19] X. Feng, Q. Zhao, and Z. Liu, "Prediction of information cascades via content and structure proximity preserved graph level embedding," *Informat. Sci.*, vol. 560, pp. 424–440, 2021.

[20] X. Chen, F. Zhou, K. Zhang, G. Trajcevski, T. Zhong, and F. Zhang, "Information diffusion prediction via recurrent cascades convolution," in *Proc. Int. Conf. Data Eng.*, 2019, pp. 770–781.

[21] Z. Huang, Z. Wang, and R. Zhang, "Cascade2vec: Learning dynamic cascade representation by recurrent graph neural networks," *IEEE Access*, vol. 7, pp. 144800–144812, 2019.

[22] Q. Cao, H. Shen, J. Gao, B. Wei, and X. Cheng, "Popularity prediction on social platforms with coupled graph neural networks," in *Proc. Int. Conf. Web Search Data Mining*, 2020, pp. 70–78.

[23] M. Wang and K. Li, "Predicting information diffusion cascades using graph attention networks," in *Proc. Int. Conf. Neural Informat. Process.*, 2020, pp. 104–112.

[24] F. Zhou, X. Jing, X. Xu, T. Zhong, G. Trajcevski, and J. Wu, "Continual information cascade learning," in *Proc. IEEE Global Commun. Conf.*, 2020, pp. 1–6.

[25] Y. Wang, H. Shen, S. Liu, J. Gao, and X. Cheng, "Cascade dynamics modeling with attention-based recurrent neural network," in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 2985–2991.

[26] C. Yang, M. Sun, H. Liu, S. Han, Z. Liu, and H. Luan, "Neural diffusion model for microscopic cascade study," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 3, pp. 1128–1139, Mar. 2021.

[27] Y. Wu, H. Huang, and H. Jin, "Information diffusion prediction with personalized graph neural networks," in *Proc. Int. Conf. Knowl. Sci. Eng. Manage.*, 2020, pp. 376–387.

[28] A. Sankar, X. Zhang, A. Krishnan, and J. Han, "Inf-VAE: A variational autoencoder framework to integrate homophily and influence in diffusion prediction," in *Proc. Int. Conf. Web Search Data Mining*, 2020, pp. 510–518.

[29] Y. Huang and H. Lu, "Hybrid hypergraph construction for facial expression recognition," in *Proc. Int. Conf. Pattern Recognit.*, 2016, pp. 4142–4147.

[30] F. Chen, Y. Gao, D. Cao, and R. Ji, "Multimodal hypergraph learning for microblog sentiment prediction," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2015, pp. 1–6.

[31] B. Fatemi, P. Taslakian, D. Vazquez, and D. Poole, "Knowledge hypergraphs: Prediction beyond binary relations," in *Proc. Int. Joint Conf. Artif. Intell.*, 2020, pp. 2191–2197.

[32] Y. Feng, H. You, Z. Zhang, R. Ji, and Y. Gao, "Hypergraph neural networks," in *Proc. Conf. Assoc. Advance. Artif. Intell.*, 2019, pp. 3558–3565.

[33] J. Jiang, Y. Wei, Y. Feng, J. Cao, and Y. Gao, "Dynamic hypergraph neural networks," in *Proc. Int. Joint Conf. Artif. Intell.*, 2019, pp. 2635–2641.

[34] N. Yadati, M. Nimishakavi, P. Yadav, V. Nitin, A. Louis, and P. Talukdar, "HyperGCN: A new method for training graph convolutional networks on hypergraphs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 1509–1520.

[35] R. Zhang, Y. Zou, and J. Ma, "Hyper-SAGNN: A self-attention based graph neural network for hypergraphs," in *Proc. Int. Conf. Learn. Representations*, 2020.

[36] S. Bai, F. Zhang, and P. H. S. Torr, "Hypergraph convolution and hypergraph attention," *Pattern Recognit.*, vol. 110, pp. 1–8, 2021.

[37] J. Yi and J. Park, "Hypergraph convolutional recurrent neural network," in *Proc. Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 3366–3376.

[38] J. Wang, K. Ding, L. Hong, H. Liu, and J. Caverlee, "Next-item recommendation with sequential hypergraphs," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Informat. Retrieval*, 2020, pp. 1101–1110.

[39] J. Yu, H. Yin, J. Li, Q. Wang, N. Q. V. Hung, and X. Zhang, "Self-supervised multi-channel hypergraph convolutional network for social recommendation," in *Proc. Int. World Wide Web Conf.*, 2021, pp. 413–424.

[40] R. I. Tyshkevich and V. E. Zverovich, "Line hypergraphs: A survey," *Acta Applicandae Mathematicae*, vol. 52, no. 1–3, pp. 209–222, 1998.

[41] T. Hogg and K. Lerman, "Social dynamics of Digg," *EPJ Data Sci.*, vol. 1, no. 1, pp. 5–30, 2012.

[42] M. Jamali, “Flixster data set,” Website, 2010. [Online]. Available: https://sites.google.com/view/mohsenjamali/flixter-data-set

[43] J. Zhang, B. Liu, J. Tang, T. Chen, and J. Li, “Social influence locality for modeling retweeting behaviors,” in *Proc. Int. Joint Conf. Artif. Intell.*, 2013, pp. 2761–2767.

Hai Jin (Fellow, IEEE) received the PhD degree in computer engineering from HUST in 1994. He is a chair professor with computer science and engineering, Huazhong University of Science and Technology (HUST) in China. In 1996, he was awarded a German Academic Exchange Service fellowship to visit the Technical University of Chemnitz in Germany. He worked with The University of Hong Kong between 1998 and 2000, and as a visiting scholar with the University of Southern California between 1999 and 2000. He was awarded Excellent Youth Award from the National Science Foundation of China in 2001. He is a Fellow of CCF, and a life member of the ACM. He has coauthored more than 20 books and published more than 900 research papers. His research interests include computer architecture, parallel and distributed computing, big data processing, data storage, and system security.

Yao Wu received the BS degree in communication from the Huazhong University of Science and Technology (HUST), Wuhan, China in 2016. Currently, she is currently working toward the PhD degree with HUST. Her research interests include data mining and social computing.

Hong Huang received the PhD degree in computer science from the University of Göttingen, Germany in 2016, and the ME degree in electronic engineering from Tsinghua University, Beijing, China in 2012. She is an associate professor with the Huazhong University of Science and Technology, China. Her research interests include social network analysis, data mining, and knowledge graph.

Yu Song received the BS degree in electronic information engineering and the MS degree in computer science and technology from the Huazhong University of Science and Technology, Wuhan, China in 2018 and 2021. His research interests include data mining and social network analysis.

Haohui Wei received the BS degree in computer science and technology from Dalian Maritime University, Dalian, China in 2020. He is currently working toward the master’s degree with the Huazhong University of Science and Technology. His research interests include knowledge graph and collaborative filtering.

Xuanhua Shi is a professor with the School of Computer Science and Technology, Huazhong University of Science and Technology, China. He is the deputy director with the National Engineering Research Center for Big Data Technology and System (NERC-BDTS). His current research interests focus on cloud computing, big data processing, and AI systems. He published more than 100 peer-reviewed publications (such as ASPLOS, VLDB, *ACM Transactions on Computer Systems*, *IEEE Transactions on Parallel and Distributed Systems*). He received research support from a variety of governmental and industrial organizations, such as the National Science Foundation of China, Ministry of Science and Technology, Ministry of Education, and European Union.