Scilight

Transactions on Graph Intelligence and Network Applications
https://www.sciltp.com/journals/tgina



Article

Check for updates

Graph-Aware Controlled Continuous Dynamics Network for Information Diffusion Prediction

Jianfu Zheng, Fei Xiong *, Zhiyuan Zhang and Pengcheng Huang

School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China
* Correspondence: xiongf@bjtu.edu.cn

How To Cite: Zheng, J.; Xiong, F.; Zhang, Z.; et al. Graph-Aware Controlled Continuous Dynamics Network for Information Diffusion Prediction. *Transactions on Graph Intelligence and Network Applications* **2026**. https://doi.org/10.53941/tgina.2026.100009

Received: 10 June 2026
Revised: 22 October 2026
Accepted: 28 October 2026
Published: 5 November 2026

Abstract: Information diffusion prediction aims to infer future user activations from observed cascade prefixes in social networks, where diffusion unfolds as a complex, irregular, continuous-time process. Existing methods typically model dynamic user preferences by discretizing diffusion into temporal graph or hypergraph snapshots. As a result, these methods produce hidden states only at discrete timestamps and do not capture the continuous dynamics between cascade events. Although neural ordinary differential equations provide a natural framework for modeling temporal dynamics, their trajectories are determined by the initial condition and cannot incorporate subsequent observations as direct controls along the evolving path. To address these limitations, we propose CoDyNet, a graph-aware Controlled continuous Dynamics Network for microscopic information diffusion prediction. CoDyNet first learns graph-aware user representations from multi-source static relations and dynamic diffusion hypergraphs. It then constructs timestamp-aware control paths that drive a neural controlled differential equation for observation-guided continuous propagation modeling. We further introduce a cascade-aware graph contrastive regularizer that enforces consistency between clean and perturbed static user representations while avoiding co-activated false negatives. The resulting continuous diffusion context and static user context are encoded by causal Transformer blocks and adaptively fused for next-user prediction. Extensive experiments on four real-world datasets show that CoDyNet achieves competitive performance against representative diffusion prediction baselines.

Keywords: information diffusion prediction; social networks; neural controlled differential equations; graph contrastive learning

1. Introduction

Online social platforms have become central channels for disseminating information, opinions, products, and public events [1,2]. Such diffusion supports advertising, recommendation, public opinion monitoring, and social governance, but it also amplifies rumors, hoaxes, and fake news [3]. Accurate information diffusion prediction is therefore important for understanding cascade dynamics, promoting beneficial dissemination, and mitigating harmful content [2,4]. This paper studies the microscopic setting: given an observed cascade prefix, the model predicts the next user to be activated [4].

Early diffusion predictors rely on feature-based or generative models [5], whereas recent methods increasingly use neural sequence models [6–9] and graph neural network models [10,11]. Sequence models capture dependencies in activation order, while graph neural network models encode social links, repost relations, heterogeneous diffusion signals, and high-order co-activation patterns. Dynamic graph and dynamic hypergraph methods further discretize diffusion histories into temporal snapshots to represent time-varying propagation structures. However, a cascade propagation process is not adequately characterized by either an ordered list of activated users or a set of discrete structural snapshots. Instead, it exhibits propagation dynamics that evolve continuously over irregular inter-event intervals, where elapsed time between adjacent events can affect user influence, interest responses, and subsequent diffusion trends [12–14].

Copyright: © 2026 by the authors. This is an open access article under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).
Publisher's Note: Scilight stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Neural ordinary differential equations provide a natural framework for representing continuous hidden dynamics [12], but an autonomous ODE trajectory is primarily determined by its initial state and vector field. In microscopic diffusion prediction, each newly observed user and timestamp constitutes an external signal that may redirect the cascade. Accordingly, the latent diffusion state should be continuously driven by observed cascade events rather than evolving solely from the initial condition. Neural controlled differential equations (NCDEs) provide this capability by treating irregularly sampled observations as a continuous control path that drives latent state evolution [13,15]. At the same time, reliable next-user prediction requires graph-aware user representations. Multi-source social and diffusion graphs provide useful structural priors for modeling user interactions and propagation dependencies. However, these graphs are often sparse, noisy, and semantically heterogeneous, which can weaken the learned user representations.

To address these challenges, we propose CoDyNet, a graph-aware controlled continuous dynamics network for next-user information diffusion prediction. The key idea is to formulate microscopic diffusion prediction as observation-controlled continuous cascade modeling: timestamp-aware diffusion observations are interpolated into a control path, and an NCDE uses this path to evolve latent propagation states between irregular cascade events. To complement this continuous dynamics module, CoDyNet learns long-term user priors from multi-source static relations, captures high-order co-diffusion patterns through dynamic hypergraph message passing, and fuses continuous diffusion context with static user context by using causal context encoders and an adaptive gate. CoDyNet further introduces a cascade-aware graph contrastive regularizer designed to reduce the sensitivity of static user representations to graph perturbations.

The main contributions of this work are summarized as follows:

- We formulate microscopic information diffusion prediction as graph-aware observation-controlled continuous dynamics modeling, and we develop an NCDE-based diffusion module that uses timestamp-aware cascade observations to drive latent propagation states under irregular event times.
- We design a graph-aware user representation framework that combines multi-source static relations with dynamic diffusion hypergraphs, enabling the model to capture both long-term user priors and high-order co-diffusion patterns.
- We introduce cascade-aware graph contrastive regularization that aligns clean and perturbed static user representations while excluding co-activated users from the negative set, and we validate CoDyNet on four real-world diffusion datasets.

2. Related Work

2.1. Information Diffusion Prediction

Existing studies on information cascade prediction can be broadly grouped into three categories: feature-engineering methods, generative diffusion models, and deep-learning-based methods.

Feature-engineering methods. Feature-engineering methods cast cascade prediction as a classification or regression task by manually extracting content semantics, user profiles, network topology, and temporal diffusion signals. Representative studies analyze user and content factors that affect retweet behavior [16], identify early cascade growth and structural properties as strong predictors of final popularity [2], or combine engineered cascade features with point-process signals for popularity prediction [5]. These methods are interpretable and easy to implement, but their performance depends heavily on the quality of hand-crafted features and platform-specific heuristics, which limits their generalization across social media scenarios.

Generative diffusion models. Generative methods regard an information cascade as a continuous-time event sequence and estimate the parameters of a predefined diffusion process from early observations. For example, behavioral dynamics models connect individual action times with macro-level cascade growth [17], Hawkes intensity processes capture the self-exciting nature of social media popularity [18], and joint point-process models further couple information diffusion with network evolution [19]. These methods provide clear mathematical explanations of diffusion mechanisms, but their effectiveness relies on strong assumptions about the underlying process. Such assumptions are difficult to specify in dynamic and heterogeneous social networks.

Deep learning methods. Deep learning methods learn cascade representations directly from data and have become mainstream approaches for diffusion prediction. Sequence-based methods regard an observed cascade as an ordered user or event sequence. DeepCas [6] samples random-walk paths from cascades to learn sequence representations, whereas DeepHawkes [7] incorporates Hawkes-process intuitions into recurrent modeling. FOREST [9] further combines GRUs with reinforcement learning for joint microscopic diffusion prediction and macroscopic popularity prediction. Attention mechanisms are also used to assign different weights to historical activations, and GRASS [20] introduces attention into GRU-based sequence modeling to learn spatio-temporal

patterns from chain-like cascades. These methods effectively model activation-order dependencies, but they usually encode structural contexts only indirectly. Graph-based methods address this limitation by representing cascades with social graphs, repost graphs, heterogeneous diffusion graphs, or hypergraphs. Inf-VAE [21] integrates social homophily and influence with a variational graph autoencoder, whereas DyHGCN [10] learns neighbor influence and dynamic user preferences on heterogeneous diffusion graphs. MS-HGAT [11] combines sequential hypergraphs with static social relations to capture dynamic interaction preferences. More recently, RotDiff [22] represents complex diffusion patterns with hyperbolic rotations, GODEN [14] models user dynamics on heterogeneous diffusion graphs with neural ODE-based graph neural networks, and PMRCA [23] introduces structural and clustering contrastive learning to model multi-source resonance and cognitive adaptation in diffusion patterns.

Despite their effectiveness, existing dynamic graph and hypergraph models usually summarize diffusion states from discrete structural snapshots. This design makes it difficult to model continuous-time dynamics under irregular activations and to describe how newly observed events control the diffusion process.

2.2. Graph Contrastive Learning

The idea of contrastive learning has long supported traditional network embedding methods, where observed structural contexts are pulled together and sampled negative contexts are pushed apart to learn node representations [24–26]. Modern graph contrastive learning extends this principle by constructing semantically related graph views and enforcing representation consistency, thereby alleviating sparsity, noise, and insufficient supervision in graph data. Representative methods contrast local–global summaries or augmented graph views, and further adapt augmentations to preserve graph semantics [27–29]. Related self-supervised methods in recommendation also improve user representations through structural perturbation or embedding noise [30–32]. Inspired by these studies, we introduce graph contrastive learning into social diffusion modeling to enhance structural user representations by enforcing consistency across different social views. This regularization mitigates noise in social graphs and improves diffusion prediction.

2.3. Neural Controlled Differential Equations

Neural ordinary differential equations generalize discrete neural networks to continuous time by using a neural network f_θ to parameterize the vector field of hidden-state dynamics. Given an initial state $\mathbf{h}(t_0)$, the hidden state at time t is defined as the solution to the following initial value problem:

$$\frac{d\mathbf{h}(t)}{dt} = f_\theta(\mathbf{h}(t), t), \quad \mathbf{h}(t) = \mathbf{h}(t_0) + \int_{t_0}^{t} f_\theta(\mathbf{h}(\tau), \tau)\, d\tau. \tag{1}$$

Here, f_θ specifies the instantaneous derivative of $\mathbf{h}(t)$, and the integral form describes the accumulated change from t_0 to t. This formulation can capture continuous dynamics between observed events [12]. However, neural ODE trajectories are primarily determined by the initial state and cannot continuously adjust their evolution using subsequent observations. Figure 1 contrasts discrete observation updates with continuous data-driven dynamics.

Neural controlled differential equations (NCDEs) [13] address this limitation by interpolating discrete observations into a continuous control path $\mathbf{X}(t)$ and using this path to drive hidden-state evolution:

$$\mathbf{z}(t) = \mathbf{z}(t_0) + \int_{t_0}^{t} f_\theta(\mathbf{z}(\tau))\, d\mathbf{X}(\tau). \tag{2}$$

In this equation, $\mathbf{X}(t)$ denotes the continuous signal constructed from the observation sequence, and $\mathbf{z}(t)$ denotes the latent state driven by this signal. Since the control path modulates the latent state over time, NCDEs are well suited for partially observed and irregularly sampled multivariate time series [13]. Later studies introduce causal interpolation for online prediction [15] and attentive NCDEs that use dual NCDE modules to generate attention weights and task-specific hidden states [33]. In information diffusion prediction, cascade events naturally arrive at irregular continuous times. NCDEs therefore provide an effective tool for modeling propagation dynamics between adjacent events and dynamically updating user or cascade states as new observations become available.

3. Problem Formulation

Let V be the set of N users, and let $\mathcal{C} = \{c_1, \ldots, c_M\}$ be a collection of information cascades. A cascade c_m is an activation sequence $((u_1^m, t_1^m), \ldots, (u_{L_m}^m, t_{L_m}^m))$, where $u_i^m \in V$ is the i-th activated user, t_i^m is the activation timestamp, L_m is the cascade length, and $t_1^m < \cdots < t_{L_m}^m$.

To characterize diffusion from multiple relational views, we use a social graph $G_s = (V, E_s)$ for explicit

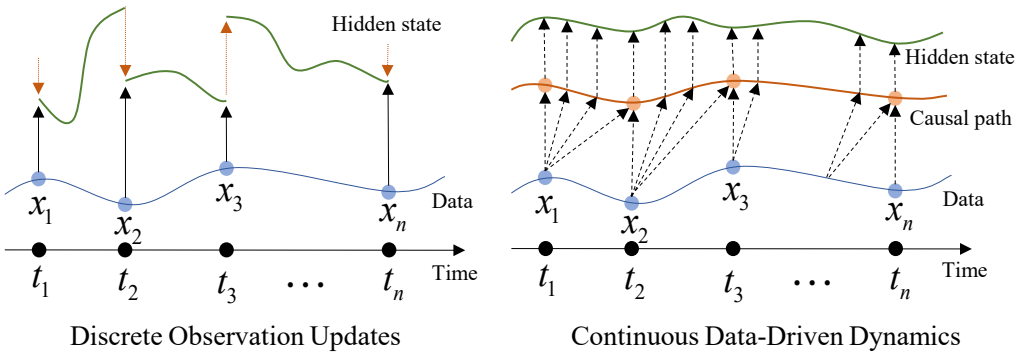


Figure 1. Comparison between discrete observation updates and continuous data-driven dynamics.

user relations and a diffusion graph $G_d = (V, E_d)$ derived from historical activation orders. In G_d, consecutive activations form edges: $A \to B \to C$ gives (A, B) and (B, C), without adding (A, C). Repeated pairs are kept as separate entries for GCN aggregation; no occurrence-count weights are assigned. We also use a user–cascade bipartite graph $G_b = (V \cup \mathcal{C}_{\text{train}}, E_b)$ linking users to the training cascades in which they appear, and a dynamic propagation hypergraph sequence $\mathcal{H} = \{\mathcal{H}_1, \ldots, \mathcal{H}_K\}$, where each $\mathcal{H}_k$ contains cascade hyperedges $h_{m,k} = \{u_i^m \mid (u_i^m, t_i^m) \in c_m,\ t_i^m \in \Delta_k\}$ formed by users activated in the same training cascade within the global time interval Δ_k. The diffusion graph, bipartite graph, and dynamic diffusion hypergraph are constructed only from training cascades to avoid information leakage. We denote these structural views collectively by $\mathcal{G} = \{G_s, G_d, G_b, \mathcal{H}\}$.

Given a prefix $c_m^{1:i} = ((u_1^m, t_1^m), \ldots, (u_i^m, t_i^m))$ with $1 \le i < L_m$, our goal is to predict the next activated user u_{i+1}^m rather than its exact activation time. A model parameterized by Θ outputs a probability distribution $\mathbf{p}_{m,i} = P_\Theta(\cdot \mid c_m^{1:i}, \mathcal{G})$ over all users, where $\mathbf{p}_{m,i}[u]$ indicates the probability that user u will be activated next under the current cascade prefix and structural views. The model selects the user with the highest predicted probability as the next activation. During training, the model is optimized to assign a high probability to the true next activated user u_{i+1}^m.

4. Method

In this section, we present the design of CoDyNet through three main methodological modules: (1) Graph-Aware User Representation Learning; (2) Controlled Cascade Context Modeling; and (3) Next-User Prediction. Figure 2 illustrates the overall workflow.

The three modules are organized to play complementary roles:

(1) Graph-Aware User Representation Learning constructs user representations from multiple structural views. It encodes stable user priors with a unified relation graph and captures high-order co-activation patterns with dynamic diffusion hypergraphs. We further apply cascade-aware graph contrastive regularization to align clean and perturbed graph views, thereby regularizing static user representations derived from heterogeneous graph structures.

(2) Controlled Cascade Context Modeling focuses on the observed cascade prefix. It retrieves graph-aware user representations, augments them with positional and temporal interval information, and uses controlled continuous dynamics to model how cascade context evolves over irregular event times.

(3) Next-User Prediction adaptively fuses the graph-aware diffusion context and the static graph context. The fused representation is then used to score all candidate users and predict the next activation.

4.1. Graph-Aware User Representation Learning

4.1.1. Static User Encoding

This module provides stable user priors for early cascade prefixes, which often contain only a few observed activations. Such priors help the model rank plausible future users when cascade-local evidence is sparse. User interests and forwarding behaviors are strongly correlated with long-term social relations. Intuitively, users tend to forward information that matches their interests; if a user is interested in posts from another user, the user is likely to follow that account and may further repost information from it. Therefore, following relations in social media provide important structural priors for diffusion prediction. Beyond explicit social links, historical activation orders and user–cascade participation also reveal stable diffusion preferences. Based on the relational graphs defined in Section 3, we construct a unified relation graph $G_r = (V_r, E_r)$, where $V_r = V \cup \mathcal{C}_{\text{train}}$ and $E_r = E_s \cup E_d \cup E_b$.

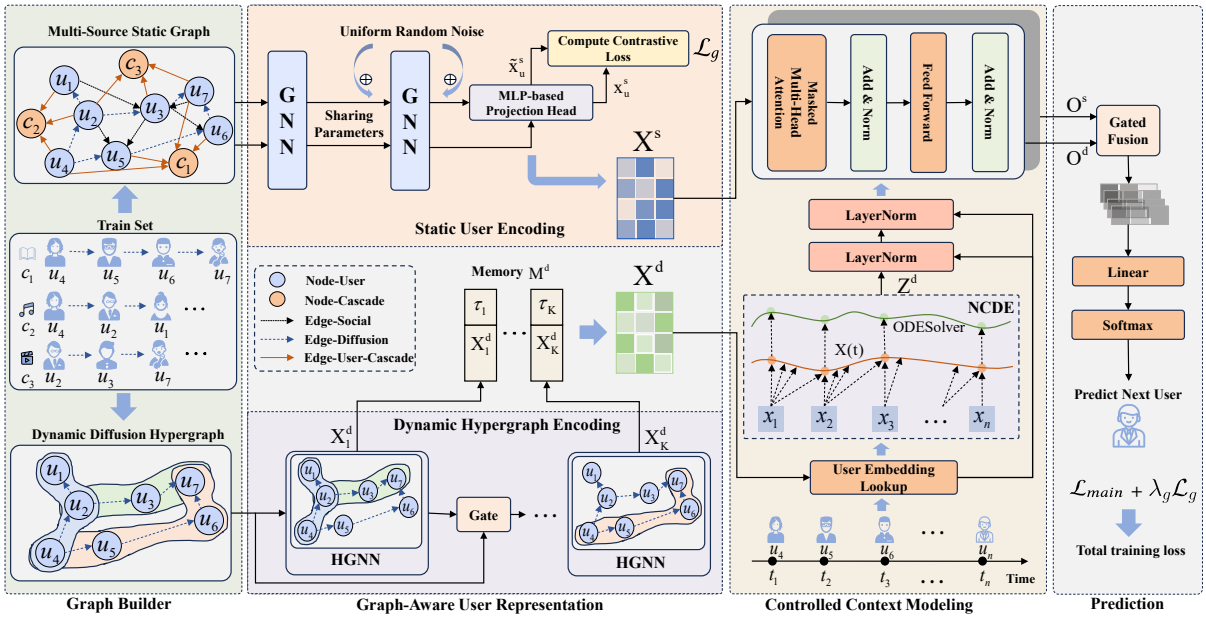


Figure 2. Overall architecture of CoDyNet. The framework contains four modules: Graph Builder, Graph-Aware User Representation, Controlled Context Modeling, and Prediction.

To extract the structural dependencies encoded in G_r, we apply a GCN to aggregate neighborhood information over the unified graph. Let $\mathbf{A}_r \in \mathbb{R}^{|V_r|\times|V_r|}$ be the adjacency matrix of G_r, $\tilde{\mathbf{A}}_r = \mathbf{A}_r + \mathbf{I}_{|V_r|}$ be the self-loop augmented adjacency matrix, and $\tilde{\mathbf{D}}_r$ be the corresponding degree matrix. The layer-wise propagation rule is

$$\mathbf{X}^{(l+1)} = \sigma\left(\tilde{\mathbf{D}}_r^{-\frac{1}{2}}\tilde{\mathbf{A}}_r\tilde{\mathbf{D}}_r^{-\frac{1}{2}}\mathbf{X}^{(l)}\mathbf{W}^{(l)}\right), \tag{3}$$

where $\mathbf{X}^{(l)} \in \mathbb{R}^{|V_r|\times d_l}$ is the input node embedding at layer l, $\mathbf{W}^{(l)} \in \mathbb{R}^{d_l\times d_{l+1}}$ is a learnable parameter matrix, and $\sigma(\cdot)$ denotes the ReLU activation. The initial embedding $\mathbf{X}^{(0)}$ is randomly initialized from a normal distribution. By stacking two GCN layers, we obtain $\mathbf{X}^{(2)} \in \mathbb{R}^{|V_r|\times d}$ and extract the embeddings corresponding to user nodes as the static user representation $\mathbf{X}^s \in \mathbb{R}^{N\times d}$.

4.1.2. Dynamic Diffusion Hypergraph Encoding

Static user representations provide long-term relational priors, but they are time-invariant and cannot describe how the diffusion roles of users change during propagation. To model temporal high-order diffusion dependencies, we construct a dynamic propagation hypergraph sequence $\mathcal{H} = \{\mathcal{H}_1, \ldots, \mathcal{H}_K\}$. Each snapshot $\mathcal{H}_k = (V, \mathcal{E}_k)$ corresponds to a global time interval Δ_k. For cascade c_m, we create a hyperedge $h_{m,k}$ by connecting users activated in c_m within Δ_k. Thus, each hyperedge represents a group-level co-diffusion pattern at a specific propagation period, allowing the model to preserve dependencies beyond pairwise links. All slice boundaries are computed from training events only. We sort their timestamps and divide the overall time span of the training events into K intervals of equal wall-clock duration. We use $K = 8$ by default because it achieves the best overall performance in the temporal-slice sensitivity study while maintaining a suitable balance between temporal resolution and per-snapshot connectivity; Section 6.4 examines both the slice count and the slicing strategy. Let $\mathbf{B}_k \in \{0,1\}^{N\times|\mathcal{E}_k|}$ be the incidence matrix of $\mathcal{H}_k$, where $\mathbf{B}_k[u,e] = 1$ if user u belongs to hyperedge $e \in \mathcal{E}_k$ and 0 otherwise. We encode each snapshot with a Hypergraph Neural Network (HGNN). Starting from the static representation $\mathbf{X}_0^d = \mathbf{X}^s$, the HGNN progressively learns time-indexed diffusion representations $\{\mathbf{X}_1^d, \ldots, \mathbf{X}_K^d\}$. For each snapshot, the update proceeds in two stages. First, vertex information is aggregated to hyperedges:

$$\mathbf{E}_k^h = \sigma\left(\mathrm{softmax}_v(\mathbf{B}_k^\top)\mathbf{X}_{k-1}^d\mathbf{W}_v\right), \tag{4}$$

where $\sigma(\cdot)$ denotes the ReLU activation. Then, hyperedge messages are propagated back to users:

$$\widehat{\mathbf{X}}_k^d = \sigma\left(\mathrm{softmax}_e(\mathbf{B}_k)\mathbf{E}_k^h\mathbf{W}_e\right). \tag{5}$$

The updated user representation is obtained through gated fusion of the previous representation and the current hypergraph-updated representation:

$$\begin{aligned}\mathbf{X}_k^d &= \boldsymbol{\alpha}_k \odot \widehat{\mathbf{X}}_k^d + (1-\boldsymbol{\alpha}_k) \odot \mathbf{X}_{k-1}^d, \\ \boldsymbol{\alpha}_k &= \frac{\exp\left(\sigma(\widehat{\mathbf{X}}_k^d \mathbf{W}_{F1})\mathbf{w}_{F2}\right)}{\exp\left(\sigma(\widehat{\mathbf{X}}_k^d \mathbf{W}_{F1})\mathbf{w}_{F2}\right) + \exp\left(\sigma(\mathbf{X}_{k-1}^d \mathbf{W}_{F1})\mathbf{w}_{F2}\right)},\end{aligned} \tag{6}$$

where $\boldsymbol{\alpha}_k$ adaptively balances the current hypergraph-updated representation and the previous user state. After processing all snapshots, the time-indexed user diffusion states $\{\mathbf{X}_1^d, \ldots, \mathbf{X}_K^d\}$ are stored as a diffusion memory $\mathcal{M}^d = \{(\tau_k, \mathbf{X}_k^d)\}_{k=1}^K$, where τ_k denotes the right boundary of the k-th snapshot. For each repost event in a cascade prefix, its timestamp is used to retrieve the corresponding user embedding from the memory, which forms the diffusion sequence for subsequent cascade context modeling.

Memory lookup uses the most recent snapshot whose right boundary strictly precedes the event timestamp, with $\mathbf{X}_0^d = \mathbf{X}^s$ as the fallback state. We define

$$k(t) = \max\big(\{0\} \cup \{k : \tau_k < t\}\big), \qquad \mathrm{Lookup}_{\mathcal{M}^d}(u,t) = \mathbf{X}_{k(t)}^d[u].$$

Thus, events with $t \le \tau_1$ use the fallback state, events with $\tau_k < t \le \tau_{k+1}$ use $\mathbf{X}_k^d$, and events after τ_K use $\mathbf{X}_K^d$. An event exactly at τ_k uses $\mathbf{X}_{k-1}^d$. This also specifies the behavior before the first boundary and after the final boundary.

4.1.3. Contrastive Graph Structure Regularization

The unified relation graph G_r integrates social, diffusion, and user–cascade links, but these heterogeneous relations may contain mixed semantics and noisy edges. If training relies only on the supervised next-user prediction loss, the static encoder may overfit noisy or spurious relations in G_r, while ordinary contrastive learning may separate users that are likely to participate in the same cascade. To improve robustness, we introduce a cascade-aware contrastive regularizer on the static user representation $\mathbf{X}^s$. Specifically, a perturbed view is generated by injecting a normalized random perturbation into the GCN representation:

$$\tilde{\mathbf{x}}_u^s = \mathbf{x}_u^s + \epsilon_g \,\mathrm{sign}(\mathbf{x}_u^s) \odot \frac{\mathbf{r}_u}{\|\mathbf{r}_u\|_2}, \tag{7}$$

where $\mathbf{x}_u^s$ is the row of $\mathbf{X}^s$ for user u, $\mathbf{r}_u$ is a random vector with independent entries sampled from $\mathrm{Uniform}(0,1)$, ϵ_g controls the perturbation magnitude, $\mathrm{sign}(\cdot)$ is the element-wise sign function, and $\odot$ denotes element-wise multiplication. The perturbed representation is denoted as $\tilde{\mathbf{x}}_u^s$. The perturbed forward pass applies this XSimGCL-style operation after each of the two GCN propagation layers, whereas the clean pass receives no perturbation. The perturbed pass reuses the BatchNorm statistics of the clean pass without updating them. Before computing the contrastive objective, both views are passed through the same MLP-based projection head $q_\phi(\cdot)$, producing $\mathbf{p}_u = q_\phi(\mathbf{x}_u^s)$ and $\tilde{\mathbf{p}}_u = q_\phi(\tilde{\mathbf{x}}_u^s)$. The projection head is implemented as a two-layer MLP whose parameters ϕ are shared by the clean and perturbed branches. For an anchor user u, the same user under the clean and perturbed views is treated as a positive pair, and the InfoNCE loss [34] is:

$$\ell_u^g = -\log \frac{\exp(\mathrm{sim}(\mathbf{p}_u, \tilde{\mathbf{p}}_u)/\tau)}{\sum_{v \in \{u\} \cup \mathcal{N}^-(u)} \exp(\mathrm{sim}(\mathbf{p}_u, \tilde{\mathbf{p}}_v)/\tau)}, \tag{8}$$

where τ is the temperature and $\mathcal{N}^-(u)$ denotes the negative set. The projection head is used only for contrastive regularization. To avoid false negatives, users that co-occur with the anchor in the same training cascade are filtered out from $\mathcal{N}^-(u)$. We use cosine similarity, implemented as the dot product of the L2-normalized projection vectors:

$$\mathrm{sim}(\mathbf{a}, \mathbf{b}) = \left\langle \frac{\mathbf{a}}{\max(\|\mathbf{a}\|_2, \delta)}, \frac{\mathbf{b}}{\max(\|\mathbf{b}\|_2, \delta)} \right\rangle, \qquad \delta = 10^{-12},$$

where δ is a numerical safeguard for normalization. Co-activated users are unsuitable negatives because participation in the same diffusion process indicates related interests or propagation roles. We therefore exclude every co-occurrence observed in the training cascades, without consulting validation or test cascades. The final graph contrastive loss averages the above loss over the sampled anchor set $\mathcal{A}$: $\mathcal{L}_g = \frac{1}{|\mathcal{A}|} \sum_{u \in \mathcal{A}} \ell_u^g$. In implementation, we average the clean-to-perturbed and perturbed-to-clean objectives. Each mini-batch supplies at most 500 distinct anchors, and each anchor additionally uses 128 globally sampled users after the same cascade-aware exclusion. This consistency constraint enforces agreement between the clean and perturbed views of static user representations, thereby reducing sensitivity to the sampled perturbation and supplying more stable inputs for subsequent dynamic hypergraph encoding and NCDE-based evolution modeling.

4.2. Controlled Cascade Context Modeling

4.2.1. Irregular-Time Control Path Construction

This module addresses irregular-time observation modeling. Treating timestamps only as discrete feature embeddings indicates when events occur, but does not allow each observation to steer a continuous latent trajectory. For clarity, we describe one supplied input sequence in this subsection and omit the cascade index. Given $c^{1:T} = ((u_1, t_1), \ldots, (u_T, t_T))$, the NCDE receives a stream of event-level diffusion observations rather than raw user identifiers. Each event first queries the dynamic diffusion memory $\mathcal{M}^d$ with its repost timestamp to obtain the user embedding learned from the corresponding time-sliced diffusion hypergraph:

$$\mathbf{x}_i^d = \mathrm{Lookup}_{\mathcal{M}^d}(u_i, t_i), \quad \mathbf{X}^d = [\mathbf{x}_1^d, \ldots, \mathbf{x}_T^d]. \tag{9}$$

In this section, $\mathbf{X}^d$ denotes the cascade-level diffusion sequence, while snapshot-level user representations are explicitly indexed as $\mathbf{X}_k^d$. Information diffusion events are usually observed at irregular timestamps, and the interval between adjacent activations reflects the local diffusion pace. To expose this temporal information to the continuous dynamics module, we append the normalized repost time to each diffusion observation:

$$\boldsymbol{\xi}_i = [\bar{t}_i; \mathrm{LN}(\mathbf{x}_i^d)] \in \mathbb{R}^{d+1}, \tag{10}$$

where $\mathrm{LN}(\cdot)$ denotes layer normalization. For each supplied input sequence, normalization uses the known range between its maximum and minimum valid timestamps. Timestamps are shifted by the minimum and divided by this range, with a minimum denominator of one; padding and EOS are excluded from the extrema. The result is scaled to the input tensor length minus one, including padding. This normalization is performed once per sequence to construct the control path used for position-wise training predictions. These augmented observations are treated as knots of a continuous control path $\mathbf{X}(t) : [\bar{t}_1, \bar{t}_T] \to \mathbb{R}^{d+1}$, rather than as a sequence of isolated tokens. The path satisfies $\mathbf{X}(\bar{t}_i) = \boldsymbol{\xi}_i$ and is constructed by Hermite cubic interpolation with backward differences:

$$\mathbf{X}(t) = \mathrm{HermiteCubic}\left(\{\boldsymbol{\xi}_i\}_{i=1}^T\right). \tag{11}$$

Here, $\mathrm{HermiteCubic}(\cdot)$ denotes Hermite cubic interpolation with backward differences, which supports causal evaluation at observation points when the input knots are prefix-determined. The resulting path is constructed from the time-aware hypergraph embeddings along the cascade. As illustrated by the right panel of Figure 1, the discrete augmented observations are connected into a continuous control path that evolves between irregular cascade events. Its increment $d\mathbf{X}(t)$ carries both changes in diffusion representations and changes in elapsed repost time, and serves as the driving signal for continuous-time diffusion dynamics. Sequence-level affine normalization preserves event ordering and relative spacing between event timestamps.

4.2.2. Continuous Diffusion Dynamics

Although dynamic diffusion hypergraphs capture stage-level dynamics, they remain discrete snapshots. Without continuous modeling, the diffusion branch changes only at retrieved snapshot states; with an autonomous ODE, later cascade observations cannot directly redirect the trajectory. We therefore use an NCDE over the diffusion branch. Unlike a neural ODE whose trajectory is determined by the initial state and an autonomous vector field, the hidden state of an NCDE is continuously driven by the observed control path. Given $\mathbf{X}(t)$, the initial hidden state is computed from the first control value as $\mathbf{z}(\bar{t}_1) = \mathbf{W}_0\mathbf{X}(\bar{t}_1) + \mathbf{b}_0$. Let $f_\theta : \mathbb{R}^d \to \mathbb{R}^{d\times(d+1)}$ be a neural vector field that maps the current hidden state to a linear transformation over the control increment. In our experiments, f_θ is implemented by CDEFUNC, a time-independent two-layer ReLU MLP with a final $\tanh(\cdot)$ nonlinearity:

$$f_\theta(\mathbf{z}) = \tanh(\mathrm{MLP}_\theta(\mathbf{z})). \tag{12}$$

The resulting vector field converts changes in all control channels, including normalized time and diffusion features, into changes in the latent diffusion state. Since the interpolated path $\mathbf{X}(t)$ is differentiable, evaluating the NCDE is straightforward. Specifically, we define the induced ODE vector field $g_{\theta,\mathbf{X}}$ as

$$g_{\theta,\mathbf{X}}(\mathbf{z}, s) = f_\theta(\mathbf{z})\frac{d\mathbf{X}}{ds}(s). \tag{13}$$

Then for $t \in [\bar{t}_1, \bar{t}_T]$, the NCDE can be rewritten as

$$\begin{aligned}\mathbf{z}(t) &= \mathbf{z}(\bar{t}_1) + \int_{\bar{t}_1}^{t} f_\theta(\mathbf{z}(s))\, d\mathbf{X}(s) \\ &= \mathbf{z}(\bar{t}_1) + \int_{\bar{t}_1}^{t} f_\theta(\mathbf{z}(s)) \frac{d\mathbf{X}}{ds}(s)\, ds \\ &= \mathbf{z}(\bar{t}_1) + \int_{\bar{t}_1}^{t} g_{\theta,\mathbf{X}}(\mathbf{z}(s), s)\, ds,\end{aligned} \tag{14}$$

and can therefore be solved using a standard ODE solver over the interpolated control path. The hidden states at the observed event times are obtained by

$$\mathbf{Z}^d = \mathrm{ODESolve}\left(g_{\theta,\mathbf{X}}, \mathbf{z}(\bar{t}_1), [\bar{t}_1, \ldots, \bar{t}_T]\right). \tag{15}$$

This formulation allows the latent diffusion state to change according to both its previous state and the irregular timestamp and diffusion representation observed along the path. The NCDE output sequence $\mathbf{Z}^d$ is then residually fused with the original diffusion sequence:

$$\mathbf{H}^d = \mathrm{LN}_2\left(\mathbf{X}^d + \mathrm{LN}_1(\mathbf{X}^d + \mathbf{Z}^d)\right). \tag{16}$$

Thus, the diffusion branch preserves the discrete hypergraph signal while being adjusted by continuous-time propagation dynamics.

4.2.3. Context Encoding

The model constructs two contextual input sequences and aggregates event representations using causal attention. The diffusion branch uses the controlled diffusion representation $\mathbf{H}^d$, while the static branch retrieves the corresponding static user representation from $\mathbf{X}^s$, denoted as $\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_T]$ with $\mathbf{s}_i = \mathbf{x}^s_{u_i}$. Both branches are augmented with learnable positional embeddings $\mathbf{p}_i$:

$$\mathbf{R}^d_i = [\mathbf{h}^d_i; \mathbf{p}_i], \quad \mathbf{R}^s_i = [\mathbf{s}_i; \mathbf{p}_i]. \tag{17}$$

In the default implementation, these inputs are further augmented with features derived from the interval to the next event. Two independent causal Transformer encoders are then used to capture dependencies among permitted input positions. For a branch $r \in \{d, s\}$, the masked multi-head attention is computed as

$$\begin{aligned}&\mathrm{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}; \mathbf{M}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_h}} + \mathbf{M}\right)\mathbf{V}, \\ &\mathbf{A}^r_a = \mathrm{Attn}\left(\mathbf{R}^r\mathbf{W}^{Q,r}_a, \mathbf{R}^r\mathbf{W}^{K,r}_a, \mathbf{R}^r\mathbf{W}^{V,r}_a; \mathbf{M}\right), \\ &\mathbf{O}^r = [\mathbf{A}^r_1; \ldots; \mathbf{A}^r_{H_a}]\mathbf{W}^{O,r}, \quad r \in \{d, s\},\end{aligned} \tag{18}$$

where d_h is the dimension of each attention head and $\mathbf{M}$ contains the causal and padding masks. Here $\mathbf{O}^d$ and $\mathbf{O}^s$ denote the contextualized diffusion and static representations, respectively. The causal attention mask ensures that each position attends only to itself and preceding positions.

4.2.4. Adaptive Gated Fusion

The diffusion branch focuses on the current propagation process, while the static branch preserves long-term relational preferences. A constant concatenation or averaging rule would force every cascade and every prefix position to use the same balance between dynamic propagation evidence and static user priors. Their relative importance may vary across cascades and prefix positions, so we use an adaptive gate. For each position, the gate is computed from the concatenation of the diffusion and static contexts:

$$\gamma_i = \mathrm{sigmoid}\left(\mathbf{w}_g^\top[\mathbf{O}^d_i; \mathbf{O}^s_i] + b_g\right), \tag{19}$$

where $\gamma_i \in (0, 1)$ is a scalar gate, and $\mathbf{w}_g$ and b_g are learnable parameters. The final cascade-context representation is

$$\mathbf{o}_i = \gamma_i\mathbf{O}^d_i + (1 - \gamma_i)\mathbf{O}^s_i. \tag{20}$$

This gate allows the model to dynamically emphasize continuous diffusion dynamics or static user priors according to the current prediction context.

4.3. Next-User Prediction

For cascade c_m, the model obtains a fused context representation $\mathbf{o}_{m,i}$ at each prefix position i, which summarizes the observed cascade context before predicting u_{i+1}^m. This representation is fed into a linear classifier followed by softmax normalization:

$$\mathbf{p}_{m,i} = \mathrm{softmax}\left(\mathbf{W}_p \mathbf{o}_{m,i} + \mathbf{b}_p\right), \quad \mathbf{p}_{m,i} \in \mathbb{R}^{N+1}, \tag{21}$$

where $N = |V|$, $\mathbf{W}_p \in \mathbb{R}^{(N+1)\times d_o}$ and $\mathbf{b}_p \in \mathbb{R}^{N+1}$ are learnable parameters, and d_o is the dimension of the fused representation. The model selects the user with the largest probability as the prediction. Padding and previously activated users are masked before softmax and Top-K ranking throughout training and evaluation. The main training objective is the cross-entropy loss over next-user and terminal targets:

$$\mathcal{L}_{main} = -\sum_{c_m \in \mathcal{C}_{\text{train}}} \sum_{i=1}^{L_m} \log \mathbf{p}_{m,i}[y_{m,i}], \tag{22}$$

where $y_{m,i} = u_{i+1}^m$ for $i < L_m$ and $y_{m,L_m} = \mathrm{EOS}$. When graph contrastive regularization is enabled, the final objective is

$$\mathcal{L} = \mathcal{L}_{main} + \lambda_g \mathcal{L}_g, \tag{23}$$

where $\mathcal{L}_g$ is the graph contrastive loss and λ_g controls its contribution.

5. Experimental Setup

5.1. Datasets

We evaluate the proposed model on four real-world information diffusion datasets: Twitter [35], Douban [36], Android [21], and Christianity [21]. (1) Twitter. This dataset records URL diffusion traces on Twitter. Each URL is treated as an information item, and users who repost or mention the same URL form a time-ordered cascade. The social graph is built from user following relations. (2) Douban. This dataset contains book-related behaviors of users and their social connections on Douban. Each book is regarded as an information item, and a user is treated as activated after reading or sharing the book. (3) Android. This dataset is collected from Stack Exchange around the Android topic. Each cascade consists of chronologically ordered user activities associated with Android-related posts, and social links are derived from user interactions. (4) Christianity. This dataset is another topic-specific Stack Exchange dataset. It contains cascade interactions and the friendship network associated with the Christianity topic, providing a smaller but structurally explicit diffusion scenario. For each dataset, we randomly split the cascades into training, validation, and test sets with a ratio of 8:1:1. Only the training set is used during model optimization; the validation and test sets are never observed by the model during training. The detailed statistics are reported in Table 1.

5.2. Baselines

We compare CoDyNet with nine representative diffusion prediction methods, spanning sequential, graph, hypergraph, disentanglement, and scalable diffusion models. NDM uses CNNs and self-attention to capture user correlations in cascade sequences [4]. FOREST combines GRUs with GCNs to jointly model sequential diffusion patterns and social structures [9]. DyHGCN discretizes the diffusion process into dynamic heterogeneous subgraphs and applies graph convolution to learn temporal user preferences [10]. MS-HGAT constructs temporal hypergraphs to capture high-order interactions and fuses them with static social relations [11]. DisenIDP disentangles latent diffusion intents of users with graph and hypergraph neural networks, assisted by a self-supervised disentanglement task [37]. MINDS captures macro-level cascade interactions and social homophily through adversarial learning [38]. MIM learns multi-scale user intents via multi-channel hypergraphs, dual temporal modeling, and hierarchical knowledge distillation [39]. MHPS models multiple-granularity hypergraph sequences and introduces position-aware cross-attention mechanisms for next-user prediction [40]. SILN learns cascade-specific sphere-based representations from structural and temporal views, improving scalability on large social networks [41]. The baseline values in Tables 2 and 3 are compiled from the experimental results reported in the corresponding baseline publications cited in this subsection, rather than from new baseline runs conducted in this study.

Table 1. Statistics of the four real-world datasets.

Dataset	Users	Social Links	Cascades	Avg. Length
Twitter	12,627	309,631	3442	32.60
Douban	12,232	348,280	3475	21.76
Android	9958	48,573	679	33.30
Christianity	2897	35,624	587	25.10

Table 2. Overall performance in terms of Hit@K on four datasets.

Model	Twitter			Douban			Android			Christianity		
	H@10	**H@50**	**H@100**	**H@10**	**H@50**	**H@100**	**H@10**	**H@50**	**H@100**	**H@10**	**H@50**	**H@100**
NDM	0.1934	0.2941	0.3573	0.1013	0.2123	0.3125	0.0339	0.0953	0.1572	0.1651	0.3510	0.4553
FOREST	0.2552	0.3850	0.4607	0.1868	0.3084	0.3857	0.0700	0.1514	0.2237	0.2632	0.4909	0.6056
DyHGCN	0.2901	0.4688	0.5719	0.1987	0.3289	0.3942	0.0842	0.1915	0.2679	0.2594	0.4976	0.6047
MS-HGAT	0.2996	0.4654	0.5735	0.2065	0.3504	0.4136	0.1049	0.1987	0.2781	0.2939	0.4596	0.5444
DisenIDP	0.3273	0.4799	0.5540	0.2059	0.3545	0.4284	0.0946	0.1916	0.2684	0.2946	0.5000	0.6049
MINDS	0.3426	0.5283	0.5954	0.1956	0.3087	0.3641	0.1096	0.1989	0.2766	0.3214	0.4978	0.6250
MIM	0.3587	0.5228	0.6141	0.2095	0.3609	**0.4410**	0.0842	0.1713	0.2386	0.3036	0.5134	0.6362
MHPS	0.3643	0.5200	0.6158	0.2070	0.3480	0.4332	0.1151	0.2134	0.2979	0.3259	0.5268	0.6250
SILN	**0.3972**	**0.5567**	**0.6503**	0.2195	0.3623	0.4348	0.1110	0.2314	0.3043	0.3214	0.5313	0.6339
CoDyNet	0.3773	0.5461	0.6355	**0.2299**	**0.3658**	0.4403	**0.1173**	**0.2378**	**0.3123**	**0.3392**	**0.5402**	**0.6518**

Note: Bold numerical values indicate the best results, and underlined numerical values indicate the second-best results in each column.

Table 3. Overall performance in terms of M@K on four datasets.

Model	Twitter			Douban			Android			Christianity		
	M@10	**M@50**	**M@100**	**M@10**	**M@50**	**M@100**	**M@10**	**M@50**	**M@100**	**M@10**	**M@50**	**M@100**
NDM	0.1169	0.1243	0.1256	0.0581	0.0651	0.0663	0.0160	0.0202	0.0210	0.0676	0.0751	0.0765
FOREST	0.1733	0.1790	0.1801	0.1086	0.1146	0.1183	0.0381	0.0416	0.0426	0.1328	0.1433	0.1449
DyHGCN	0.1880	0.1951	0.1965	0.1122	0.1187	0.1198	0.0633	0.0675	0.0685	0.1303	0.1415	0.1432
MS-HGAT	0.1751	0.1832	0.1847	0.1048	0.1114	0.1148	0.0458	0.0503	0.0514	0.1751	0.1832	0.1847
DisenIDP	0.2159	0.2228	0.2239	0.1041	0.1110	0.1121	0.0582	0.0623	0.0634	0.1858	0.1946	0.1962
MINDS	0.2290	0.2328	0.2346	0.1142	0.1199	0.1213	0.0677	0.0716	0.0727	0.1955	0.2037	0.2054
MIM	0.2394	0.2469	0.2482	0.1081	0.1153	0.1164	0.0496	0.0533	0.0542	0.1857	0.1951	0.1968
MHPS	0.2447	0.2520	0.2533	0.1099	0.1167	0.1179	0.0651	0.0692	0.0704	0.2009	0.2103	0.2118
SILN	**0.2823**	**0.2879**	**0.2909**	0.1287	0.1341	0.1358	0.0717	0.0769	0.0779	0.1965	0.2052	0.2066
CoDyNet	0.2476	0.2553	0.2566	**0.1404**	**0.1466**	**0.1476**	**0.0739**	**0.0792**	**0.0802**	**0.2148**	**0.2235**	**0.2252**

Note: Bold numerical values indicate the best results, and underlined numerical values indicate the second-best results in each column.

5.3. Implementation Details

We implement CoDyNet in PyTorch and conduct all experiments on a server equipped with an NVIDIA RTX 3080 Ti GPU. Unless otherwise specified, the embedding and hidden dimensions are set to 64. The batch size is 64, the dropout rate is 0.3, and the maximum cascade length is 200. We train the model on Twitter, Android, and Christianity for 80 epochs, and on Douban for 30 epochs. Adam is used as the optimizer with the learning rate initialized at 0.001, $\beta_1 = 0.9$, and $\beta_2 = 0.98$. The implementation uses a warm-up learning-rate schedule with 1000 warm-up steps. We use the fourth-order Runge–Kutta method (RK-4) [42] as the ODE solver for the NCDE module. For graph contrastive regularization, the loss weight λ_g is 0.003 and the perturbation scale ϵ_g is 0.05. The contrastive temperature is set to 0.5, 0.6, 0.7, and 0.5 on Twitter, Douban, Android, and Christianity, respectively.

For reproducibility, the NCDE uses causal Hermite cubic interpolation with backward differences and fixed-step RK-4 without adjoint computation, and each context branch uses an eight-head causal Transformer. Graph contrastive learning uses the shared two-layer projection head and symmetric InfoNCE described in Section 4. The CoDyNet and ablation entries in the main tables are averaged over five independent runs under the same experimental settings.

Following the next-user prediction setting, we regard the task as a ranking problem over candidate users. We report Hit@K and MAP@K with $K \in \{10, 50, 100\}$. Hit@K measures whether the ground-truth next user appears in the top-K ranked candidates, while MAP@K further considers the rank position of the correct user. For compactness in the result tables, Hit@K and MAP@K are abbreviated as H@K and M@K, respectively. Because each prefix has exactly one relevant next user, AP@K equals the reciprocal rank of that user when it appears in the top K and zero otherwise; MAP@K is the mean of this truncated reciprocal rank over prefixes.

6. Results

6.1. Overall Performance

Tables 2 and 3 summarize the main results. Overall, CoDyNet achieves the most consistent performance across the four datasets. It obtains the best Hit@K scores on Android and Christianity and leads Hit@10 and Hit@50 on Douban. This pattern suggests that CoDyNet improves both narrow top-ranked retrieval and broader candidate screening. On Douban, CoDyNet remains highly competitive at the largest cutoff, indicating that its advantage is reflected primarily in ranking the correct user earlier rather than merely expanding the candidate set. Compared with the strongest non-CoDyNet baseline, CoDyNet improves the average Hit@K on Android and Christianity by 2.44% and 2.74%, respectively. On Douban, CoDyNet improves Hit@10 and Hit@50 by 4.74% and 0.97%, respectively, while remaining nearly tied with the best baseline on Hit@100 (0.4403 versus 0.4410).

The MAP@K results further support this observation. CoDyNet ranks first on Douban, Android, and Christianity across all three cutoffs, showing that the proposed graph-aware continuous dynamics modeling places the ground-truth user closer to the top of the ranked list. Relative to the best baseline, the average MAP@K gains are 9.03% on Douban, 3.00% on Android, and 6.51% on Christianity; across the nine MAP@K entries on these datasets, the average gain is 6.18%. Twitter is the main exception, where SILN performs best on both Hit@K and MAP@K. Nevertheless, CoDyNet remains competitive and outperforms most neural diffusion baselines. These results indicate that CoDyNet is particularly effective when high-order diffusion patterns and graph-aware regularization provide complementary signals for next-user prediction.

Twitter has the largest candidate-user space among the four datasets, which may favor SILN's scalable global structural and temporal representations over CoDyNet's more local high-order diffusion modeling. This interpretation is observational; the current experiments do not establish the cause of the gap.

6.2. Ablation Study

To examine the effect of the main components, we compare CoDyNet with six variants: w/o Graph CL removes the contrastive graph structure regularization, w/o NCDE removes the continuous diffusion dynamics module while keeping the dynamic hypergraph branch, ODE-GRU replaces the NCDE solver with ODE-GRU dynamics, Social only retains the original social graph, w/o Hypergraph bypasses the hypergraph encoder and feeds static features to the continuous path, and w/o Gate removes the adaptive fusion gate and uses equal branch weights. Table 4 reports Hit@50, Hit@100, MAP@50, and MAP@100 on Douban, Android, and Christianity.

The ablation results show that CoDyNet achieves the best results on all Douban and Christianity metrics, indicating that graph contrastive regularization, hypergraph modeling, and NCDE-based continuous dynamics provide complementary signals. The ODE-GRU and Social only variants lead to clear drops on Douban and Christianity, which supports the need for controlled continuous dynamics and multi-source graph construction. Android shows a less uniform pattern: CoDyNet remains strongest on Hit@50 and MAP@K, while ODE-GRU obtains the highest Hit@100 value. However, this advantage is limited to Android Hit@100 and does not consistently generalize across datasets or metrics.

Overall, the ablation results confirm that the performance of CoDyNet does not come from a single component. The static graph branch provides stable user priors, and graph contrastive regularization constrains clean and perturbed graph views. The dynamic hypergraph branch captures high-order diffusion structure, while the NCDE module refines cascade representations through continuous-time evolution. Their combination yields more reliable ranking performance across different diffusion scenarios. Because the temporal alternatives are not parameter matched, these ablations support the utility of the complete design but do not isolate the effect of continuous-time control from differences in model capacity.

6.3. User Representation Visualization

To make the role of graph embedding interpretable, we visualize how the same users move through the representation pipeline. For each dataset, we first build a training-cascade co-activation graph: two users are connected if they appear in the same training cascade, and the edge weight counts how often they are co-activated. We then apply Louvain clustering [43] and use the five largest communities as an external structural reference. These communities are derived only from the training cascades and are not used as supervision by the encoders.

We select 500 core users from the detected communities and keep this user set fixed across all stages. Each point in Figure 3 denotes one selected user. Its location is obtained by projecting the corresponding high-dimensional user embedding to a two-dimensional coordinate using t-SNE [44]. The projection is only used for visualization, so we compare local neighborhood structure and color organization rather than absolute coordinates across panels. The

Table 4. Ablation results on Douban, Android, and Christianity.

Variant	Douban				Android				Christianity			
	H@50	H@100	M@50	M@100	H@50	H@100	M@50	M@100	H@50	H@100	M@50	M@100
CoDyNet	**0.3658**	**0.4403**	**0.1466**	**0.1476**	**0.2378**	0.3123	**0.0792**	**0.0802**	**0.5402**	**0.6518**	**0.2235**	**0.2252**
w/o Graph CL	0.3618	0.4332	0.1441	0.1451	0.2299	0.3100	0.0772	0.0783	0.5290	0.6295	0.2165	0.2180
w/o NCDE	0.3657	0.4346	0.1410	0.1420	0.2253	0.3092	0.0703	0.0714	0.5290	0.6421	0.2180	0.2198
ODE-GRU	0.3534	0.4259	0.1412	0.1423	0.2300	**0.3178**	0.0790	0.0798	0.4933	0.5938	0.1978	0.1992
Social only	0.3504	0.4211	0.1348	0.1358	0.2152	0.2999	0.0752	0.0764	0.5223	0.6339	0.2166	0.2182
w/o Hypergraph	0.3615	0.4370	0.1397	0.1407	0.2300	0.2999	0.0747	0.0757	0.5246	0.6339	0.2104	0.2120
w/o Gate	0.3613	0.4377	0.1425	0.1436	0.2261	0.3077	0.0788	0.0799	0.5290	0.6473	0.2232	0.2249

Note: Bold numerical values indicate the best results in each column.

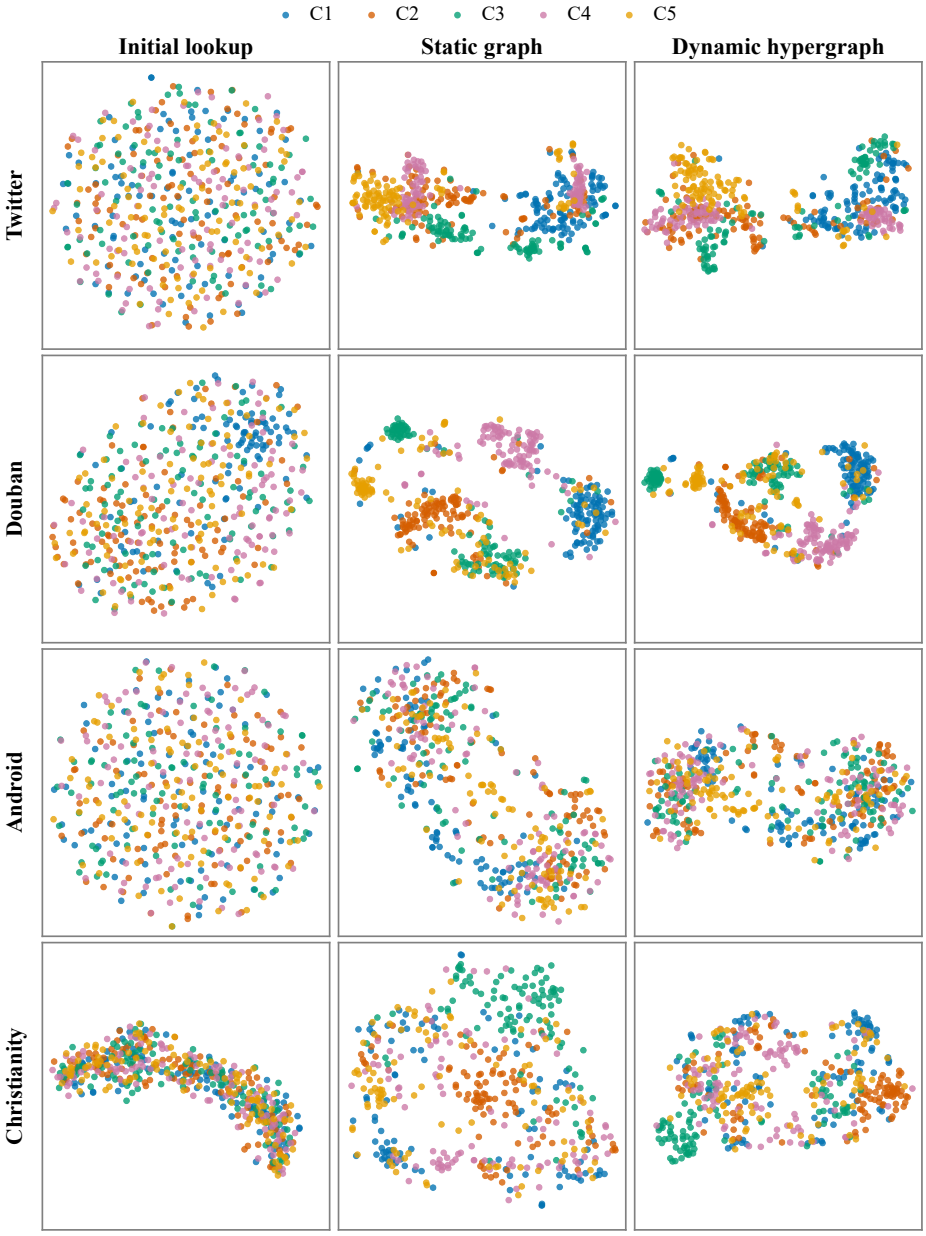


Figure 3. User representation evolution on the four datasets. Rows correspond to datasets, and columns show the learned lookup embeddings before graph propagation, static graph representations produced by the GCN, and dynamic hypergraph representations produced by the HGNN. Each point is a user embedding projected to two dimensions, and colors C1–C5 denote the five largest propagation communities identified independently from training cascades for each dataset.

three columns show the raw lookup embeddings, static graph embeddings after GCN propagation, and dynamic hypergraph embeddings after HGNN propagation, respectively.

Figure 3 provides an explanatory view of the graph embedding process. In the lookup space, different colors are heavily interleaved, indicating that ID-level embeddings alone provide limited explicit organization with respect to propagation structure. After GCN propagation, same-color users become more locally organized, showing that multi-source graph neighbors inject relational priors into the embedding space. The dynamic hypergraph embeddings further reshape the distribution according to high-order co-diffusion contexts, while still preserving role-level variation instead of collapsing all users into compact community centroids.

Overall, the column-wise changes show that the graph encoders transform raw lookup features into representations better aligned with the propagation communities observed in training cascades. Since the community colors are obtained independently from the visualization protocol and are not part of the training objective, this alignment provides interpretable evidence that graph embedding injects structural diffusion signals into CoDyNet.

6.4. Hyperparameter Analysis

For hyperparameter analysis, we follow a controlled protocol within each dataset. We vary one hyperparameter at a time while keeping the remaining settings fixed, so that the observed changes reflect the effect of the tested factor.

Figure 4 shows the effect of the contrastive temperature on Douban and Android. The performance changes smoothly across the tested range, but the best setting is not the same across datasets. On Douban, stronger performance appears in a different temperature region from Android, indicating that the contrastive objective should adapt to the graph structure and diffusion sparsity of each dataset. These results suggest that there is no universally optimal temperature coefficient. Instead, τ controls the balance between sharpening positive graph pairs and preserving separation among hard negatives, and its optimal value depends on the dataset-specific diffusion pattern.

Figure 5 further studies the depth of the neural vector field in the NCDE module. The results show that the vector field requires an appropriate depth rather than a monotonically deeper architecture. On Android, a moderate-depth vector field gives stronger ranking quality, whereas adding more layers leads to degradation and only partial recovery. On Christianity, the one- or two-layer settings remain competitive, while deeper configurations do not bring consistent gains. This pattern supports the design choice of using a two-layer vector field: it provides sufficient capacity to transform the controlled hidden state, while avoiding unnecessary depth that can overfit sparse cascade signals or make the continuous dynamics harder to optimize.

Figure 6 compares equal-duration temporal slicing on Android. Considering Hit@K and MAP@K across all evaluated cutoffs, $K = 8$ achieves the best overall performance and provides the most balanced trade-off between temporal resolution and hypergraph connectivity. Increasing the slice number yields only marginal gains on a few individual metrics while reducing others, whereas coarser settings lose more temporal detail.

7. Discussion

The results indicate that the static graph priors, dynamic hypergraphs, and controlled diffusion branch provide complementary signals. The ablations are consistent with a benefit from observation-driven continuous dynamics, but the unmatched alternatives do not separate this effect from differences in model capacity. The dataset-level gains are also heterogeneous: CoDyNet is strongest on most Douban, Android, and Christianity metrics, whereas SILN remains stronger on Twitter. Section 6.4 and the overall-performance analysis describe these variations without making causal claims.

Several limitations remain. The full architecture incurs the combined cost of multi-view graph construction, hypergraph encoding, NCDE solving, and Transformer context encoding. Moreover, CoDyNet predicts neither activation time nor ranking confidence. Future work can add a positive inter-event-time head for joint user–time prediction, report standardized efficiency measurements, and study confidence-aware ranking and scalable graph construction.

8. Conclusions

This paper presented CoDyNet, a graph-aware controlled continuous dynamics network for microscopic information diffusion prediction. CoDyNet addresses the limitations of purely discrete diffusion modeling by combining multi-source graph representation learning, temporal hypergraph encoding, and NCDE-based continuous cascade dynamics. The model further introduces cascade-aware graph contrastive regularization to align clean and perturbed static user representations while reducing false negatives caused by users co-occurring in the same cascade. By adaptively fusing the continuous diffusion branch with the static branch, CoDyNet captures both short-term propagation dynamics and long-term relational preferences for next-user prediction.

Experiments on Twitter, Douban, Android, and Christianity show that CoDyNet achieves competitive or superior performance against state-of-the-art models, particularly for MAP@K on three datasets. Ablation studies

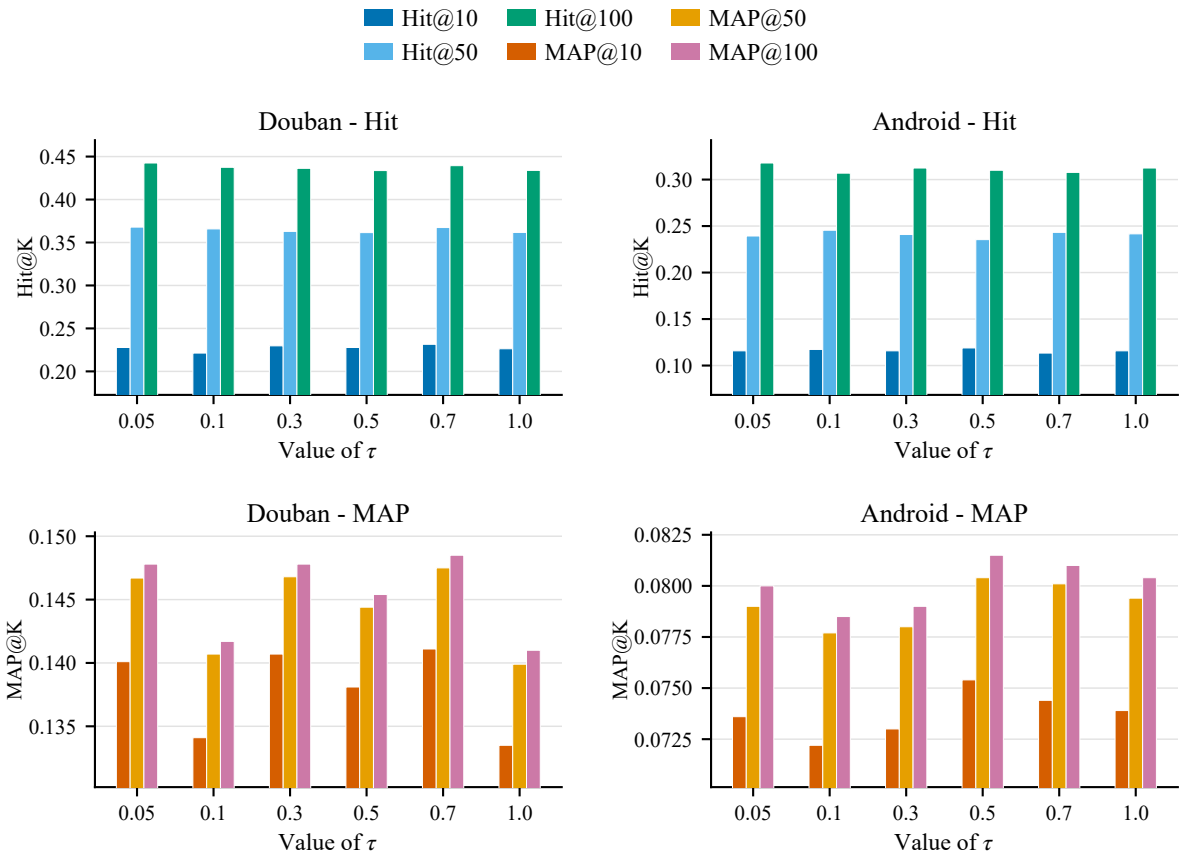


Figure 4. Effect of the contrastive temperature τ on Douban and Android.

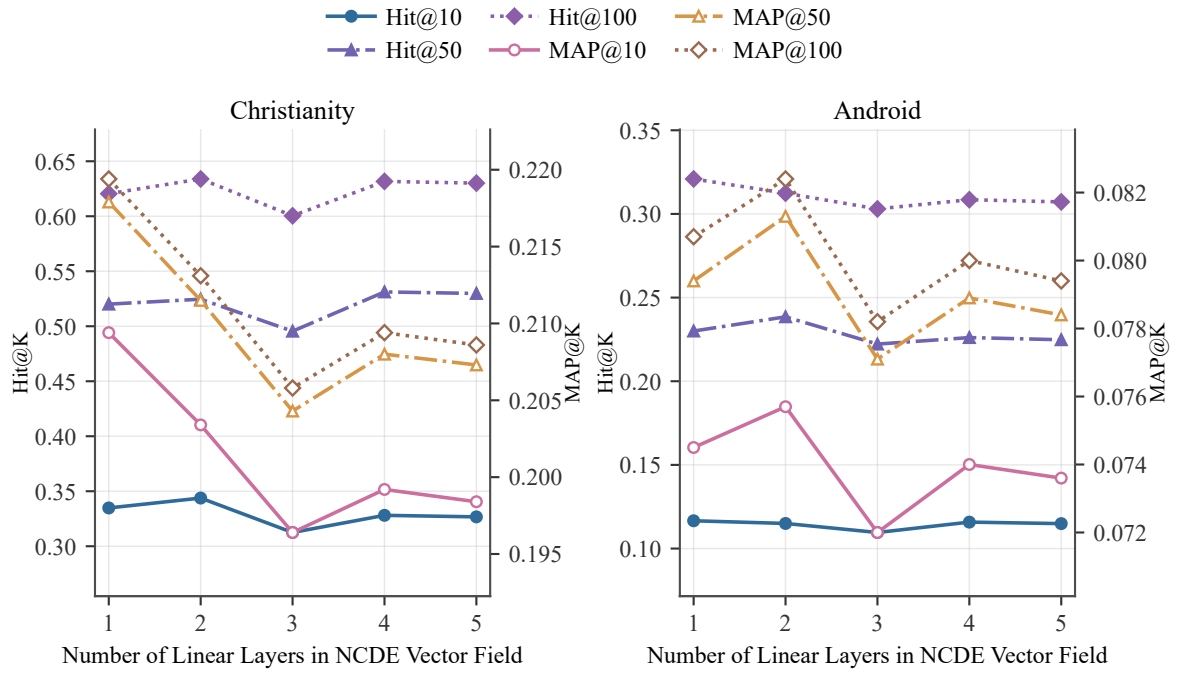


Figure 5. Effect of NCDE vector-field depth on Christianity and Android.

provide evidence for the complementarity of the proposed modules, and hyperparameter analysis characterizes sensitivity to key design choices. These findings highlight the value of controlled continuous dynamics for irregular information diffusion modeling.

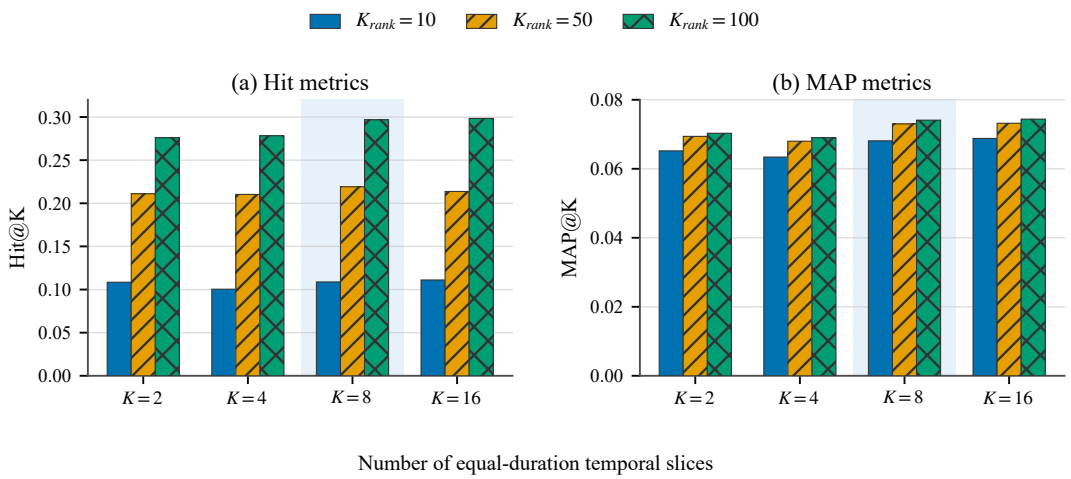


Figure 6. Sensitivity to temporal hypergraph slicing on Android. Each setting uses equal-duration slicing and the same 40-epoch budget; reported values are averaged over five independent runs. The shaded column marks the default $K = 8$ setting.

Author Contributions

J.Z.: conceptualization, methodology, software, data curation, formal analysis, investigation, validation, visualization, writing—original draft preparation, and writing—review and editing; F.X.: supervision, methodology, funding acquisition, and writing—review and editing; Z.Z.: methodology and writing—review and editing; P.H.: formal analysis and writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding

This research was funded by the National Natural Science Foundation of China under Grant 62472024.

Institutional Review Board Statement

Not applicable. This study involved secondary analysis of publicly available datasets and did not include human participant recruitment, interventions, or animal experiments.

Informed Consent Statement

Not applicable. The study used existing publicly available datasets and did not collect data directly from participants.

Data Availability Statement

The Twitter and Douban datasets are described in Refs. [35,36], and the Android and Christianity datasets are described in Ref. [21]. Public benchmark copies are available in the MS-HGAT data repository (https://github.com/slingling/MS-HGAT/tree/main/data) [11]. The Android and Christianity datasets are also available in the Inf-VAE repository (https://github.com/aravindsankar28/Inf-VAE/tree/master/data).

Conflicts of Interest

The authors declare no conflict of interest.

Use of AI and AI-Assisted Technologies

During the preparation of this work, the authors used OpenAI ChatGPT to assist with manuscript format migration and language editing. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

References

1. Kempe, D.; Kleinberg, J.; Tardos, É. Maximizing the Spread of Influence through a Social Network. In Proceedings of the KDD'03: The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 24–27 August 2003; pp. 137–146. https://doi.org/10.1145/956750.956769.

2. Cheng, J.; Adamic, L.; Dow, P.A.; et al. Can Cascades Be Predicted? In Proceedings of the WWW'14: 23rd International World Wide Web Conference, Seoul, Korea, 7–11 April 2014; pp. 925–936. https://doi.org/10.1145/2566486.2567997.
3. Vosoughi, S.; Roy, D.; Aral, S. The Spread of True and False News Online. *Science* **2018**, *359*, 1146–1151. https://doi.org/10.1126/science.aap9559.
4. Yang, C.; Sun, M.; Liu, H.; et al. Neural Diffusion Model for Microscopic Cascade Study. *IEEE Trans. Knowl. Data Eng.* **2021**, *33*, 1128–1139. https://doi.org/10.1109/tkde.2019.2939796.
5. Mishra, S.; Rizoiu, M.A.; Xie, L. Feature Driven and Point Process Approaches for Popularity Prediction. In Proceedings of the CIKM'16: ACM Conference on Information and Knowledge Management, Indianapolis, IN, USA, 24–28 October 2016; pp. 1069–1078. https://doi.org/10.1145/2983323.2983812.
6. Li, C.; Ma, J.; Guo, X.; et al. DeepCas: An End-to-End Predictor of Information Cascades. In Proceedings of the WWW'17: 26th International World Wide Web Conference, Perth, WA, Australia, 3–7 April 2017; pp. 577–586. https://doi.org/10.1145/3038912.3052643.
7. Cao, Q.; Shen, H.; Cen, K.; et al. DeepHawkes: Bridging the Gap between Prediction and Understanding of Information Cascades. In Proceedings of the CIKM'17: ACM Conference on Information and Knowledge Management, Singapore, 6–10 November 2017; pp. 1149–1158. https://doi.org/10.1145/3132847.3132973.
8. Wang, Y.; Shen, H.; Liu, S.; et al. Cascade Dynamics Modeling with Attention-Based Recurrent Neural Network. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, Melbourne, VIC, Australia, 19–25 August 2017; pp. 2985–2991. https://doi.org/10.24963/ijcai.2017/416.
9. Yang, C.; Tang, J.; Sun, M.; et al. Multi-Scale Information Diffusion Prediction with Reinforced Recurrent Networks. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, Macao, China, 10–16 August 2019; pp. 4033–4039. https://doi.org/10.24963/ijcai.2019/560.
10. Yuan, C.; Li, J.; Zhou, W.; et al. DyHGCN: A Dynamic Heterogeneous Graph Convolutional Network to Learn Users' Dynamic Preferences for Information Diffusion Prediction. In *Machine Learning and Knowledge Discovery in Databases, Proceedings of the European Conference, ECML PKDD 2020, Ghent, Belgium, 14–18 September 2020*; Springer International Publishing: Cham, Switzerland, 2021; pp. 347–363. https://doi.org/10.1007/978-3-030-67664-3_21.
11. Sun, L.; Rao, Y.; Zhang, X.; et al. MS-HGAT: Memory-Enhanced Sequential Hypergraph Attention Network for Information Diffusion Prediction. In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 22 February–1 March 2022; pp. 4156–4164.
12. Chen, R.T.Q.; Rubanova, Y.; Bettencourt, J.; et al. Neural Ordinary Differential Equations. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 2–8 December 2018; Volume 31, pp. 6571–6583.
13. Kidger, P.; Morrill, J.; Foster, J.; et al. Neural Controlled Differential Equations for Irregular Time Series. In Proceedings of the 34th International Conference on Neural Information Processing Systems, Online, 6–12 December 2020; Volume 33, pp. 6696–6707.
14. Wang, D.; Zhou, W.; Hu, S. Information Diffusion Prediction with Graph Neural Ordinary Differential Equation Network. In Proceedings of the MM'24: The 32nd ACM International Conference on Multimedia, Melbourne, VIC, Australia, 28 October–1 November 2024; pp. 9699–9708. https://doi.org/10.1145/3664647.3681363.
15. Morrill, J.; Kidger, P.; Yang, L.; et al. Neural Controlled Differential Equations for Online Prediction Tasks. *arXiv* **2021**, arXiv:2106.11028v1. http://arxiv.org/abs/2106.11028v1.
16. Suh, B.; Hong, L.; Pirolli, P.; et al. Want to Be Retweeted? Large Scale Analytics on Factors Impacting Retweet in Twitter Network. In Proceedings of the 2010 IEEE Second International Conference on Social Computing (SocialCom), Minneapolis, MN, USA, 20–22 August 2010; pp. 177–184. https://doi.org/10.1109/socialcom.2010.33.
17. Yu, L.; Cui, P.; Wang, F.; et al. From Micro to Macro: Uncovering and Predicting Information Cascading Process with Behavioral Dynamics. In Proceedings of the 2015 IEEE International Conference on Data Mining (ICDM), Atlantic City, NJ, USA, 14–17 November 2015; pp. 559–568. https://doi.org/10.1109/icdm.2015.79.
18. Rizoiu, M.A.; Xie, L.; Sanner, S.; et al. Expecting to Be HIP: Hawkes Intensity Processes for Social Media Popularity. In Proceedings of the WWW'17: 26th International World Wide Web Conference, Perth, WA, Australia, 3–7 April 2017; pp. 735–744. https://doi.org/10.1145/3038912.3052650.
19. Farajtabar, M.; Wang, Y.; Gomez-Rodriguez, M.; et al. COEVOLVE: A Joint Point Process Model for Information Diffusion and Network Evolution. *J. Mach. Learn. Res.* **2017**, *18*, 1305–1353.
20. Li, H.; Xia, C.; Wang, T.; et al. GRASS: Learning Spatial–Temporal Properties from Chainlike Cascade Data for Microscopic Diffusion Prediction. *IEEE Trans. Neural Netw. Learn. Syst.* **2024**, *35*, 16313–16327. https://doi.org/10.1109/tnnls.2023.3293689.
21. Sankar, A.; Zhang, X.; Krishnan, A.; et al. Inf-VAE: A Variational Autoencoder Framework to Integrate Homophily and Influence in Diffusion Prediction. In Proceedings of the WSDM'20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, 3–7 February 2020; pp. 510–518. https://doi.org/10.1145/3336191.3371811.
22. Qiao, H.; Feng, S.; Li, X.; et al. RotDiff: A Hyperbolic Rotation Representation Model for Information Diffusion Prediction. In Proceedings of the CIKM'23: The 32nd ACM International Conference on Information and Knowledge Management, Birmingham, UK, 21–25 October 2023; pp. 2065–2074. https://doi.org/10.1145/3583780.3615041.

23. He, W.; Xiao, Y.; Huang, M.; et al. A Pattern-Driven Information Diffusion Prediction Model Based on Multi-source Resonance and Cognitive Adaptation. In Proceedings of the SIGIR'25: The 48th International ACM SIGIR Conference on Research and Development in Information Retrieval, Padua, Italy, 13–18 July 2025; pp. 592–601. https://doi.org/10.1145/3726302.3729883.
24. Perozzi, B.; Al-Rfou, R.; Skiena, S. DeepWalk: Online Learning of Social Representations. In Proceedings of the KDD'14: The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 24–27 August 2014; pp. 701–710. https://doi.org/10.1145/2623330.2623732.
25. Tang, J.; Qu, M.; Wang, M.; et al. LINE: Large-Scale Information Network Embedding. In Proceedings of the WWW'15: 24th International World Wide Web Conference, Florence, Italy, 18–22 May 2015; pp. 1067–1077. https://doi.org/10.1145/2736277.2741093.
26. Grover, A.; Leskovec, J. node2vec: Scalable Feature Learning for Networks. In Proceedings of the KDD'16: The 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 855–864. https://doi.org/10.1145/2939672.2939754.
27. Veličković, P.; Fedus, W.; Hamilton, W.L.; et al. Deep Graph Infomax. *arXiv* **2018**, arXiv:1809.10341.
28. You, Y.; Chen, T.; Sui, Y.; et al. Graph Contrastive Learning with Augmentations. In Proceedings of the 34th International Conference on Neural Information Processing Systems, Online, 6–12 December 2020; Volume 33, pp. 5812–5823.
29. Zhu, Y.; Xu, Y.; Yu, F.; et al. Graph Contrastive Learning with Adaptive Augmentation. In Proceedings of the WWW'21: The Web Conference 2021, Online, 19–23 April 2021; pp. 2069–2080. https://doi.org/10.1145/3442381.3449802.
30. Wu, J.; Wang, X.; Feng, F.; et al. Self-Supervised Graph Learning for Recommendation. In Proceedings of the SIGIR'21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Online, 11–15 July 2021; pp. 726–735. https://doi.org/10.1145/3404835.3462862.
31. Yu, J.; Yin, H.; Xia, X.; et al. Are Graph Augmentations Necessary? Simple Graph Contrastive Learning for Recommendation. In Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, 11–15 July 2022; pp. 1294–1303. https://doi.org/10.1145/3477495.3531937.
32. Yu, J.; Xia, X.; Chen, T.; et al. XSimGCL: Towards Extremely Simple Graph Contrastive Learning for Recommendation. *IEEE Trans. Knowl. Data Eng.* **2024**, *36*, 913–926. https://doi.org/10.1109/tkde.2023.3288135.
33. Jhin, S.Y.; Shin, H.; Kim, S.; et al. Attentive Neural Controlled Differential Equations for Time-Series Classification and Forecasting. *Knowl. Inf. Syst.* **2024**, *66*, 1885–1915. https://doi.org/10.1007/s10115-023-01977-5.
34. van den Oord, A.; Li, Y.; Vinyals, O. Representation Learning with Contrastive Predictive Coding. *arXiv* **2018**, arXiv:1807.03748v2.
35. Hodas, N.O.; Lerman, K. The Simple Rules of Social Contagion. *Sci. Rep.* **2014**, *4*, 4343. https://doi.org/10.1038/srep04343.
36. Zhong, E.; Fan, W.; Wang, J.; et al. ComSoc: Adaptive Transfer of User Behaviors over Composite Social Network. In Proceedings of the KDD'12: The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Beijing, China, 12–16 August 2012; pp. 696–704. https://doi.org/10.1145/2339530.2339641.
37. Cheng, Z.; Ye, W.; Liu, L.; et al. Enhancing Information Diffusion Prediction with Self-Supervised Disentangled User and Cascade Representations. In Proceedings of the CIKM'23: The 32nd ACM International Conference on Information and Knowledge Management, Birmingham, UK, 21–25 October 2023; pp. 3808–3812. https://doi.org/10.1145/3583780.3615230.
38. Jiao, P.; Chen, H.; Bao, Q.; et al. Enhancing Multi-Scale Diffusion Prediction via Sequential Hypergraphs and Adversarial Learning. In Proceedings of the 38th AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 20–27 February 2024; Volume 38, pp. 8571–8581. https://doi.org/10.1609/aaai.v38i8.28701.
39. Cheng, Z.; Liu, Y.; Zhong, T.; et al. Disentangling Inter- and Intra-Cascades Dynamics for Information Diffusion Prediction. *IEEE Trans. Knowl. Data Eng.* **2025**, *37*, 4548–4563. https://doi.org/10.1109/tkde.2025.3568289.
40. Jing, W.; Wang, Y.; Du, H.; et al. Enhancing Information Diffusion Prediction via Multiple Granularity Hypergraphs and Position-Aware Sequence Model. In Proceedings of the CIKM'25: The 34th ACM International Conference on Information and Knowledge Management, Seoul, Korea, 10–14 November 2025; pp. 1239–1248. https://doi.org/10.1145/3746252.3761259.
41. Feng, Z.; Yang, Y.; Huang, X.; et al. Efficient Sphere-Effect Based Information Diffusion Prediction on Large-Scale Social Networks. In Proceedings of the KDD'25: The 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Toronto, ON, Canada, 3–7 August 2025; pp. 615–625. https://doi.org/10.1145/3711896.3736925.
42. Runge, C. Ueber Die Numerische Auflösung Von Differentialgleichungen. *Math. Ann.* **1895**, *46*, 167–178. https://doi.org/10.1007/bf01446807.
43. Blondel, V.D.; Guillaume, J.L.; Lambiotte, R.; et al. Fast Unfolding of Communities in Large Networks. *J. Stat. Mech. Theory Exp.* **2008**, *2008*, P10008. https://doi.org/10.1088/1742-5468/2008/10/p10008.
44. van der Maaten, L.; Hinton, G.E. Visualizing Data Using T-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.