Information Diffusion Prediction via Recurrent Cascades Convolution

Xueqin Chen∗, Fan Zhou‡, Kunpeng Zhang†, Goce Trajcevski§, Ting Zhong∗, Fengli Zhang∗
∗School of Information and Software Engineering, University of Electronic Science and Technology of China
‡Department of Decision, Operations & Information Technologies, University of Maryland, college park MD
§Department of Electrical and Computer Engineering, Iowa State University, Ames IA
†Corresponding author: fan.zhou@uestc.edu.cn

Abstract—Effectively predicting the size of an information cascade is critical for many applications spanning from identifying viral marketing and fake news to precise recommendation and online advertising. Traditional approaches either heavily depend on underlying diffusion models and are not optimized for popularity prediction, or use complicated hand-crafted features that cannot be easily generalized to different types of cascades. Recent generative approaches allow for understanding the spreading mechanisms, but with unsatisfactory prediction accuracy.

To capture both the underlying structures governing the spread of information and inherent dependencies between re-tweeting behaviors of users, we propose a semi-supervised method, called Recurrent Cascades Convolutional Networks (CasCN), which explicitly models and predicts cascades through learning the latent representation of both structural and temporal information, without involving any other features. In contrast to the existing single, undirected and stationary Graph Convolutional Networks (GCNs), CasCN is a novel multi directional/dynamic GCN. Our experiments conducted on real-world datasets show that CasCN significantly improves the prediction accuracy and reduces the computational cost compared to state-of-the-art approaches.

Index Terms—information cascade, structural-temporal information, cascade size prediction, deep learning

I. INTRODUCTION

Online social platforms allow their users to generate and share various contents and communicate on topics of mutual interest. Such activities facilitate fast diffusion of information and, consequently, spur the phenomenon of information cascades. The phenomenon is ubiquitous – i.e., it has been identified in various settings: paper citations [1], blogging space [2], [3], email forwarding [4], [5]; as well as in social sites (e.g., Sina Weibo [6] and Twitter [7], [8]). A body of research in various domains has focused on modeling cascades, with significant implications for a number of applications, such as marketing viral discrimination [9], influence maximization [10], [11], media advertising [12] and fake news detection [13]–[15]. Cascade prediction problem turns out to be of utmost importance since it enables controlling (or accelerating) information spreading in various scenarios.

The plethora of approaches proposed to tackle cascade prediction problem fall into four main categories: (1) diffusion model-based approaches [16]–[19], which characterize the diffusion process of information – but heavily depend on assumed underlying diffusion models and are not optimized for cascade prediction; (2) feature-based approaches – mostly focusing on identifying and incorporating complicated hand-crafted features, e.g., structural [20]–[22], content [23]–[26], temporal [27], [28], etc. Their performance strongly depends on extracted features requiring extensive domain knowledge, which is hard to be generalized to new domains; (3) generative approaches – typically relying on Hawkes point process [6], [29], [30], which models the intensity function of the arrival process for each message independently, enabling knowledge regarding the popularity dynamics of information – but with less desirable predictive power; and (4) deep learning based methods, especially Recurrent Neural Networks (RNN) based approaches [6], [8], [31], [32] – which automatically learn temporal characteristics but fall short in the intrinsic structural information of cascades, essential for cascade prediction [33].

Challenges and Our Approach: Effective and efficient prediction of the size of cascades has several challenges: (1) lack of knowledge of complete network structure through which the cascades propagate [34]. This impedes many global structure based approaches since obtaining or further embedding a complete graph is hard, if not impossible. (2) efficient representation of cascades – difficult due to their varying size (from very few to millions [33]), making the random walk based cascade sampling methods biased and ill-suited. (3) modeling diffusion dynamics of information cascades not only requires locally structural characteristics (e.g., community size and activity degree of users) but also needs some temporal characteristics – e.g., information within the first few hours plays crucial role in determining the cascades’ size.

To address above challenges, we propose a novel framework called CasCN (Recurrent Cascades Convolutional Networks) which, while relying on existing paradigms, incorporates both structural and temporal characteristics for predicting the future size of a given cascade. Specifically, CasCN samples sub-cascade graphs rather than a set of random-walk sequences from a cascade, and learns the local structures of each sub-cascades by graph convolutional operations. The convoluted spatial structures are then fed into a recurrent neural network for training and capturing the evolving process of a cascade structure. Our main contributions and advantages of CasCN are:

• Use of less information: We rely solely on the structural and
temporal information of cascades, avoiding massive and complex feature engineering, and our model is more generalizable to new domains. In addition, CasCN leverages deep learning to learn latent semantics of cascades in an end-to-end manner.

- **Representation of a cascade graph**: We sample a cascade graph as a sequence of sub-cascade graphs and use an adjacency matrix to represent each sub-cascade graph. This fully preserves the structural dynamics of cascades as well as the topological structure at each diffusion time, while eliminating the intensive computational cost when operating large graphs.

- **Additional impacting factors**: CasCN takes into account two additional crucial factors for estimating cascade size – the directionality of cascade graphs and the time of re-tweeting (e.g., decay effects).

- **Multi-cascade convolutional networks**: We propose a holistic approach, with variants capturing spatial, structural, and directional patterns in multiple sub-graphs, aware of temporal evolution of dynamic graphs – making our methodology readily applicable to other spatio-temporal data prediction tasks.

- **Evaluations on real-world datasets**: We conduct extensive evaluations on several publicly available benchmark datasets, demonstrating that CasCN significantly outperforms the state-of-the-art baselines.

**Organization**: In the rest of this paper, Section II reviews the related work, followed by Section III which formalizes the problem and introduces the preliminary background. In Section IV, we describe the main aspects of CasCN methodology in details. Experimental evaluations quantifying the benefits of our approach are presented in Section V and Section VI concludes the paper and outlines directions for future work.

II. RELATED WORK

There exists a large body of related research on information cascade prediction and graph representation learning.

A. Information Cascades Modeling & Prediction

The works on information cascades modeling mainly focus on two levels: (1) At micro-level, local patterns of social influence are studied – e.g., inferring the action status of a user [31], [32]. The approaches predict the likelihood of a user propagating a particular piece of information, or forecast when the next propagation might occur given a certain information cascade [32]. (2) At macro-level, typical studies include cascade growth prediction [6], [8], [22], [23], [31] and outbreak prediction (above a certain threshold) [7], [22], [33], [35].

The methods on information cascades prediction fall into the following four categories:

- **Diffusion model-based approaches** strongly assume that the underlying diffusion model is known as a prior. Typical examples include independent cascade model [16]–[19] and linear threshold model [17]. Specifically, latent influence and susceptibility (LIS) model to directly learn user-specific influence and susceptibility, and naturally capture context-dependent factors is proposed in [18]. Information propagation via survival theory is described in [16], and [17] implements both independent cascade model and linear threshold model for information propagation, assuming a uniform or a degree-modulated propagation probability for a piece of information. While these methods gain success at characterizing the diffusion process of information, they heavily depend on the underlying diffusion model and are not quite appropriate for cascade prediction.

- **Generative process approaches** focus on modeling the intensity function for each message arrival independently. Typically, they observe every event and learn parameters by maximizing the probability of events occurring during the observation time window. There exist two typical generative processes: (1) Poisson process – [1], [36], mainly modeling the stochastic popularity dynamics by employing Reinforced Poisson Process and incorporate it into the Bayesian framework for external factor inference and parameter estimation, (2) Hawkes process [6], [29], [30] – constructing predictors that combine Hawkes self-exciting point process for modeling each cascade and leverage feature-driven method for estimating the content virality, memory decay, and user influence (cf. [29]). These methods demonstrate an enhanced comprehensibility, but are unable to fully leverage the implicit information in the cascade dynamics for satisfactory prediction.

- **Feature-based approaches** extract various hand-crafted features from raw data, typically including information content features [23]–[26], user characteristics [35], [37], [38], cascade’s structural [20]–[22] and temporal features [27], [28] – and then feed them into discriminative machine learning algorithms to perform prediction. Combining content information with other types of features, e.g., temporal and structural features, can significantly reduce the prediction error [23]. Incorporating features related to early-adopters (cf. [37]) demonstrated that user features are informative predictors; and recent results comparing the prediction power of models using different sets of features (cf. [39]), found that temporal features have largest impact on prediction. However, [33] concluded that both temporal and structural features are almost equally effective in predicting cascade size. All these features heavily depend on domain knowledge. The non-existence of a standard and systematic way to design features, makes such methods hard to generalize. Moreover, the conclusions of existing works are sometimes contradictory, largely due to the heterogeneity among different types of social networks.

- **Deep learning based approaches** are inspired by the recent successes of deep learning in many fields, and cascade prediction using deep neural networks has achieved significant performance improvements [6], [8], [31], [32]. The first deep learning based predictor of information cascades (DeepCas), presented [8], transforms the cascade graph as node sequences through random walk and automatically learns the representation of individual graphs. A deep learning based process that inherits the high interpretability of Hawkes process, with the high predictive power of deep learning methods was proposed in [6]. Coverage and attention mechanisms for capturing the cross-dependencies in cascades and alignments to better reflect the structural information was introduced in [32], whereas [31] employs topological RNNs to explore
the dynamic directed acyclic graph diffusion structure and tailor it for the task of node activation prediction. Overall, these approaches treat the cascade modeling as a sequence modeling problem using RNN – avoiding strong prior knowledge imposed by the diffusion models and feature design, while flexibly capturing sequential dependence in cascades. However, they lack good learning abilities in modeling structural information and dynamics in cascade, largely due to the biased cascade sampling methods and inefficient local structure embedding. Furthermore, the methods incur intensive computation overhead from the node sampling and subsequent embedding, especially for larger cascades.

B. Graph Convolutional Network

Graph convolutional networks (GCN [40]) learn a convolutional operation in Fourier domain by computing the eigen-decomposition of the Laplacian graph, bridging the spectral graph theory and deep neural networks on graph learning. An extension on fast localized convolutional filters on graphs is presented in [41] where the filters are approximated via a Chebyshev expansion of the Laplacian. GCNs model proposed in [42] simplifies previous works by restricting the filters in a first-order approximation of spectral graph convolution. Recently, a self-attention mechanism was introduced into GCN in [43]. Authors extend GCN to model multivariate time series structured data to identify dynamic patterns for modeling large and dense graphs. Merging CNN and RNN for graph convolutional networks (GCN) learns a convolutional operation in Fourier domain by computing the eigen-coefficients.

### Table I

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_i$</td>
<td>A message post, e.g., a tweet or a paper.</td>
</tr>
<tr>
<td>$C_i(t)$</td>
<td>A cascade graph regarding post $p_i$.</td>
</tr>
<tr>
<td>$T$</td>
<td>Observation time window.</td>
</tr>
<tr>
<td>$g_i^t$, $a_i^t$</td>
<td>A sub-cascade graph of $C_i(t)$ at diffusion time $t_j$ and its adjacency matrix.</td>
</tr>
<tr>
<td>$G_i^t$, $A_i^t$</td>
<td>A sequence of sub-cascade graphs of $C_i(t)$ and the corresponding adjacency matrices.</td>
</tr>
<tr>
<td>$\Delta t$</td>
<td>The fixed time interval.</td>
</tr>
<tr>
<td>$\Delta S_i$</td>
<td>The increment size of $p_i$ after $\Delta t$.</td>
</tr>
<tr>
<td>$T_c$</td>
<td>Transition matrix of a cascade.</td>
</tr>
<tr>
<td>$\phi$, $\Phi$</td>
<td>Stationary transition distribution and diagonalized $\phi$.</td>
</tr>
<tr>
<td>$\Delta_\lambda$</td>
<td>Laplacian of a cascade.</td>
</tr>
<tr>
<td>$\lambda_{max}$</td>
<td>The largest eigenvalue of Laplacian.</td>
</tr>
<tr>
<td>$K$</td>
<td>Maximum steps from the central node, i.e., $K^{th}$-order neighborhood or Chebyshev coefficients.</td>
</tr>
</tbody>
</table>

**Definition 1. Cascade Graph.** Suppose we have $n$ posts, $P = \{p_i, i \in [1, n]\}$. An evolving cascade graph for a given message post $p_i$, is a sequence $C_i(t) = [\{U_i(t_1), E_i(t_1)\}, \ldots, \{U_i(t_m), E_i(t_m)\}]$ where $U_i(t_j)$ is a set of nodes and $E_i(t_j) \subset U_i(t_j) \times U_i(t_j)$ is a set of edges, corresponding to the nodes and edges of $C_i(t)$ that are associated with $p_i$ at time-instant $t_j$. Thus, the cascade graphs are evolving sequences of directed acyclic graphs (DAG).

![Fig. 1. The cascade graph of a post $p_i$. Node $V_0$ initiates the original message $p_i$.](image)

We use $g_i^t = \{U_i^t, E_i^t, t_j\}$ as a shorthand for the snapshot of cascade graph $C_i(t)$ that reflecting the diffusion status of post $p_i$ at $t_j$. For example: a node $x \in U_i^t$ represents a user who tweets or re-tweets the post $p_i$ from some sources (e.g., other users) in Twitter or a paper in the citation networks; an edge $(x, y) \in E_i^t$ represents a relationship between $x$ and $y$ (e.g., re-tweet or citation); and $t_j$ represents the time-instant when the re-tweeting or citation behavior occurs. Fig. 1 illustrates how the cascade graph can be represented as $g_i^0 = \{(V_0), \{\emptyset\}, t_0\}, g_i^1 = \{(V_0, V_1), \{(V_0, V_1)\}, t_1\}, \ldots, g_i^5 = \{(V_0, V_1), (V_0, V_2), (V_1, V_3), (V_4, V_5), (V_0, V_1), (V_0, V_2), (V_1, V_3), (V_1, V_4), (V_3, V_5)\}, t_5\).

Given a cascade graph $C_i(t)$ of a post $p_i$ within an observation time window $T$, we can get different snapshots on a social network to describe the process of information diffusion (e.g., specific tweets, rumors, etc.), where individuals can merely observe their immediate neighbors.

**A. Problem Definition**

We cast the cascade size prediction as a regression problem aiming at predicting the size of information cascades, used

**III. Preliminaries**

We now present the necessary background and formally define the problem, and introduce the preliminaries regarding structural and temporal modeling in information cascades.

**A. Problem Definition**

We cast the cascade size prediction as a regression problem aiming at predicting the size of information cascades, used
\( g_{ij}^t \) of \( C_i(t) \), which form a sequence of sub-cascade graphs \( G_i^T \). In this paper, our task is to predict the increment size \( \Delta S_i \) regarding the post \( p_i \) for a fixed time interval \( \Delta t \), i.e., \( \Delta S_i = |U_i^{T+\Delta t}| - |U_i^T| \).

**Definition 2.** The cascade size predictor is a function \( f(\cdot) \) that is to be learned, mapping \( G_i^T \) to \( \Delta S_i \):

\[
G_i^T = \left\{ g_{i1}^0, \ldots, g_{im}^{t-1}; t_j \in [0, T] \right\} f(\cdot) \Delta S_i
\]  

(1)

**B. Recurrent Neural Network & Graph Convolution Networks**

The common method to model temporal dependencies is to leverage recurrent neural networks (RNNs), typically including popular Long Short-Term Memory (LSTM) \([47]\) and Gated Recurrent Units (GRU) \([48]\). In particular, LSTM has been proven stable and powerful for modeling long-range dependencies in various general-purpose sequence modeling tasks \([49]\--\[51]\) and cascade prediction \([8]\, [31]\).

We model the structural information of cascades using Deferrard’s graph convNet \([41]\) – a popular Graph Convolutional Network (GCN) method that defines a spectral formulation in Fourier domain for the convolution operator on graphs. Ferraros graph convNet \([41]\) is the generator of the post \( p_i \). CasCN leverages LSTM and GCN to fully extract temporal and structural information from the cascade graph. After an overview of CasCN, we focus on the details in the respective sub-sections.

Our deep learning framework \( \text{CasCN} \) takes a cascade graph \( C_i(t) \) as an input and predicts the increment size \( \Delta S_i \) regarding certain information (e.g., a post) \( p_i \). CasCN leverages LSTM and GCN to fully extract temporal and structural information from the cascade graph. After an overview of CasCN, we focus on the details in the respective sub-sections.

An end-to-end type of framework, \( \text{CasCN} \) consists of three basic components, depicted in Fig. 2: (1) Cascade graph sampling: dynamically samples a sequence of sub-cascade graphs from the original cascade graph, and then represents sub-cascade graphs as a sequence of adjacency matrices; (2) Structural and temporal modeling: feeds the adjacency matrix sequences and the structural information of cascade graphs (i.e., the Laplacian matrices of cascade graphs) within an observation window into a neural network. It combines recurrent neural networks and graph convolutional networks with time decay function to learn the representation of cascades; (3) Prediction network: a Multi-Layer Perceptron (MLP) is used to predict the increment cascade size based on the representation learned from previous steps.

**A. Cascade Graph as Sub-cascade Graph Sequences**

Given a post \( p_i \), the first step in \( \text{CasCN} \) is to initialize the representation of its corresponding cascading graph, \( C_i(t) \). Existing methods typically treat the graph in two ways: either sampling the graph as a bag of nodes, which ignores both local and global structural information, or denoting the graph as a set of paths. For example, DeepCas \([8]\) samples a set of paths from each cascade. The sampling process could be generalized as performing a random walk over a cascade graph similar to DeepWalk which, however, fails to consider dynamics of cascades – one of the most important factors in information diffusion. DeepHawkes \([6]\) transforms the cascade graph into a set of diffusion paths according to the diffusion time, each of which depicts the process of information propagation between users within the observation time; however, this method ignores the structural information of cascade graphs.

Our approach samples the cascade graph \( C_i(t) \) to obtain a sequence of sub-cascade graphs \( G_i^T \) which is used to represent cascades within the observation time \( T \). \( G_i^T \) is denoted as:

\[
G_i^T = \left\{ g_{i1}^0, g_{i1}^1, \ldots, g_{im}^{t-1} \right\}, t_j \in [0, T] and \ j \in [1, m].
\]

The sampling process is illustrated in Fig. 3, where each sub-cascade graph is represented by an adjacency matrix; the rows correspond to the alphabetical order of nodes’ labels (top and bottom) and the columns correspond to edges, as illustrated above each instance of the alphabetical order of nodes’ labels. The first sub-cascade graph in \( G_i^T \) only contains one single node because it is the generator of the post \( p_i \), so we add a self-connection for this initiator. Thus, \( G_i^T \) is represented with a sequence of adjacency matrices \( A_i^T = \left\{ a_{i1}^1, a_{i1}^2, \ldots, a_{i1}^{t-1} \right\} \).
B. Laplacian Transformation of Cascades

CasCN is inspired by GCN [41] in the sense of using spectral graph theory to define a convolutional operator. However, classical GCN methods cannot be applied for cascades modeling since they focus on fixed and undirected graphs which, in turn, cannot consider the temporal information of cascade evolution. In contrast, cascade graphs in our problem are dynamic directed graphs (DAGs). As mentioned, the graph Laplacian for an undirected graph is a symmetric difference operator \( L = D - W \), where \( D \) is the degree matrix and \( W \) is the weight matrix of the graph, which cannot be adapted in DAGs.

Recently, Li et al. [46] propose DCRNN to model the traffic flow as a diffusion process on a fixed DAG (a directed sensor graph), and define a diffusion convolution as:

\[
y = g_\theta * \mathcal{G} x = \sum_{k=0}^{K} \left( \theta_{k,1} (D^{-1}_O W)^K + \theta_{k,2} (D^{-1}_T W^T)^K \right) x
\]

where \(-D^{-1}W\) is the random walk matrix used to replace Laplacian \( \tilde{L} \) in Eq.(3). This operation is actually a diffusion process convolution proposed by Atwood and Towsley [52] where the diffusion process is modeled as Markov process and may converge to a stationary distribution \( \mathcal{P} \in \mathbb{R}^{n \times n} \) after many steps, and the \( i^{th} \) row \( \mathcal{P}_{:,i} \in \mathbb{R}^n \) represents the likelihood of diffusion from node \( v_i \).

In our settings, various cascades are different DAGs, all of which require incorporating special structure and direction information rather than a single and fixed sensor network in [46]. To overcome this challenge, we introduce Laplacian of cascade \( \Delta_c \), called CasLaplacian, for modeling the convolution operation over a single cascade signal \( X \) as:

\[
y = g_\theta * \mathcal{G} X = \sum_{k=0}^{K} \theta_k T_k \left( \overline{\Delta_c} \right) X
\]

where \( \overline{\Delta_c} = \frac{2}{\lambda_{max}} \Delta_c - I_N \) is a scaled Laplacian.

Now we introduce the way of computing Laplacian of cascade \( \Delta_c \), which can capture special structural and directional characteristics of different cascades. For a directed graph, we define the normalized directed Laplacian as:

\[
\mathcal{L} = I - \frac{\Phi \Phi^T P \Phi^T \Phi}{2}, \tag{5}
\]

(cf. [53]) where \( P \) is a transition probability matrix, \( \Phi \) is a diagonal matrix with entries \( \Phi(v,v) = \phi(v) \), and \( \phi = [\phi_1]_{1 \leq i \leq n} \) is the column vector of the stationary probabilities distribution of \( P \).
However, such a symmetrical \( L \) can not capture the unique characteristic of the random walk on the different cascades. For example, given a cascade with transition probability matrix \( P_c \), there exist cascades which have the same stationary distribution matrix \( P_c \). Thus, all these cascades have the same Laplacian matrix. To solve this problem, we relied on \textit{Diplacian} [54] which computes Laplacian of DAGs as:

\[
\Gamma = \Phi^{\frac{1}{2}} (I - P) \Phi^{-\frac{1}{2}}
\]

where the transition probability matrix \( P \) was defined as \( P = D^{-1} W \) with the hypothesis that the graph is strongly connected graphs (SCGs) [54]. In contrast, our cascade graphs are not SCGs. Thus, we define a transition probability matrix \( P_c \) of given cascade graph as:

\[
P_c = (1 - \alpha) \frac{E}{n} + \alpha (D^{-1} W),
\]

where \( E \in \mathbb{R}^{n \times n} \) is an all–one matrix and \( \alpha \in (0, 1) \) is an initial probability, used to restrict the state transition matrix \( D^{-1} W \) to be a strongly connected matrix without 0 anywhere. Then the transition matrix \( P_c \) is irreducible, and has a unique stationary probability distribution \( \{ \phi_i \mid \phi_i > 0, 1 \leq i \leq n \} \). The stationary distribution vector \( \{ \phi_i \} \) can be obtained by solving an eigenvalue problem \( \phi^T P_c = \phi^T \) subject to a normalized equation \( \phi^T e = 1 \), where \( e \in \mathbb{R}^n \) is an all-one vector.

Finally, we can compute \textit{CasLaplacian} as:

\[
\Delta_c = \Phi^{\frac{1}{2}} (I - P_c) \Phi^{-\frac{1}{2}}.
\]

**Relationship with GCN:** We now explain the relationship between our directed CasLaplacian and the normalized one in GCN, as well as the rationale behind CasLaplacian.

A random walk on \textit{undirected} graph \( G \) is a Markov chain defined on \( G \) with the transition probability matrix \( P = D^{-1} A \), and there exists a unique stationary distribution \( \{ \phi_1, \phi_2, \ldots, \phi_n \} \). Let \( \phi = [\phi_1]_{1 \leq i \leq n} \) be the column vector of the stationary probabilities, where \( \phi^T P = \phi^T \). Note that, as for undirected graph, the normalized Laplacian \( L \) can be transformed as:

\[
L = D^{-\frac{1}{2}} (D - A) D^{-\frac{1}{2}} = D^{\frac{1}{2}} (I - P) D^{-\frac{1}{2}}.
\]

Also, the \( \phi \) of undirected graph can be calculated as

\[
\phi_i = \frac{d_i}{\sum_k d_k}, \quad i = 1, 2, \ldots, n,
\]

and \( \Phi^{\frac{1}{2}} = \sqrt{\text{diag}(\phi)} \), where \( \phi = [\phi_1, \phi_2, \ldots, \phi_n]^T \), which can be used to approximate degree matrix \( D \):

\[
L = D^{\frac{1}{2}} (I - P) D^{-\frac{1}{2}} \approx \Phi^\frac{1}{2} (I - P) \Phi^{-\frac{1}{2}}.
\]

Algorithm 1 formalizes the process of constructing the Laplacian of cascades.

**Algorithm 1 Laplacian of cascade.**

**Input:** A cascade graph \( G \), initial probability \( \alpha \).

**Output:** \textit{CasLaplacian}–Laplacian of cascade \( \Delta_c \).

1. Compute degree matrix \( D \) and weighted adjacency matrix \( W \) of a cascade graph \( G \).
2. Compute transition probability matrix \( P_c \) of cascade graph according to Eq.(7).
3. Solve the eigenvalue problem \( \phi^T P_c = \phi^T \) subject to a normalized equation \( \phi^T e = 1 \) to get \( \{ \phi_i \} \).
4. \( \Phi = \text{diag}(\phi) \).
5. Compute \textit{CasLaplacian} \( \Delta_c \) according to Eq.(8).

**C. Structural and Temporal Modeling**

We represent and model the cascade graph in a structural-temporal way. After obtaining the adjacency representation of sub-cascade graph sequence \( A^T \) and the Laplacian matrix \( \Delta_c \) for each cascade graph, \textit{CasCN} turns to learn the structural and temporal patterns via the combination of classical LSTM and GCN.

We leverage the RNNs to model the temporal dependence of diffusion – in particular, using the Long Short-Term Memory (LSTM) [47], which is a stable and powerful variant of RNNs. We replace the multiplications by dense matrices \( W \) with graph convolutions to incorporate the structural information:

\[
i_t = \sigma (W_i \ast G X_t + U_i \ast G h_{t-1} + V_i \circ c_{t-1} + b_i) \\
f_t = \sigma (W_f \ast G X_t + U_f \ast G h_{t-1} + V_f \circ c_{t-1} + b_f) \\
o_t = \sigma (W_o \ast G X_t + U_o \ast G h_{t-1} + V_o \circ c_t + b_o)
\]

where \( \ast \) denotes the graph convolution defined in Eq.(4), signal \( X_t \) is the cascade graph sequences \( A^T \in \mathbb{R}^{d_T \times n \times n} \), \( d_T \) denotes the number of diffusion time steps of post \( p_t \). We leverage \( W_i \ast G X_t \) to mean a graph convolution of signal \( X_t \) with \( d_h \times n \) filters which are functions of the graph Laplacian \( L \) parametrized by K Chebyshev coefficients. \( \sigma (\cdot) \) is the logistic sigmoid function and \( i_t, f_t, o_t, b_t \) are respectively the input gate, forget gate, output gate and bias vector. The matrices \( W \in \mathbb{R}^{K \times d_h \times d_h} \), \( U \in \mathbb{R}^{K \times d_h \times d_h} \) and \( V \in \mathbb{R}^{n \times d_h} \) are the different gate parameters, and \( n \) denotes the number of nodes in a cascade graph, and \( d_h \) is the size of cell states.

In particular, the memory cell \( c_t \) is updated by replacing the existing memory unit with a new cell \( c_t \) as:

\[
c_t = f_t \circ c_{t-1} + i_t \circ \tanh (W_c \ast G X_t + U_c \ast G h_{t-1} + b_c)
\]

The hidden state is then updated by

\[
h_t = o_t \circ \tanh (c_t)
\]

tanh (\cdot) refer to the hyperbolic tangent function, and \( \circ \) is the entry-wise product.

**D. Cascades Size Prediction**

Previous works [29], [30] have shown the existence of time decay effect – i.e., that the influence of a node on other nodes will decrease over time. Various time decay functions have been defined: (1) power-low functions \( \phi^T (T) = (T + \epsilon)^{-\alpha} \); (2) exponential functions \( \phi^T (T) = e^{-\theta T} \); (3) Reyleigh functions \( \phi^T (T) = e^{-\frac{1}{2} \theta T^2} \). In practice, the choice of such function varies for different scenarios, e.g.,
exponential functions are suitable for financial data while Reyleigh functions perform better for epidemiology and power law functions are more applicable in geophysics and social networks [55, 56].

However, all the above time-decay functions have the limitation of parametric assumption which is greatly influenced by assumed prior distribution (and intuition). In this paper, we employ a non-parametric way to define the time decay function. More specifically, we assume that the time window of the observed cascade is \([0, T]\), and then split the time window into \(l\) disjoint time intervals \([t_0 = 0, t_1], [t_1, t_2], [t_2, t_3], \ldots, [t_{l-1}, t_l = T]\) to make the continuously time window into discrete. It not only allocates each diffusion time a corresponding interval, but also allows us to learn the discrete variable of time decay effect \(\lambda = \{\lambda_m, m \in (1, 2, \cdots, l)\}\). Therefore, we define a function to compute the corresponding time interval \(m\) of time decay effect for a re-tweet at time \(t\):

\[
m = \lfloor \frac{(t - t_0)}{T/l} \rfloor
\]  

(15)

Where \(t_0\) is the time of original post, \(l\) is the number of time interval, \(\lfloor \cdot \rfloor\) and \(\lceil \cdot \rceil\) are floor and ceiling operation.

For a cascade graph \(C_t(t)\) regarding post \(p_i\) within the observation time window \([0, T]\), we get the hidden states \(\{h_1, h_2, \cdots, h_T\}\) and we multiply a time decay effect \(\lambda_m\) for each hidden state to obtain \(\{h'_1, h'_2, \cdots, h'_T\}\) by:

\[
h'_t = \lambda_m h_t
\]  

(16)

summed up to get the representation vector for the cascade graph \(C_t(t)\):

\[
h'(C_t(t)) = \sum_{t=1}^{d_T} h'_t
\]  

(17)

The last component of CasCN is a multi-layer perceptron (MLP) with one final output unit. Given the representation \(h'(C_t(t))\), we calculate the increment size \(\Delta S_i\) as:

\[
\Delta S_i = f(C_t(t)) = MLP(h'(C_t(t)))
\]  

(18)

Our ultimate task is to predict the increment size for a fixed time interval, which can be done by minimizing the following loss function:

\[
\ell(\Delta S_i, \Delta \tilde{S}_i) = \frac{1}{P} \sum_{i=1}^{P} \left( \log \Delta S_i - \log \Delta \tilde{S}_i \right)^2
\]  

(19)

where \(P\) is the number of posts, \(\Delta S_i\) is the predicted incremental size for post \(p_i\), and \(\Delta \tilde{S}_i\) is the ground truth.

The process of training CasCN is shown in Algorithm 2.

V. EXPERIMENTS

In this section, we compare the performance of our proposed model CasCN with several state-of-the-art approaches that we use as baselines, and a few variants of CasCN itself, for cascade size prediction using two real-world datasets.
publication time before 8 am and after 6 pm being filtered out. Finally, we sort the cascades in terms of their publication time after preprocessing and choose the first 70% of cascades for training and the rest for validation and testing via even split.

- **HEP-PH** (http://snap.stanford.edu/data/cit-HepPh.html): HEP-PH dataset is from the e-print arXiv and covers papers in the period from January 1993 to April 2003 (124 months). If a paper $i$ cites paper $j$, the graph contains a directed edge from $i$ to $j$. The data was originally released as a part of 2003 KDD Cup [57]. For the observation window, we choose $T = 3, 4$ and 5 years corresponding to the year that the popularity reaches about 50%, 60% and 70% of the final size, as shown in Fig. 5(b). Then, we pick up 70% of cascades for training and the rest for validation and testing via even split.

![Distribution of cascades size](image1.png)

**Fig. 4.** Distribution of cascades size, the X axis is the size of cascades, and the Y axis is the number of cascades corresponding to the different sizes.

![Percentage distribution between time and the number of cascades](image2.png)

**Fig. 5.** Percentage distribution between time and the number of cascades

### B. Baselines

In section II, we mentioned that existing relevant methods for information cascade prediction are mainly falling into four categories: (1) Diffusion Model-based approaches, (2) Generative process approaches, (3) Feature-based approaches, and (4) Deep learning-based approaches. Therefore, we select several methods in each group as baselines. For deep learning methods, we select three representative methods in each group as baselines. For deep learning methods, we select three representative methods in each group as baselines. For deep learning methods, we select three representative methods in each group as baselines. For deep learning methods, we select three representative methods in each group as baselines. For deep learning methods, we select three representative methods in each group as baselines.

- **Feature-based**: Recent studies [29], [33], [35], [39] show that structural features, temporal features, and other features (e.g., content features and user features) are informative for information cascade prediction. In our implementations, we include all features mentioned above that could be generalized across datasets. These features include:

  - **Structural features**: We count the number of leaf nodes, the average degree (both in-degree and out-degree), average and max length of retweet path of cascades as measures of structural features.

  - **Temporal features**: We extract the time elapsed since the initial post for each retweet, the cumulative growth and incremental growth every 10 minutes for Sina Weibo and every 31 days for HEP-PH, for the reason that the time in Sina Weibo can be accurate to minutes, and the unit in HEP-PH is a day.

  - **Other features**: We use node ids as node identity feature.

After we extracting all the cascade features, we use two models, i.e., **Feature-linear** and **Feature-deep**, to perform information cascade prediction. The label (incremental size of cascade) has been logarithmically transformed before feeding into models, so that the baseline of feature-based methods optimizes the same loss function as CasCN.

- **Feature-linear**: We feed the features into a linear regression model with $L_2$ regularization, and the details of the $L_2$-coefficient setting can be found in Section V-E.

- **Feature-deep**: For fairness of comparison of the performance of the feature-based approaches with CasCN, we propose a strong baseline denoted as Feature-deep, which also uses a MLP model to predict the incremental size of cascade with hand-craft feature vectors.

- **DeepCas** [8]: The first deep learning architecture for information cascade prediction, which represents a cascade graph as a set of random walk paths and piped through bi-directional GRU neural network with an attention mechanism to predict the size of the cascade. It mainly utilizes the information of structure and node identities for prediction.

- **DeepHawkes** [6]: DeepHawkes model integrates the high prediction power of end-to-end deep learning into interpretable factors of Hawkes process for popularity prediction. The marriage between deep learning technique and a well-established interpretable process for modeling cascade dynamics bridges the gap between prediction and understanding of information cascades. This method belongs to both generative approaches and deep learning-based approaches.

- **Topo-LSTM** [31]: A novel topological recurrent neural network which is a directed acyclic graph-structured (DAG-structured) RNN takes dynamic DAGs as inputs and generates a topology-aware embedding for each node in the DAGs as
outputs. The original application of Topo-LSTM is to predict node activations. We replace the logistic classifier in Topo-LSTM with a diffusion size regressor to predict the size of cascades.

C. Variants of CasCN

In addition to comparison with existing baselines, we also derive a few variants of CasCN:

CasCN-GL: CasCN-GL replaces the structural-temporal modeling component of CasCN with the combination of GCN and LSTM for modeling structural and temporal patterns, respectively.

CasCN-GRU: This method replaces the LSTM of CasCN with GRU. Similar to LSTM, CasCN with GRU models structural-temporal information using extra gating units, but without separated memory cells. Formally, we update the state of \( h_t \) by a linear interpolation between the last state \( h_{t-1} \) and the candidate state \( h_t \).

CasCN-Path: In CasCN-Path, we use random walks to represent a cascade graph (shown in Fig. 6) rather than sub-cascade graphs used in CasCN. Therefore, we first embed users into a 50-dimensional space to represent the latent (re-tweeting) relationships among users in a cascade graph. Next, we use random walks to sample sufficient number of sequences for all cascade graphs. Finally, we feed them to CasCN and predict the size of information cascades.

CasCN-Undirected: In CasCN-Undirected, we regard the cascade graphs as undirected graphs and calculate the normalized Laplacian according to \( L = I - \frac{1}{2}WD^{-\frac{1}{2}} \).

CasCN-Time: In CasCN-Time, we do not consider the time decay effect of re-tweeting.

D. Evaluation Metric

Following the existing works, we choose standard evaluation metrics – MSLE (mean square log-transformed error) in our experiments [6], [8], [31]. Note that the smaller MSLE, the better the prediction performance. Specifically, MSLE is the metric for evaluating the linking accuracy, defined as:

\[
MSLE = \frac{1}{P} \sum_{i=1}^{P} \left( \log \Delta S_i - \log \tilde{S}_i \right)
\]  

(20)

where \( P \) is the total number of posts, \( \Delta S_i \) is the predicted incremental size for post \( p_i \), and \( \tilde{S}_i \) is the ground truth.

E. Hyper-parameter

Models mentioned above involve several hyper-parameters. For example, \( L_2 \) coefficient in Feature-linear are chosen from \( \{1, 0.5, 0.1, 0.05, \ldots, 10^{-3}\} \). For Feature-deep, parameters are similar to deep learning-based approaches.

For LIS, we follow the work in [18], the maximum epoch \( M \) is \( 1 \times 10^5 \). We use random values to initialize regularization parameters \( \gamma_f \) and \( \gamma_S \).

For Node2Vec, we follow the work in [58], i.e., parameters \( p \) and \( q \) are selected from \( \{0.25, 0.50, 1, 2, 4\} \), the length of walk is chosen from \( \{10, 25, 50, 75, 100\} \), and the number of walks per node varies from \( \{5, 10, 15, 20\} \).

For DeepCas, DeepHawkes and Topo-LSTM, we follow the setting of DeepCas [8], where the embedding dimensionality of users is 50, the hidden layer of each GRU has 32 units and the hidden dimensions of the two-layer MLP are 32 and 16, respectively. The learning rate for user embeddings is \( 5 \times 10^{-4} \) and the learning rate for other variables is \( 5 \times 10^{-3} \). The batch size for each iteration is 32 and the training process will stop when the loss of validation set does not decline for 10 consecutive iterations. The time interval of DeepHawkes is set to 10 minutes for Sina Weibo and 2 months for HEP-PH. For CasCN, the basic parameters (e.g., learning rate and batch size, etc.) are the same as above deep learning-based approaches, except that we choose the support \( K = 2 \) of GCN and calculate the max eigenvalue \( \lambda_{\text{max}} \) of cascade Laplacian.

F. Performance comparison

We first report the performance of various methods on cascade size prediction. The results are illustrated in Table III.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Weibo Dataset</th>
<th>HEP-PH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metric</td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Features-linear</td>
<td>3.68</td>
<td>3.361</td>
</tr>
<tr>
<td>LIS</td>
<td>3.731</td>
<td>3.621</td>
</tr>
<tr>
<td>Node2Vec</td>
<td>3.795</td>
<td>3.523</td>
</tr>
<tr>
<td>DeepCas</td>
<td>2.958</td>
<td>2.089</td>
</tr>
<tr>
<td>Topo-LSTM</td>
<td>2.772</td>
<td>2.043</td>
</tr>
<tr>
<td>DeepHawkes</td>
<td>2.441</td>
<td>2.287</td>
</tr>
<tr>
<td>CasCN</td>
<td>2.242</td>
<td>2.036</td>
</tr>
</tbody>
</table>

TABLE IV

OVERALL PERFORMANCE COMPARISON OF INFORMATION CASCADES PREDICTION AMONG DIFFERENT APPROACHES. M: MODEL; T: OBSERVATION TIME WINDOW.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Weibo Dataset</th>
<th>HEP-PH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metric</td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CasCN</td>
<td>2.242</td>
<td>2.036</td>
</tr>
<tr>
<td>CasCN-GRU</td>
<td>2.288</td>
<td>2.052</td>
</tr>
<tr>
<td>CasCN-Path</td>
<td>2.557</td>
<td>2.483</td>
</tr>
<tr>
<td>CasCN-GL</td>
<td>2.312</td>
<td>2.028</td>
</tr>
<tr>
<td>CasCN-Undirected</td>
<td>2.309</td>
<td>2.132</td>
</tr>
<tr>
<td>CasCN-Time</td>
<td>2.652</td>
<td>2.547</td>
</tr>
</tbody>
</table>

TABLE III

OVERALL PERFORMANCE COMPARISON OF INFORMATION CASCADES PREDICTION AMONG DIFFERENT APPROACHES. M: MODEL; T: OBSERVATION TIME WINDOW.
Then we compare the performance of CasCN with a few variants, and the results are shown in Table IV. Lastly, we do some analyses with different parameter settings of CasCN, and the results can be found in Table V. The following sections will describe these empirical observations in details.

**CasCN vs. Baselines:** Table III summarizes the performance comparison among CasCN and baselines on both Sina Weibo dataset and HEP-PH dataset. In general, the proposed CasCN model performs relatively well on information cascade prediction for both datasets (post re-tweeting and paper citing). It outperforms traditional approaches, e.g., feature-based and generative approaches, as well as superior to the state-of-the-art deep learning methods, with a statistically significant drop of MSLE. Now we step into the details of comparison:

The performance gap between these Feature-deep and Feature-linear is quite small meaning that if we have a set of representative features of information cascades, deep learning does not always perform better than traditional predicting methods. However, as discussed earlier, the performance of such methods heavily depend on hand-crafted features which are difficult to select for different scenarios in practice.

For embedding methods, Node2Vec [58] does not perform well. Through the comparison with DeepCas [8], it proves that only taking the node embeddings as the graph representation is not enough and is not comparable with representing the graph as a set of random paths.

DeepCas, as the first proposed end-to-end deep learning method for cascade size prediction, exhibits better performance than feature-based approaches and traditional generative process-based approaches. But it still way worse than other deep learning based approaches, because of failing to consider temporal information and the topological structure of cascade graphs. The latest method Topo-LSTM also lacks time feature, so that it performs a little bit worse than DeepHawkes and our model. DeepHawkes, while successful in modeling cascades in a deep generative way, it does not perform the best due to its weak ability to learn structural information.

Finally, our proposed CasCN model, which purely relies on and fully explores structural and temporal information, significantly outperforms all baselines. For example, it achieves excellent prediction results with MSLE = 1.916 when observing for 3 hours in Sina Weibo and MSLE = 0.851 when observing for 7 years in HEP-PH, respectively. It reduces the prediction error by 15.2% and 30.9% comparing to the second best DeepHawkes.

When comparing methods with different observation window T, we clearly see a general pattern that the larger the T, the easier to make a good prediction. It is mainly because of the fact that longer T reveals more information for prediction.

**CasCN with Variants:** To investigate and demonstrate the effectiveness of each component of our model (e.g., to understand the effect of the sampling part of CasCN), we present five variants of CasCN, where all are built upon the original CasCN model with some components changed. Their details can be found in Section V-C.

The experimental results are shown in Table IV, from which we can see that original CasCN leads to a certain reduction of prediction error when compared with other variants. From the comparison to CasCN-Undirected and CasCN-Time, we find that directionality and time decay effect are proved to be indispensable for cascade size prediction. Similarly, CasCN-Path brings a remarkable decrease of the prediction performance, which tells the necessity and effectiveness of sampling in CasCN. This indicates that the way to sample cascade graph as sub-cascade graph sequence can better reflect the dynamics of the cascade structure and the topological structure of each diffusion time.

In summary, sub-graphs sampling and structural-temporal modeling are critical components in CasCN, both of which essentially improve the performance of information cascade prediction as presented in the results.

<table>
<thead>
<tr>
<th>Table V</th>
<th>Analysis of Parameter Impact on Performance.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset</td>
<td>Weibo Dataset</td>
</tr>
<tr>
<td>Metric</td>
<td>MSLE</td>
</tr>
<tr>
<td>Parameter</td>
<td>1 hour</td>
</tr>
<tr>
<td>K=1</td>
<td>2.284</td>
</tr>
<tr>
<td>K=2</td>
<td>2.242</td>
</tr>
<tr>
<td>K=3</td>
<td>2.312</td>
</tr>
<tr>
<td>λ_{max} = 2</td>
<td>2.418</td>
</tr>
<tr>
<td>λ_{max} = real</td>
<td>2.242</td>
</tr>
</tbody>
</table>

Fig. 7. Loss of CasCN on the validation set with varying K

**Parameter analysis in CasCN:** We now turn to investigating whether the parameters of CasCN have impact on the performance of cascade size prediction. Results are summarized in Table V. We consider two vital parameters of graph convolutional kernel, i.e., Chebyshev coefficients K and the largest eigenvalue λ_{max} of Laplacian matrix Δ_c. For Chebyshev coefficients K we selected from {1, 2, 3}. To obtain λ_{max}, we have two ways: the first is to approximate it as λ_{max} ≈ 2, and the second is to compute the exact value of λ_{max} for each cascade graph. Table V shows that CasCN with K = 2 can achieve better performance than K=1 and 3. And in Fig. 7, the validation loss in each epoch steadily declines. There is no evidence showing that a larger or smaller K is better than a median one. Further, bigger K will increase the computational cost. For the value of λ_{max}, precise values can lead to better prediction results.

We also investigated the performance of CasCN when the observed cascades are small – e.g., the size of the cascades is
within 10, 20. Fig. 8(a) gives the statistics of Weibo dataset illustrating the average cascade size increasing with time. Fig. 8(b) shows that the MSLE results for various size decrease with training epochs. Apparently, the larger of observed cascade size, the lower MSLE value CasCN achieves.

**Discussions on feature learning:** Finally, we discuss and demonstrate the capability of CasCN on feature learning in a visual way. We use the latent representation of each cascade graph $C_i(t)$ to plot a heatmap (as shown in Fig. 9). The value in each dimension corresponds to some implicit or explicit features related to predicting the cascade size. Fig. 9 tells us that the latent representation varies with cascade size. And surprisingly, there exists a clear pattern separation between outbreak (larger cascades) and non-outbreak (smaller) cascades, which indicates that CasCN is able to learn a good latent representation of cascades with different sizes and thus can be applied for outbreaking prediction.

Next, we try to understand/interpret the importance of some hand-crafted features in cascade size prediction. First, we use t-SNE [61] to project the vector representation summarized in $h'(C_i(t))$ to one data point in a 2-D space. Note that cascade graphs with similar vector representations are placed closely. Second, we color each data point (transformed from a cascade graph) using different strategies, such as based on the value of a certain feature $f$ (e.g., number of leaf nodes, mean time, etc.), or the true increment size (the ground-truth label). The distribution of colors suggests a connection between the learned representations and network properties. That is, if a colored plot based on a certain feature $f$ is well correlated with that of the true increment size, this feature is positively useful for cascade size prediction. Take the Weibo dataset as an example: Fig.9(c) and 9(e) have similar color distributions with the true increment size $9(g)$ – we believe that leaf nodes and mean time are two important features for cascade size prediction.

**VI. CONCLUSIONS AND FUTURE WORK**

We proposed a novel deep learning based framework – CasCN – an end-to-end modeling framework for cascade growth prediction that does not rely heavily on feature engineering and can be easily generalized; and enabling the information cascade prediction by exploiting both structural and temporal information. The CasCN model can learn a better latent representation for cascade graph with less information, using structural and temporal information of cascades within a deep learning framework. Incorporating the directionality of cascades and time decay effect further improves the prediction performance. Our experiments conducted on two scenarios, i.e., forecasting the size of re-tweeting cascades in Sina Weibo and predicting the citation of papers in HEP-PH, demonstrate that CasCN outperforms various state-of-the-art methods.

Our future work focuses on three tasks: (1) introducing more sophisticated training techniques (e.g., attention mechanism or batch training), to transform CasCN to inductive model and
model larger graphs; (2) efficient incorporation of updates into LSTM; and (3) investigating the potential coupling CasCN with model-based approaches to improve the accuracy.

Acknowledgments: Work supported by National Natural Science Foundation of China (Grants No.61602097 and No.61472064), NSF Grants III 1213038 and CNS 1646107, ONR grant N00014-14-10215, and the Fundamental Research Funds for the Central Universities (No.ZYGX2015J072).

REFERENCES

[38] B. Shulman, A. Sharma, and D. Cosley, “Predictability of popularity: Gaps between prediction and understanding,” in ICIWSM, 2016.