Hierarchical Diffusion Attention Network

Zhitao Wang and **Wenjie Li**
Department of Computing, The Hong Kong Polytechnic University, Hong Kong
{csztwang, cswjli}@comp.polyu.edu.hk

Abstract

A series of recent studies formulated the diffusion prediction problem as a sequence prediction task and proposed several sequential models based on recurrent neural networks. However, non-sequential properties exist in real diffusion cascades, which do not strictly follow the sequential assumptions of previous work. In this paper, we propose a hierarchical diffusion attention network (HiDAN), which adopts a non-sequential framework and two-level attention mechanisms, for diffusion prediction. At the user level, a dependency attention mechanism is proposed to dynamically capture historical user-to-user dependencies and extract the dependency-aware user information. At the cascade (i.e., sequence) level, a time-aware influence attention is designed to infer possible future user's dependencies on historical users by considering both inherent user importance and time decay effects. Significantly higher effectiveness and efficiency of HiDAN over state-of-the-art sequential models are demonstrated when evaluated on three real diffusion datasets. The further case studies illustrate that HiDAN can accurately capture diffusion dependencies.

1 Introduction

The emergence of online media has brought fundamental changes to information diffusion styles. Due to these changes, diffusion cascades are massively triggered and traced. These observed diffusion processes provide rich sources for companies to do marketing through forecasting advertisement diffusion or for governments to maintain stability through tracking opinion diffusion. All these related applications require for a good understanding of diffusion mechanisms and accurate predictions of diffusion dynamics. These great requirements have driven many researchers, in recent years, to study diffusion phenomena on online media and particularly focus on diffusion prediction problem [Bourigault *et al.*, 2016; Wang *et al.*, 2017b; 2017a].

Since retrievable information diffusion cascades are often recorded as sequences, researchers recently formulated the problem as a sequence prediction task: given the historically infected users in an information cascade, the next infected user is predicted. With the great success of recurrent neural network (RNN) in sequence modeling, a series of RNN-based sequential models were proposed and their effectiveness was demonstrated on the real diffusion data [Du *et al.*, 2016; Wang *et al.*, 2017b; 2017a]. These models sequentially encode the historical information as hidden states and predict next infected user based on the compressed states. Though a cascade is in the form of sequence, the real diffusion process behind it does not strictly follow the sequential assumption. This is because there exists an underlying user connection graph, which may not be explicitly unobserved but can directly determine the diffusion dependencies among users. For example, given a cascade $\{(A, t_A), (B, t_B), (C, t_C), (D, t_D)\}$ and an underlying graph as shown in Figure 1, the sequential models assume that the infections of C and D are influenced by the hidden states h_2 (compressed information of A and B) and h_3 (compressed information of A, B and C) respectively. But in fact, C and D are directly dependent on A and B according to the graph structure. This kind of non-sequential dependency has also been identified as an important characteristic of diffusion sequences in the previous work [Wang *et al.*, 2017b]. Though the gating mechanisms in existing sequential models [Hochreiter and Schmidhuber, 1997] can selectively drop the information of B from hidden state h_2 when generating C, they also lead to the loss of dependency of D on B in hidden state h_3. The hidden states of the compressed information are not expressive enough for such non-sequential diffusion

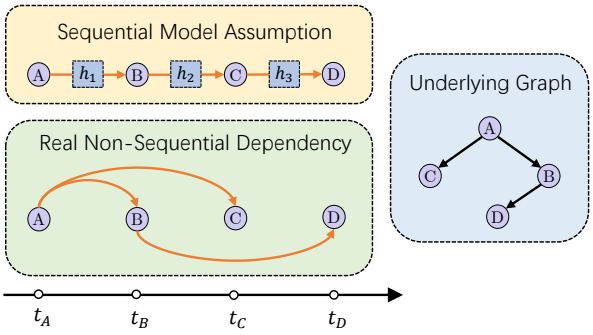


Figure 1: An example of non-sequential diffusion dependency.

dependency, thus the prediction power is limited.

In this paper, we propose a **Hi**erarchical **D**iffusion **A**ttention neural **N**etwork (**HiDAN**) for the problem of predicting diffusion when the underlying graph is unknown. To capture unique properties of diffusion sequences, we devise a non-sequential architecture with two-level attention mechanisms. Specifically, a user-level dependency attention is suggested to dynamically capture diffusion dependencies among historical users. A fusion gate is then designed to selectively integrate user's self information and its dependency context. Based on the dependency-aware historical user information, a cascade-level influence attention, which considers both inherent importance and time-decay effects, is developed to infer the influence of historical users on potentially infected future users. The inferred influence can be interpreted as the possible dependencies of the future user on all historical users. We evaluate the proposed model against state-of-the-art sequential diffusion prediction models on three real diffusion datasets. The significantly better performance demonstrates the effectiveness of our model. The case studies on synthetic datasets further indicate that the learned dependency attentions are mostly consistent with the true underlying graph. In addition, HiDAN also shows its higher efficiency than sequential models in experiments. The main contributions of this paper are summarized as follows:

- To the best of our knowledge, we are the first to develop a **non-sequential neural network framework** for diffusion prediction problem, which is well-adapted to properties of real diffusion cascades.
- We propose two-level attention mechanisms for cascade modeling, i.e., a **user-level dynamic dependency attention**, which effectively captures historical diffusion dependencies, and a **cascade-level time-aware influence attention**, which infers future dependencies by modeling user inherent importance and time-decay effects.
- The experiments on three real datasets demonstrate the significantly improved **effectiveness** and **efficiency** of the proposed model compared with state-of-the-art approaches.

2 Method

2.1 Preliminary

A diffusion cascade can be represented as $c = \{(u_1, t_1), (u_2, t_2), ..., (u_c, t_c)\}$, where the element (u_i, t_i) denotes that user u_i is infected in this cascade at time t_i. The infected users are ordered by time, thus $t_{i-1} < t_i$. Generally, the diffusion prediction problem can be formulated as: given the infection history $\{(u_1, t_1), ..., (u_i, t_i)\}$ of a cascade, the task is to predict user u_{i+1}, who will be infected next. Given a training cascades set $C = \{c^1, ..., c^M\}$, the goal is to build a model that is able to learn the function of the conditional probability $p(u_{i+1}|\{(u_1, t_1), ..., (u_i, t_i)\})$.

2.2 The HiDAN Model

The framework of the proposed HiDAN is illustrated in Fig. 2. Initially, each user of an input cascade is embedded as a user embedding. Given the embedded user information, the user-level attention mechanism dynamically captures the diffusion dependencies between each user and its context users. A gating mechanism is developed to integrate a user's own information and his/her dependency context. Based on the dependency-aware user information, the cascade-level attention computes the influence of historical users on possible future users by capturing users' inherent importance and the time-decay effects. Given the cascade embedding constructed with the influence attention, the model then predicts the next infected user.

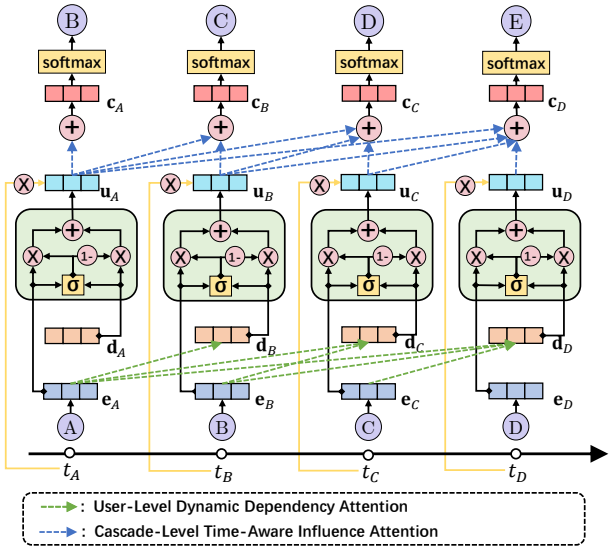


Figure 2: Overview of the HiDAN model. $\mathbf{e}$, $\mathbf{d}$, $\mathbf{u}$ and $\mathbf{c}$ correspond to user embedding, dependency context embedding derived by user-level attention, dependency-aware user embedding constructed by fusion gate and cascade embedding composed by cascade-level attention, respectively. The details will be described in subsection 2.2.

User Embedding

At time t_i, the sequence of already infected users $\{u_1, ..., u_i\}$, ordered by infection time, is regarded as the input to the model. The raw representation of each input user $u_j \in \{u_1, ..., u_i\}$ is the one hot vector of user ID, i.e., $\mathbf{x}_j \in \mathbb{R}^N$, where N is the total number of distinct users. To extract expressive high-level features of users, we transform the raw input $\mathbf{x}$ to the user embedding $\mathbf{e}$ via a fully-connected layer:

$$\mathbf{e}_j = f_x(\mathbf{W}_x \mathbf{x}_j + \mathbf{b}_x) \tag{1}$$

where $\mathbf{W}_x \in \mathbb{R}^{d \times N}$, $\mathbf{b}_x \in \mathbb{R}^d$ are learnable parameters, d is the size of the embedding and f_x is the non-linear activation function.

User-Level Dynamic Dependency Attention

This attention mechanism aims at capturing diffusion dependencies among input cascade users and extracting dependency-aware user features. The diffusion dependency describes who possibly infect(s) whom in the diffusion process, which possesses the following two characteristics. (1) Each cascade user can only be infected by its previous users, thus the dependency of u_j only exists on $\{u_1, ..., u_{j-1}\}$. The previously infected users $\{u_1, ..., u_{j-1}\}$ are referred to as the diffusion context users of u_j. (2) Diffusion dependency is

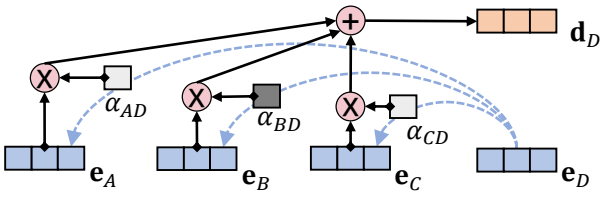


Figure 3: Dependency attention mechanism: an example for u_D.

directional, i.e., the high dependency of u_j on u_k does not indicate same level of dependency of u_k on u_j. This is mainly because the dependency relationship is often directed in the real user graph. For example, in Twitter, a star user, who is followed by millions of users, frequently infects his/her followers instead of being infected by followers. Therefore, it requires differentiating different roles that users play in the directed dependencies.

Based on the above understanding, we propose a dynamic attention mechanism to capture the diffusion dependency for each input user. The dependency attention score between user $u_j \in \{u_1, ..., u_i\}$ and its context user $u_k \in \{u_1, ..., u_{j-1}\}$ is measured as follows:

$$\alpha_{kj} = \frac{\exp(\langle \mathbf{W}_e^c \mathbf{e}_k, \mathbf{W}_e^t \mathbf{e}_j \rangle)}{\sum_{l=1}^{j-1} \exp(\langle \mathbf{W}_e^c \mathbf{e}_l, \mathbf{W}_e^t \mathbf{e}_j \rangle)} \tag{2}$$

where $\mathbf{W}_e^c, \mathbf{W}_e^t \in \mathbb{R}^{d \times d}$ are transformation matrices for the context user and the target user respectively; $\langle \cdot, \cdot \rangle$ represents the inner product. $\mathbf{W}_e^c, \mathbf{W}_e^t$ are employed to differentiate user roles in directed dependencies. When checking whether u_j is dependent on u_k, u_k is regarded as the context user and transformed by $\mathbf{W}_e^c$, while u_j is treated as the target user and transformed by $\mathbf{W}_e^t$. When checking dependency with the opposite direction, the roles of u_k and u_j are reversed, thus dependency scores are different. Similar to most attention mechanism, we apply a softmax function to derive the probability distribution. Therefore, α_{kj} denotes the probability that u_j is dependent on u_k over all his/her context users.

The useful context information of u_j, denoted as $\mathbf{d}_j$, is computed via the following attention weighted sum:

$$\mathbf{d}_j = \sum_{k=1}^{j-1} \alpha_{kj} \mathbf{e}_k \tag{3}$$

The above dependency attention process for one specific user is illustrated in Fig. 3. Since dependency attentions and context embeddings are computed independently for each input user in the proposed non-sequential framework, we are able to parallelize the dynamic processes as matrix computation. Given a input cascade c, where the cascade length is l and the embedding matrix of l users is $\mathbf{E} \in \mathbb{R}^{d \times l}$, we replace time-consuming enumeration by using a mask matrix $\mathbf{M} \in \mathbb{R}^{l \times l}$, where each entry $M_{i,j} = 0$ if $i < j$; otherwise $M_{i,j} = -\infty$. Then we derive the matrix of attentions as:

$$\mathbf{A} = \text{softmax}((\mathbf{W}_e^c \mathbf{E})^T (\mathbf{W}_e^t \mathbf{E}) + \mathbf{M}) \tag{4}$$

where $\mathbf{A} \in \mathbb{R}^{l \times l}$. Each row vector $\boldsymbol{\alpha}_j$ in $\mathbf{A}$ represents u_j's attentions on its context users. The mask forces the softmax function to compute valid attentions only over u_j's context users and assign 0 to other users (infected later than u_j). The matrix of context embeddings can be derived as: $\mathbf{D} = \mathbf{A}\mathbf{E}^T$.

Dependency-Aware Fusion Gating

For each input cascade user u_j, we now have user embedding $\mathbf{e}_j$ and his/her diffusion context embedding $\mathbf{d}_j$. To selectively integrate the important information of two embeddings, a concise and effective fusion gating mechanism is employed. It produces a dependency-aware user representation $\mathbf{u}_j$ as follows:

$$\mathbf{g}_j = \text{sigmoid}(\mathbf{W}_g^1 \mathbf{e}_j + \mathbf{W}_g^2 \mathbf{d}_j + \mathbf{b}_g) \tag{5}$$

$$\mathbf{u}_j = \mathbf{g}_j \odot \mathbf{e}_j + (1 - \mathbf{g}_j) \odot \mathbf{d}_j \tag{6}$$

where $\mathbf{W}_g^1, \mathbf{W}_g^2 \in \mathbb{R}^{d \times d}$ and $\mathbf{b}_g \in \mathbb{R}^d$. $\mathbf{g}$ is used to drop unimportant parts of user embedding $\mathbf{e}_j$ and add new information of its diffusion context embedding $\mathbf{d}_j$ such that the fused embedding $\mathbf{u}_j$ is aware of the diffusion dependency.

Cascade-Level Time-Aware Influence Attention

At the cascade level, we also consider the non-sequential dependency, in which all historical users could trigger the future infection with different probabilities. We interpret such future dependencies as dynamic influence of historical users on the whole cascade, and propose a time-aware influence attention mechanism. Based on the inferred influence, we compose dependency-aware embeddings $\mathbf{u}$ to cascade-level features for final prediction. This attention mechanism captures two factors: the inherent importance of users to cascade and the dynamic time-decay effects.

The inherent importance of u_j describes how important the information in the dependency-aware embedding $\mathbf{u}_j$ is to the cascade. If only considering the inherent importance, we can define the influence with the self-attention mechanism [Lin *et al.*, 2017] as: $\langle \mathbf{w}, f_u(\mathbf{W}_u \mathbf{u}_j + \mathbf{b}_u) \rangle$.

However, user influence is generally assumed to decrease as time passes. This is known as the time-decay effect. Empirical studies [Gomez Rodriguez *et al.*, 2011] have shown that time-decay patterns in different data are not identical, thus predefining the form of time-decay function is often impractical [Cao *et al.*, 2017]. We estimate the time-decay factor in HiDAN via a neural function directly. For $u_j \in \{(u_1, t_1), ..., (u_i, t_i)\}$, the past time at current step t_i can be represented as $\Delta t^j = t_i - t_j$. We discretize the information of past time as a one-hot vector $\text{vec}(\Delta t^j) = \mathbf{t}^j \in \mathbb{R}^T$, where $\mathbf{t}_n^j = 1$ if $t_{n-1} < t_i - t_j < t_n$. The critical time points of the discretization, such as t_{n-1} and t_n, are defined by splitting the time range $(0, T_{\text{max}}]$ into T intervals $\{(0, t_1], ...(t_{n-1}, t_n], ..., (t_{T-1}, T_{\text{max}}]\}$, where T_{max} is the max observation time. We aim at mapping the past time $\mathbf{t}^j$ to a vector $\boldsymbol{\lambda}_j$, describing latent features of time-decay. To capture the non-linearity of the decay effect, we compute $\boldsymbol{\lambda}_j$ via the following fully connected layer:

$$\boldsymbol{\lambda}_j = \text{sigmoid}(\mathbf{W}_t \mathbf{t}^j + \mathbf{b}_t) \tag{7}$$

where $\mathbf{W}_t \in \mathbb{R}^{d \times T}$ and $\mathbf{b}_t \in \mathbb{R}^d$.

Taking both inherent importance and time decay factors into consideration, we define the following time-aware influence attention:

$$\beta_j = \frac{\exp(\langle \mathbf{w}, \boldsymbol{\lambda}_j \odot f_u(\mathbf{W}_u \mathbf{u}_j + \mathbf{b}_u) \rangle)}{\sum_{k=1}^{i} \exp(\langle \mathbf{w}, \boldsymbol{\lambda}_k \odot f_u(\mathbf{W}_u \mathbf{u}_k + \mathbf{b}_u) \rangle)} \tag{8}$$

where $\mathbf{W}_u \in \mathbb{R}^{d\times d}$, $\mathbf{b}_u \in \mathbb{R}^d$ and $\mathbf{w} \in \mathbb{R}^d$. The decay factor vector $\boldsymbol{\lambda}_j$ serves as a soft gate, which selectively drops the information of already infected users according to their infection times.

Given the user influence β_j, this layer finally composes all dependency-aware user embeddings and constructs the cascade embedding at time t_i as follows:

$$\mathbf{c}_i = \sum_{j=1}^{i} \beta_j \mathbf{u}_j \tag{9}$$

Prediction Layer

Given cascade embedding $\mathbf{c}_i$ at t_i, HiDAN predicts the probability of next infected user over all possible users as:

$$\hat{p}(u_{i+1}|\mathbf{c}_i) = \text{softmax}(\mathbf{W}_c\mathbf{c}_i + \mathbf{b}_c) \tag{10}$$

where $\mathbf{W}_c \in \mathbb{R}^{N\times d}$, $\mathbf{b}_c \in \mathbb{R}^N$.

2.3 Model Optimization

Given the training set $C = \{c^1, ..., c^M\}$, the learning objective is to minimize the following negative log-likelihood loss:

$$\mathcal{L}(C) = -\sum_{m=1}^{M}\sum_{i=1}^{n_m-1} \log \hat{p}(u_{i+1}|\mathbf{c}_i^m) \tag{11}$$

where u_{i+1} is truly infected user in cascade c^m at time t_{i+1}. The backpropagation algorithm is utilized in the training process. As for parameters updating, we employ stochastic gradient descent (SGD) method with mini-batch and adopt the Adam optimizer [Kingma and Ba, 2015].

3 Experiments

3.1 Data

To verify the effectiveness of the proposed model, we conduct comparative experiments on the following three real datasets, which are representative in information diffusion studies.

- **Memes** [Leskovec *et al.*, 2009]: This dataset contains articles from mainstream news websites or blogs. Each cascade records the diffusion process of a specific key phrase and is represented by a sequence of webpage links associated with corresponding timestamps.
- **Weibo** [Zhang *et al.*, 2013]: This dataset consists of content reposting logs crawled from Sina Weibo, a Chinese microblogging site. Each reposting log represents a diffusion process, in which users are ordered as a sequence according to the time they repost.
- **Twitter** [Weng *et al.*, 2013]: This dataset records the diffusion processes of hash-tags in Twitter. The sequences of users and timestamps of using the same hash-tags are traced as diffusion cascades.

Following the previous work [Wang *et al.*, 2017b], we select frequent users and corresponding cascades as experimental data. The detailed statistics are presented in Table 1. We randomly sample 80% of cascades for training and the rest for validating and testing with an even split.

	Memes	**Weibo**	**Twitter**
# Users	1,109	8,190	13,755
# Cascades	42,492	43,365	72,103
Avg. Cascade Length	8.8	22.5	9.4

Table 1: Statistics of experimental data.

3.2 Baselines

We compare the proposed model, HiDAN, with the following popular and strong sequential baselines.

- **RNN**: RNN represents the basic recurrent neural network sequential model.
- **LSTM** [Hochreiter and Schmidhuber, 1997]: Long short-term memory (LSTM) network is a stronger RNN-based sequential model, which employs an effective gating mechanism to control the information flow in sequence.
- **RMTPP** [Du *et al.*, 2016]: Recurrent marked temporal point process (RMTPP) is the state-of-the-art sequential models for sequence prediction. Besides modeling marker (diffusion user) sequence, it additionally models timing information with a temporal point process.

The following state-of-the-art attention based sequential models are compared. All of them compute attentions on hidden states.

- **Att-RNN**: Att-RNN employs a representative attention mechanism [Luong *et al.*, 2015] in RNN. Attentions are computed between current hidden state and previous states.
- **Att-LSTM**: Att-LSTM employs the same attention mechanism as Att-RNN in the LSTM framework.
- **CYAN-RNN** [Wang *et al.*, 2017b]: This is the latest attention-based sequential method for cascade prediction. Instead of using a single-chain RNN, it employs a encoder-decoder architecture where a coverage-based alignment mechanism is applied. Attentions are computed between current decoder states and previous encoder states.

The following variants of HiDAN are also evaluated.

- **HiDAN$_{\text{NUA}}$**: This is a user-level attention-free variant, which replaces dependency attention and fusion gate with an average operation over embeddings of context users.
- **HiDAN$_{\text{NCA}}$**: This is a cascade-level attention-free variant. Influence attention is substituted by the average operation.
- **HiDAN$_{\text{NT}}$**: This is a variant without considering time-decay in cascade-level attention.

3.3 Evaluation Metrics and Settings

The performance is evaluated by predicting the next infected user based on previous infections. Due to the large number of potential targets, this prediction task is often regarded as a ranking problem [Wang *et al.*, 2017b]. Given the output probabilities of all users, the ground-truth user, who is truly infected at next step, is expected to get higher probability. We adopt two widely used ranking metrics for evaluation: Mean Reciprocal Rank (*MRR*) and Accuracy on top k ($A@k$) [Wang *et al.*, 2017b].

The size of hidden unit is set to 64 for all baselines. Other parameters of baselines follow the recommended settings in

	Memes				Weibo				Twitter			
Model	*MRR*	*A@10*	*A@50*	*A@100*	*MRR*	*A@10*	*A@50*	*A@100*	*MRR*	*A@10*	*A@50*	*A@100*
RNN	23.26	40.23	66.33	77.24	1.33	2.25	6.04	9.49	2.04	3.68	9.83	14.92
LSTM	24.08	41.49	67.23	77.92	1.40	2.63	7.23	11.49	2.47	4.69	11.78	16.63
RMTPP	23.35	41.37	66.34	76.99	1.35	2.28	6.69	10.73	1.73	3.17	8.96	13.64
Att-RNN	23.51	42.05	67.14	78.10	1.57	2.52	7.51	12.18	2.53	4.56	13.68	20.14
Att-LSTM	24.39	43.11	68.69	79.55	1.64	2.93	8.20	12.60	2.73	5.08	14.78	21.71
CYANRNN	17.62	35.84	57.29	69.81	1.06	1.53	5.18	7.83	1.19	1.82	5.69	8.97
$\text{HiDAN}_{\text{NUA}}$	16.60	33.41	61.59	73.75	1.44	2.76	8.14	12.69	2.50	4.72	11.86	16.71
$\text{HiDAN}_{\text{NCA}}$	20.41	38.78	65.55	77.70	1.38	2.47	7.14	11.29	2.45	4.60	11.02	16.32
HiDAN_{NT}	27.12	47.92	73.47	83.26	2.39	4.06	11.02	16.83	5.46	11.05	23.08	29.12
HiDAN	**27.91**	**48.89**	**74.63**	**84.44**	**2.48**	**4.30**	**11.31**	**17.30**	**5.74**	**11.18**	**23.61**	**30.41**

Table 2: Diffusion prediction performance.

original papers. For the proposed models, the dimension size of d is also 64, the learning rate is 0.001, the max observation time T_{max} is 120 hours, the number of splitting time interval T is 40, and the non-linear activation functions f_x, f_u are selected as Exponential Linear Unit (ELU) [Clevert *et al.*, 2016]. We also apply the Dropout [Srivastava *et al.*, 2014] with the keep probability 0.8 and the L2 regularization on parameters to avoid over-fitting.

3.4 Evaluation Results

As shown in Table 2, the proposed HiDAN consistently and remarkably outperforms all compared methods in terms of MRR, A@10, A@50 and A@100 on three datasets. The superiority of HiDAN on diffusion prediction is clearly demonstrated. In addition, we observe the following important findings.

- **The non-sequential framework is capable for cascade modeling**. HiDAN significantly outperforms all baselines. Even without either of the important attention mechanisms, its variants can still achieve competitive and better performance than attention-free sequential models on Weibo and Twitter datasets. This indicates that the proposed non-sequential architecture is capable of modeling cascade without sequential assumptions.
- **Attention mechanism is beneficial for diffusion prediction**. Almost all attention-based sequential models perform better than their non-attention versions. Excluding attentions from the full HiDAN also brings notable decreases on performance. The proposed hierarchical attentions are specific for capturing historical non-sequential dependencies and inferring future dependencies. Attentions in sequential models also aim at computing long-term dependencies to alleviate sequential assumptions. This finding is consistent with our argument that modeling non-sequential dependencies is important for diffusion prediction.
- **HiDAN is more effective in capturing diffusion dependencies**. Compared with state-of-the-art attention-based sequential models, HiDAN gains a very significant improvement. The attention-based sequential models define attentions on hidden states, which represent the accumulated information but not independent information of each historical user. They cannot clearly capture user-to-user dependencies. Differently, HiDAN does not sequentially compress user information but directly computes user-level attentions with unique user embeddings. HiDAN captures dependencies more explicitly and effectively.
- **Time decay matters to influence modeling**. HiDAN outperforms its variant HiDAN_{NT} across all datasets. Taking time information into account helps infer user influence (dependencies) on future infections more accurately.

3.5 Case Studies on Diffusion Dependency

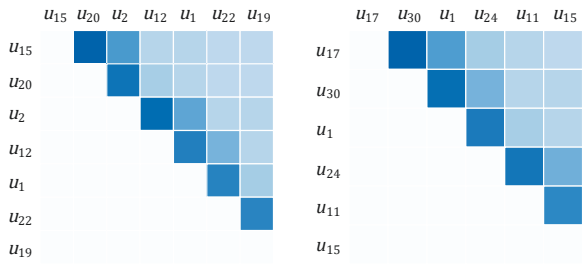
The experimental results have shown that user-level dependency attention plays an important role in HiDAN. Here, we further investigate whether the proposed mechanism is better at capturing user-to-user diffusion dependencies. Since it is very difficult to access the complete graph of real data, we utilize two synthetic data (CP-Exp and RD-Exp) provided in the previous work [Wang *et al.*, 2017b] for case studies. The graphs are created by Kronecker Generator [Leskovec *et al.*, 2010]. Given the created graph, cascades are generated by simulation processes [Gomez Rodriguez *et al.*, 2011].

We visualize attention matrices of the sampled cases learned by HiDAN and its best competitor Att-LSTM. Meanwhile, the adjacency matrices of the users who are involved in the cases are visualized as ground-truth. As illustrated in Fig. 4, each element (u_i, u_j) in the learned attention matrices indicates how much u_j is infected by u_i. The deeper the color is, the greater the attention is. Each element (u_k, u_l) in the ground-truth matrices denotes whether or not there is a directed edge from u_k to u_l (i.e., black indicates 'yes' and gray indicates 'no').

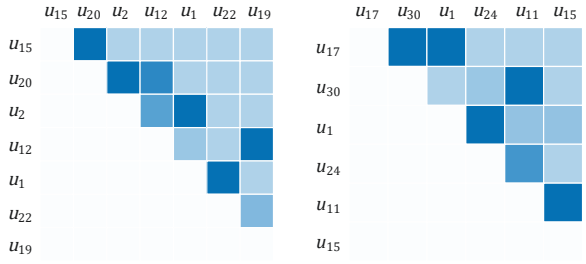
Compared with Att-LSTM, attentions learned by HiDAN are more consistent with ground-truth dependencies. Despite of employing attention mechanism, Att-LSTM mainly focuses on the most recent hidden state. On the contrary, HiDAN is more aware of cross dependencies, especially long-term and multi- dependencies. In the sampled case of the CP-Exp data (left 3 images), the dependencies of all users except u_{12} are correctly allocated by HiDAN. In the sampled case of the RD-Exp (right 3 images), HiDAN accurately captures dependencies for u_{30}, u_1 and u_{24}. It is interesting that both u_{11} and u_{15} have multiple paths connected to previously infected users, as shown in the ground-truth matrix. HiDAN is able to recognize such kind of multiple diffusion paths.

3.6 Efficiency Analysis

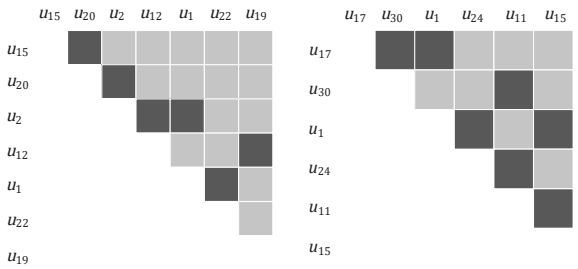
Apart from effectiveness, higher efficiency is another crucial advantage of HiDAN. To demonstrate this, we conduct a



(a) Dependencies learned by Att-LSTM



(b) Dependencies learned by HiDAN



(c) Ground-truth dependencies

Figure 4: Case studies on learned dependencies.

training time comparison. All models except CYAN-RNN[1] are implemented with Tensorflow and trained on the same GTX1080Ti graphic card with the same batch size.

As shown in Table 3, HiDAN has relatively fewer parameters than Att-LSTM and CYAN-RNN. More importantly, HiDAN is super faster than all compared sequential models. This can be attributed to its non-sequential architecture. The recurrent layer in sequential models has an approximate $O(ld^2)$ complexity. However, HiDAN replaces the recurrent layer with the dependency attention, which can be parallelized with matrix computation as shown in Eq.(4), and time complexity is only about $O(l^2d)$. Compared with hidden size d (64 in experiments), average cascade length l (9, 23 and 10 in experiments) is often smaller in real data. Therefore, HiDAN has a lower complexity at the historical user encoding layer. Additionally, without sequential assumptions, HiDAN can compute outputs of all steps in parallel with a $O(1)$ complexity at the prediction layer, whereas sequential models need to output step by step with a $O(l)$ complexity. These dramatically speed up the training of HiDAN especially when the cascade length is getting larger.

[1]Released code is in JAVA. GPU training time is unknown. But as an RNN-based model, it is at least not faster than RNN.

	# Param.	Memes	Weibo	Twitter
RNN	6.2k	22 s	145 s	261 s
LSTM	24.8k	34 s	190 s	346 s
RMTPP	8.8k	24 s	148 s	265 s
Att-RNN	14.5k	29 s	152 s	273 s
Att-LSTM	33.1k	38 s	203 s	422 s
CYAN-RNN	27.1k	$\geq$22 s	$\geq$145 s	$\geq$261 s
HiDAN	25.6k	**12 s**	**36 s**	**85 s**

Table 3: Average training time (seconds) per epoch.

4 Related Work

Diffusion Prediction

Inspired by the theoretical independent cascade (IC) model [Kempe *et al.*, 2003], most previous work focused on IC-based methods for diffusion prediction [Saito *et al.*, 2009; Gomez Rodriguez *et al.*, 2011]. Since these methods require strong hypothesis of diffusion patterns, which is hard to specify and verify in practice, they are often not effective on real data [Wang *et al.*, 2017b]. Several recent studies formulated this problem as sequence prediction task and a series of RNN-based models were proposed. RMTPP [Du *et al.*, 2016] is the first RNN-based model for predicting cascade dynamic. It models both timing and user information in sequences with a basic RNN framework. Topo-LSTM [Wang *et al.*, 2017a] employs the LSTM framework and considers the topology. However, the requirement for complete graph limits its application on the data without graph information. Another representative work is CYANRNN[Wang *et al.*, 2017b], which adopts an encoder-decoder framework and a machine translation alignment mechanism. Different from previous sequential models, we are the first to develop a non-sequential neural network for diffusion prediction.

Attention in Neural Network

Attention mechanism has been widely used in many sequence-based tasks, including neural machine translation [Luong *et al.*, 2015; Bahdanau *et al.*, 2015] and sequence embedding [Lin *et al.*, 2017]. In a series of recent works, the attention mechanisms have proven more expressive in the RNN-free architecture. The RNN-free attention neural network Transformer [Vaswani *et al.*, 2017] was firstly proposed for machine translation task. Another representative model [Shen *et al.*, 2018] was then proposed for various single sequence problems. Besides, some researchers [Lin *et al.*, 2017] focused on developing self attention network to capture the importance of elements in sequences. Inspired by them, we propose hierarchical attentions which are well adapted to unique properties of diffusion cascades on different levels.

5 Conclusion

In this paper, we propose a hierarchal attention neural network for diffusion prediction, which is well adapted to the non-sequential characteristics of diffusion cascades. The proposed two-level attentions are able to capture historical user-to-user dependencies and infer future dependencies. The experiments on three real diffusion datasets demonstrate the effectiveness and efficiency of our model when compared with state-of-the-art sequential models.

Acknowledgments

The work described in this paper was supported by National Natural Science Foundation of China (61672445) and The Hong Kong Polytechnic University (G-YBJP).

References

[Bahdanau *et al.*, 2015] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *International Conference on Learning Representations*, 2015.

[Bourigault *et al.*, 2016] Simon Bourigault, Sylvain Lamprier, and Patrick Gallinari. Representation learning for information diffusion through social networks: an embedded cascade model. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 573–582. ACM, 2016.

[Cao *et al.*, 2017] Qi Cao, Huawei Shen, Keting Cen, Wentao Ouyang, and Xueqi Cheng. Deephawkes: Bridging the gap between prediction and understanding of information cascades. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1149–1158. ACM, 2017.

[Clevert *et al.*, 2016] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *International Conference on Learning Representations*, 2016.

[Du *et al.*, 2016] Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1555–1564. ACM, 2016.

[Gomez Rodriguez *et al.*, 2011] Manuel Gomez Rodriguez, David Balduzzi, and Bernhard Schölkopf. Uncovering the temporal dynamics of diffusion networks. In *Proceedings of the 28th International Conference on Machine Learning*, pages 561–568. ACM, 2011.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Kempe *et al.*, 2003] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[Kingma and Ba, 2015] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[Leskovec *et al.*, 2009] Jure Leskovec, Lars Backstrom, and Jon Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 497–506. ACM, 2009.

[Leskovec *et al.*, 2010] Jure Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. Kronecker graphs: An approach to modeling networks. *Journal of Machine Learning Research*, 11(Feb):985–1042, 2010.

[Lin *et al.*, 2017] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. *International Conference on Learning Representations*, 2017.

[Luong *et al.*, 2015] Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, 2015.

[Saito *et al.*, 2009] Kazumi Saito, Masahiro Kimura, Kouzou Ohara, and Hiroshi Motoda. Learning continuous-time information diffusion model for social behavioral data analysis. In *Asian Conference on Machine Learning*, pages 322–337. Springer, 2009.

[Shen *et al.*, 2018] Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. Disan: Directional self-attention network for rnn/cnn-free language understanding. In *AAAI Conference on Artificial Intelligence*, 2018.

[Srivastava *et al.*, 2014] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958, 2014.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010. 2017.

[Wang *et al.*, 2017a] Jia Wang, Vincent W Zheng, Zemin Liu, and Kevin Chen-Chuan Chang. Topological recurrent neural network for diffusion prediction. In *Data Mining (ICDM), 2017 IEEE International Conference on*, pages 475–484. IEEE, 2017.

[Wang *et al.*, 2017b] Yongqing Wang, Huawei Shen, Shenghua Liu, Jinhua Gao, and Xueqi Cheng. Cascade dynamics modeling with attention-based recurrent neural network. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 2985–2991, 2017.

[Weng *et al.*, 2013] L. Weng, F Menczer, and Y. Y. Ahn. Virality prediction and community structure in social networks. *Scientific Reports*, 3(8):618–618, 2013.

[Zhang *et al.*, 2013] Jing Zhang, Biao Liu, Jie Tang, Ting Chen, and Juanzi Li. Social influence locality for modeling retweeting behaviors. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 2761–2767, 2013.