HyperIDP: Customizing Temporal Hypergraph Neural Networks for Multi-Scale Information Diffusion Prediction

Haowei Xu[1], **Chao Gao**[1], **Xianghua Li**[1,*], **Zhen Wang**[1,2],

[1]School of Artificial Intelligence, Optics and Electronics (iOPEN),
Northwestern Polytechnical University, P.R. China

[2]School of Cybersecurity, Northwestern Polytechnical University, P.R. China

Correspondence: li_xianghua@nwpu.edu.cn

Abstract

Information diffusion prediction is crucial for understanding how information spreads within social networks, addressing both macroscopic and microscopic prediction tasks. Macroscopic prediction assesses the overall impact of diffusion, while microscopic prediction focuses on identifying the next user likely to be influenced. However, few studies have focused on both scales of diffusion. This paper presents **HyperIDP**, a novel **Hyper**graph-based model designed to manage both macroscopic and microscopic **I**nformation **D**iffusion **P**rediction tasks. The model captures interactions and dynamics of cascades at the macro level with hypergraph neural networks (HGNNs) while integrating social homophily at the micro level. Considering the diverse data distributions across social media platforms, which necessitate extensive tuning of HGNN architectures, a search space is constructed to accommodate diffusion hypergraphs, with optimal architectures derived through differentiable search strategies. Additionally, cooperative-adversarial loss, inspired by multi-task learning, is introduced to ensure that the model can leverage the advantages of the shared representation when handling both tasks, while also avoiding potential conflicts. Experimental results show that the proposed model significantly outperforms baselines.

1 Introduction

Social platforms are integral to modern life, enhancing instant communication and facilitating rapid information dissemination. User activity patterns within these networks are crucial to the spread of information, often resulting in information cascades. A comprehensive understanding of the mechanisms underlying information diffusion offers significant economic and social benefits, with applications in areas such as fake news detection (Kim et al., 2021), viral marketing (AlSuwaidan and Ykhlef, 2016), and recommendation systems (Wu et al., 2022).

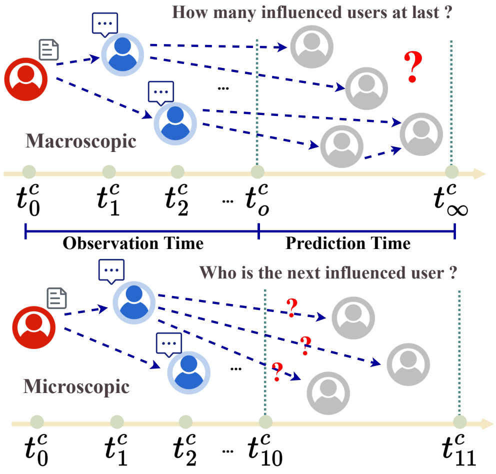


Figure 1: The illustrations demonstrate two key aspects: the prediction of macroscopic cascade size on the left and the prediction of the next user likely to be influenced at the microscopic level on the right.

Current research on modeling information cascades primarily addresses two key aspects: macroscopic prediction, which estimates the incremental or total size of a cascade (Li et al., 2017; Chen et al., 2019b; Sun et al., 2023), and microscopic prediction, which identifies the next user likely to be influenced within the cascade (Wang et al., 2017, 2018; Yu et al., 2022). On the one hand, macro-prediction concentrates on overarching patterns and trends, employing network topology and dissemination models to forecast information propagation. On the other hand, micro-prediction delves into the particulars of individual users' behaviors and attributes, utilizing analyses of user and content characteristics to anticipate the impact of information diffusion. Macro-prediction and micro-prediction collectively provide a comprehensive understanding of information dissemination and can mutually reinforce and enhance each other (Guo et al., 2024). Since both tasks require learning propagation features from observed cascades, they inherently share commonalities. (Jiao et al., 2024) In the context

Proceedings of the 31st International Conference on Computational Linguistics, pages 964–977
January 19–24, 2025. ©2025 Association for Computational Linguistics

of multi-task learning (Zhang and Yang, 2018), the need to improve prediction accuracy through the extraction of common features across tasks is of critical importance. Furthermore, balancing shared and task-specific representations is essential. Although encouraging shared representations can enhance overall performance, it may also create conflicts with task-specific representations, potentially limiting generalization (Zhang and Yang, 2021).

However, simultaneously conducting diffusion prediction at two different scales presents several major challenges, which can be categorized into two main types. The first is ***complexity of interactions***. Information dissemination involves intricate interactions both within individual cascades and across different cascades, making it difficult to effectively capture and model these dynamics (Jin et al., 2022; Sun et al., 2022). The second is ***cross-platform generalization***. The substantial variation in data distribution and user behavior across social media platforms complicates the transferability of models trained on one platform to another. Existing methods struggle to adapt to the specific characteristics of each platform, resulting in poor cross-platform generalization. Moreover, manually adjusting model architectures for different platforms is time-intensive and often fails to achieve optimal, data-specific results (Ren et al., 2021a).

To overcome the aforementioned challenges, **HyperIDP** is introduced as a streamlined and efficient **Hyper**graph-based framework for multi-scale **I**nformation **D**iffusion **P**rediction. At the macro level, sequential hypergraphs are constructed to effectively capture interactions and dynamics among cascades, aligning with the hypergraph's capacity to model complex user and cascade interactions. Dividing the time period into sequential windows allows for an accurate depiction of the dynamic evolution of cascades. At the micro level, the framework emphasizes the role of social homophily within social networks. Additionally, an uncertainty-weighted center loss, inspired by multi-task learning, is employed to preserve the integrity of shared features. Moreover, a differentiable hypergraph neural architecture search method is proposed for automatic hypergraph learning. Key contributions of this work are summarized as follows:

1) **Temporal hypergraph-based cooperative-adversarial cascade diffusion modeling**. To address the complexity of interactions, HyperIDP integrates both macro and micro prediction tasks, leveraging their mutual reinforcement to enhance overall performance. The method models information diffusion within temporal hypergraphs, capturing the interactions and dynamics between cascades. Inspired by the multi-task learning paradigm, a cooperative-adversarial loss function is employed to preserve the integrity of shared features while simultaneously reducing conflicts.

2) **Automated hypergraph neural network design**. To enhance the cross-platform generalizability, a differentiable neural architecture search method is introduced to enable automatic diffusion hypergraph learning. By designing a comprehensive search space, HyperIDP outperforms the performance of existing human-designed baselines.

3) Experimental results on real-world social media datasets demonstrate that the proposed method significantly outperforms existing approaches in terms of accuracy and robustness, validating the framework's effectiveness.

2 Methodology

This section provides an in-depth explanation of the components and design principles of the proposed framework, as depicted in Figure 2. HyperIDP comprises four primary components:

1) **Global Interaction Learning Module:** This component extracts user preferences at each time interval and models the dynamic changes in cascades using Hypergraph Neural Networks (HGNNs), with a fusion layer facilitating integration at the cascade level.

2) **Social Relationship Learning Module:** It captures social relationships at the individual user level through the application of Graph Neural Networks (GNNs).

3) **Diffusion Prediction Module:** This module employs uncertainty-weighted center loss to learn both shared and task-specific representations for multi-scale diffusion prediction.

4) **Neural Architecure Search Module:** This component constructs a comprehensive search space based on diffusion hypergraphs and social graphs, utilizing a differentiable architecture search algorithm to identify the optimal model configuration.

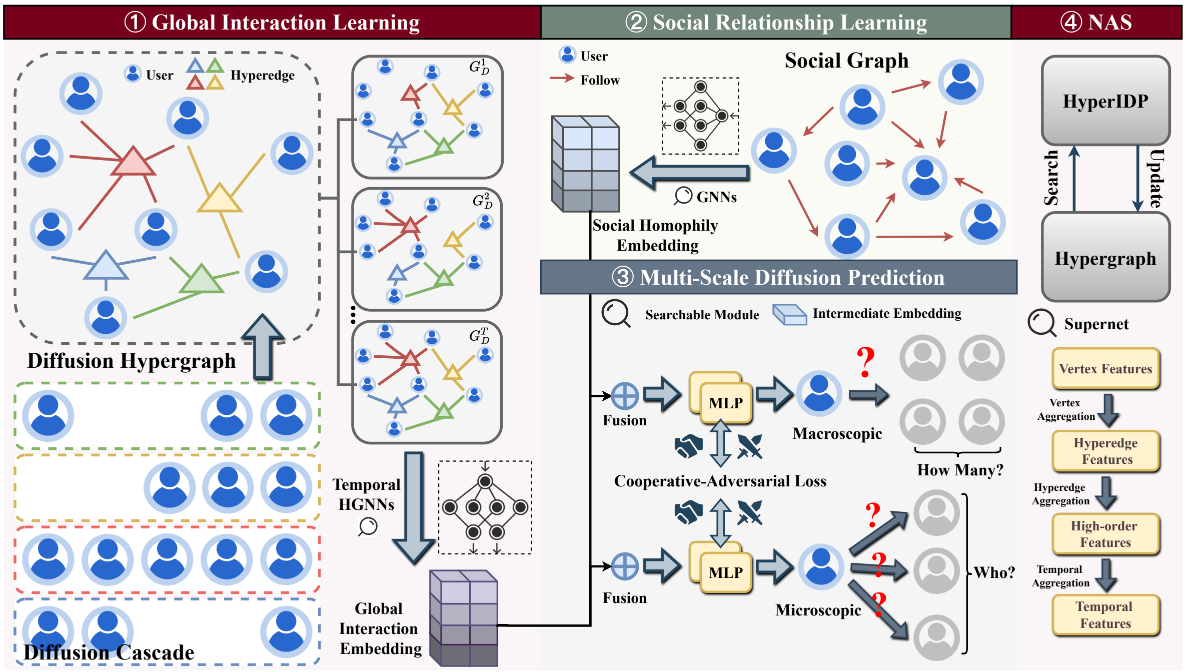


Figure 2: The architectural overview of our model.

2.1 Problem Formulation

To commence, we present the social graph and diffusion hypergraphs that constitute the foundation for diffusion prediction within our model. The social graph is denoted as $G_S = (U, E)$, where U is the user set and E is the edge set. Each edge $(u_i, u_j) \in E$ represents a social relationship between user u_i and u_j. The observed diffusion cascades $D = \{d_1, d_2, \ldots, d_M\}, |D| = N$ are split into T subsets according to timestamps for constructing sequential diffusion hypergraphs $G_D = \{G_D^t \mid t = 1, 2, \ldots, T\}, G_D^t = (U^t, \mathcal{E}^t)$, where U^t is the user set and $\mathcal{E}^t$ is the hyperedge set. In the diffusion hypergraph, users participate in the same cascade and are connected by a hyperedge, in other words, a hyperedge represents a cascade. Note that the set of nodes connected by hyperedge is different in each hypergraph. It means that if u_i participates in d_m during the t-th time interval, then u_i being connected to hyperedge e_m only occurs in diffusion hypergraph G_D^t. This work aims to address both the macroscopic and microscopic problems based on the above introductions.

Macroscopic Diffusion Prediction: Given a social graph G_S, diffusion hypergraphs G_D and an observed diffusion sequence $d_m = \{(u_i^m, t_i^m) \mid u_i^m \in U\}$, estimate the final size $|d_m|$ of cascade d_m.

Microscopic Diffusion Prediction: Given a social graph G_S, diffusion hypergraphs G_D and an observed diffusion sequence $d_m = \{(u_i^m, t_i^m) \mid u_i^m \in U\}$, predict which user will participate in d_m in the next step.

2.2 User Global Interaction Learning

To account for global interactions among cascades and the dynamic changes within them simultaneously, the HGNN is employed based on the constructed sequential diffusion hypergraphs. HGNN captures global user interactions within each distinct time interval at the cascade level, while a fusion layer between consecutive time intervals models the evolving dynamics of cascades.

Hypergraph Neural Network

User interactions are modeled at each time interval using HGNNs, as illustrated in Figure 3. In a standard graph, graph convolution aggregates neighboring vertices to generate a new representation of the central vertex, with information flowing through the edges. Similarly, in a hypergraph, hyperedges serve as channels for information transmission. The message aggregation process within a hypergraph is executed in two stages: (1) Vertex Aggregation and (2) Hyperedge Aggregation.

Vertex Aggregation

Given a diffusion hypergraph snapshot G_D^t, the features of the hyperedge need to be obtained by aggregating the features of the vertices in the hyperedge. Specifically, hyperedge feature can be calculated by:

$$X_{e,t} = \mathrm{conv}\left(\mathrm{Merg}\left(X_{v,t}^{(i)}, \forall i \in \{0, \ldots, k\}\right)\right), \tag{1}$$

where $X_{v,t}^{(i)}$ is the features of the i-th vertex in a hyperedge $X_{e,t}$. The $\mathrm{Merg}(\cdot)$ mechanism merges the message of all the vertices and the $\mathrm{conv}(\cdot)$ operator indicates that 1-dimension convolution is used to compact the derived result.

Hyperedge Aggregation

We regard each vertex as a center point c, and then aggregate the hyperedge features associated with it to obtain the high-order feature of c, denoted as $X_{h,t}$ at t-th time interval. The attention mechanism is employed to generate the weights for each hyperedge in different ways. The high-order feature is calculated as:

$$X_{h,t} = \sum_{i=0}^{m} w_{e,t}^{(i)} X_{e,t}^{(i)}, \tag{2}$$

where m represents the number of hyperedges associated with the centroid vertex, and w represents the calculated weights of hyperedges.

Sequential HGNNs with Fusion Layer

The above two-stage convolution operation only learns user interaction at a specific time interval, which can not adequately characterize the evolution of cascades in propagation. Therefore, we design a fusion strategy to connect the interactions at different time intervals learned by HGNN in chronological order, which is defined as:

$$\mathbf{X}_D = \mathrm{Fuse}\left(X_{h,1}, \cdots, X_{h,T}\right), \tag{3}$$

where$\mathbf{X}_D$ is the final global interactive representation obtained through sequential HGNNs. $[X_{h,1}, \cdots, X_{h,T}]$ denote the representation of every diffusion hypergraphs. And $\mathrm{Fuse}(\cdot)$ operator represents various fusion strategies.

2.3 User Social Relationship Learning

User tends to have more social interactions with users who are similar to them and this refers to the principle called social relationship. Close friends, who are usually friends alike in certain qualities or interests, have more influence on each other than dissimilar ones. Users' social relationship can be reflected through social network structure. We introduce the social graph to model user social relationships and apply a multi-layer GNN to embed social relationship. Given social graph $G_S = (U, E)$, the user social relationship embedding matrix $\mathbf{X}_S^l$ at l-th layer is updated by:

$$\mathbf{X}_S^{l+1} = \mathrm{GNN}\left(\tilde{\mathbf{A}}_S, \mathbf{X}_S^l\right), \tag{4}$$

where $\tilde{\mathbf{A}_S}$ is the adjacent matrix of self-looped G_S. The initial relationship embedding matrix $\mathbf{X}_S^0 \in \mathbb{R}^{N \times d}$ is randomly initialized from a normal distribution, and d is the dimension of embedding. $\mathrm{GNN}(\cdot)$ operator indicates different types of GNNs. We can obtain the final social relationship representation $\mathbf{X}_S$ after several layers of them.

2.4 Multi-Scale Diffusion Prediction

We concatenate the global interaction and social relationship representation $\mathbf{X}_D$, $\mathbf{X}_S$ and then feed them into distinct output layers dedicated to the multi-scale diffusion prediction process.

Macroscopic Diffusion Prediction

For macroscopic diffusion prediction, we aim to predict the final cascade size in the future. We calculate the final size of diffusion cascade d_m by:

$$S_m = \mathrm{Linear}\left(\mathrm{concat}\left(\mathbf{X}_D, \mathbf{X}_S\right)\right), \tag{5}$$

where concat $(\cdot, \cdot)$ is the concatenation operation and $\mathrm{Linear}(\cdot)$ represents a multilayer perceptron (MLP). We train the macroscopic task by minimizing the following loss function:

$$\mathcal{L}_{\text{macro}} = \frac{1}{M} \sum_{m=1}^{M} \left(S_m - \hat{S}_m\right)^2, \tag{6}$$

where M is the number of diffusion cascades and $\hat{S}_m$ is the ground truth.

Microscopic Diffusion Prediction

For microscopic diffusion prediction, the next influenced probability $p_i \in \mathbb{R}^{|d_m|}$ for user u_i is predicted by:

$$p_i = \mathrm{softmax}\left(\mathrm{Linear}\left(\mathrm{concat}\left(\mathbf{X}_D, \mathbf{X}_S\right)\right)\right), \tag{7}$$

We adopt the cross entropy (CE) loss for microscopic training:

$$\mathcal{L}_{\text{micro}} = -\sum_{j=2}^{|d_m|} \sum_{i=1}^{|U|} \hat{p}_{ji} \log\left(p_{ji}\right), \tag{8}$$

where $|U|$ is the number of users and $\tilde{p}$ is true probability. If user u_i participate in cascade d_m at the step j, then $\hat{p}_{ji} = 1$, otherwise $\hat{p}_{ji} = 0$.

Training with Cooperative-Adversarial Loss

In multi-task learning (MTL), balancing shared and conflicting task representations is crucial (Zhang and Yang, 2018). Although promoting shared representations can enhance overall performance, it may also cause conflicts between task-specific representations, which can hinder generalization (Yang et al., 2019b).

To address this, we introduce a novel loss function combining cooperative and adversarial components. This loss function fosters consistency between task representations while reducing conflicts, thereby improving MTL performance. Specifically, the cooperative loss encourages consistent or complementary representations within shared layers, quantified as the Euclidean distance between the outputs of the shared representation space:

$$\mathcal{L}_{\text{coop}} = \gamma \left\| \mathbf{H}_{\text{macro}} - \mathbf{H}_{\text{micro}} \right\|^2 , \tag{9}$$

where $\mathbf{H}_{\text{macro}}$ and $\mathbf{H}_{\text{micro}}$ represent the shared representation space for the two scales of diffusion prediction tasks, and γ is the weight of the cooperative loss. To prevent one task from overly dominating the shared layer and leading to the collapse of the representation space, adversarial loss (L_{adv}) is introduced to ensure that the representations of different tasks maintain diversity and independence. The adversarial loss is realized by limiting the consistency of gradient directions between macro-level and micro-level prediction, defined as the inner product of the gradients of the two tasks:

$$\mathcal{L}_{\text{adv}} = \delta \left\| \nabla \mathcal{L}_{\text{macro}} \cdot \nabla \mathcal{L}_{\text{micro}} \right\| , \tag{10}$$

where $\nabla_\theta L_{\text{macro}}$ and $\nabla_\theta L_{\text{micro}}$ represent the gradients of the two scales of diffusion prediction tasks, and δ is the weight of the adversarial loss. The final total loss function combines the task losses, cooperative loss, and adversarial loss as follows:

$$\mathcal{L} = \lambda \mathcal{L}_{\text{macro}} + (1-\lambda) \mathcal{L}_{\text{micro}} + \gamma \mathcal{L}_{\text{coop}} + (1-\gamma) \mathcal{L}_{\text{adv}} , \tag{11}$$

where λ is the weight of the task losses, and γ represents the weight coefficients for the cooperative and adversarial losses, respectively.

2.5 Hypergraph Neural Architecture Search

Search Space

Although a graph can be considered a special case of a hypergraph, the search space in existing graph NAS methods cannot be directly applied to hypergraph NAS. Therefore, it is essential to develop a search space specifically tailored for hypergraph neural architecture search. As shown in Table 1, to create an expressive search space suitable for sequential diffusion hypergraphs, we focus on three key components: vertex aggregation, hyperedge aggregation, skip-connection aggregation, and sequential aggregation. We denote the set of vertex aggregators as $\mathcal{O}_v$, the set of skip-connection aggregators as $\mathcal{O}_s$, and the set of temporal sequential aggregators as $\mathcal{O}_t$. In this work, we employ four different methods to aggregate high-order features and original centroid vertex features to generate new centroid vertex features.

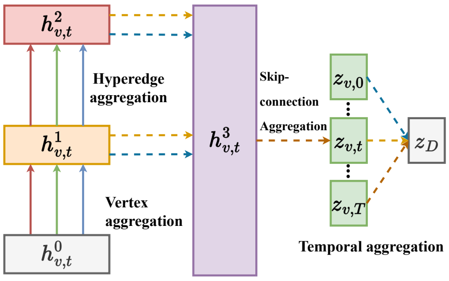


Figure 3: The proposed hypergraph NAS framework, with the supernet being generated from the defined search space.

	Operations
$\mathcal{O}_v$	HGCN, HGAT, SAGE$_{\text{SUM}}$, SAGE$_{\text{MEAN}}$, Trans
$\mathcal{O}_h$	ATT$_{\text{SUM}}$, ATT$_{\text{MEAN}}$, ATT$_{\text{MAX}}$, ATT$_{\text{MLP}}$
$\mathcal{O}_s$	CONCAT, SUM, MEAN, MAX
$\mathcal{O}_t$	RNN, LSTM, GRU

Table 1: Search space for diffusion hypergraphs.

Differentiable Search

To create a continuous search space, as proposed by DARTS (Liu et al., 2018), we replace the categorical selection of a specific operation with a softmax function applied over all possible operations:

$$\bar{o}(x) = \sum_{o \in \mathcal{O}} \frac{\exp\left(\alpha_o\right)}{\sum_{o' \in \mathcal{O}} \exp\left(\alpha_{o'}\right)} o(x), \tag{12}$$

where the operation mixing weights for each node c are parameterized by a vector α of dimension $|\mathcal{O}|$, where $\mathcal{O}$ is drawn from four sets of operations: $\mathcal{O}_v$, $\mathcal{O}_h$, $\mathcal{O}_s$, and $\mathcal{O}_t$, as described in the previous section. The corresponding parameters α_v, $\alpha_{h,}\alpha_s$, and α_t are then computed. In this context, x represents the input features of a given layer. Let

$\bar{o}_v, \bar{o}_h, \bar{o}_s$, and $\bar{o}_t$ denote the mixed operations corresponding to $\mathcal{O}_v, \mathcal{O}_h, \mathcal{O}_s$, and $\mathcal{O}_t$, respectively, as defined in Eq. 12. The vertex aggregation and hyperedge aggregation processes in HyperIDP are then formulated as:

$$\begin{aligned} X_{e,t} &= \bar{o}_v\left(X_{v,t}^{(k)}, \forall X_{v,t}^{(k)} \in e\right), \\ X_{h,t} &= \bar{o}_h\left(X_{e,t}, \forall e \ni c, \forall t \in \{0, \dots, T\}\right), \end{aligned} \tag{13}$$

Then the embedding of the center point c for diffusion hypergraph $G_D^t$is computed as:

$$Z_{v,t} = \bar{o}_s\left(X_{v,t}^{(c)}, X_{h,t}\right), \tag{14}$$

The final representation of the diffusion hypergraph sequence G_D is then obtained through temporal aggregation operations:

$$Z_D = \bar{o}_t\left(Z_{v,t}, \forall t \in \{0, \dots, T\}\right), \tag{15}$$

Upon completing the architecture search, we retain the top-k highest-weighted operations in each module to form the final architecture, which is subsequently fine-tuned using validation data. For the sake of simplicity, we set $k = 1$, thereby replacing each mixed operation $\bar{o}$ with its highest-weighted counterpart, $o = \arg\max_{o \in \alpha} \boldsymbol{\alpha}_o$.

3 Experiments

A thorough experimental analysis is conducted on three real-world datasets to evaluate the effectiveness of the proposed method. This section addresses the following research questions:

- **RQ1: Effectiveness.** How does the proposed method compare to other state-of-the-art (SOTA) approaches in terms of performance? (Section 3.2)
- **RQ2: Modularity.** What is the impact of different components on the overall model performance? (Section 3.3)
- **RQ3: Sensitivity.** How do variations in hyperparameters influence the final performance? (Appendix C.2)

3.1 Experimental Setting

Datasets

Experiments are performed on three datasets: Christianity (Sankar et al., 2020), Android (Sankar et al., 2020), and Douban (Zhong et al., 2012). Table 2 presents the statistics of these datasets, with a detailed description available in the Appendix. Our code can be found at `https://github.com/HowieHsu0126/HyperIDP`.

Dataset	Christ	Android	Douban
# Users	2,897	9,958	12,232
# Links	35,624	48,573	39,658
# Cascades	589	679	3,475
Avg. Length	22.9	33.3	21.76

Table 2: Statistics of datasets. Christ is short for the dataset Christianity.

Baselines

Several representative baseline models are evaluated in comparison to the proposed models. For macroscopic prediction, the following models are analyzed: DeepCas (Li et al., 2017), DeepHawkes (Cao et al., 2017), CasCN (Chen et al., 2019b), and CasFlow (Xu et al., 2023). For microscopic prediction, these models are examined: TopoLSTM (Wang et al., 2017), NDM (Yang et al., 2018), SNIDSA (Wang et al., 2018), Inf-VAE (Sankar et al., 2020), and DyHGCN (Yuan et al., 2021). For multi-scale prediction, these models are considered: FOREST (Yang et al., 2019a) and DMT-LIC (Chen et al., 2019a). This study also incorporates various NAS techniques, including Random search (Li and Talwalkar, 2020), Bayesian-based search (White et al., 2021), and GraphNAS, a reinforcement learning-based NAS approach for GNN (Gao et al., 2021). Detailed descriptions of these baseline models are provided in the Appendix.

3.2 Performance Comparison (RQ1)

For macroscopic prediction, the evaluation metric applied is the Mean Squared Logarithmic Error (MSLE), a method frequently adopted in previous studies. For microscopic prediction, two ranking metrics are used: Mean Average Precision at top k (MAP@k) and Hits Scores at top k (Hits@k), with k values of [10, 50, 100].

A comprehensive evaluation of HyperIDP against multiple baseline models is conducted across three datasets, focusing on both microscopic and macroscopic diffusion prediction tasks. The results, presented in Tables 3, 4, and 5, reveal several key findings: 1) HyperIDP consistently outperforms all state-of-the-art baselines in both macroscopic and microscopic prediction tasks, leveraging sequential hypergraphs to dynamically model cascade interactions, which significantly improves performances. 2) As illustrated in Figure 4, the search cost of HyperIDP is compared with three representative NAS methods, and HyperIDP ex-

Models	Christianity			Android			Douban		
	@10	@50	@100	@10	@50	@100	@10	@50	@100
TopoLSTM (Wang et al., 2017)	0.1548	0.3642	0.4768	0.0471	0.1307	0.2092	0.0317	0.0152	0.0173
NDM (Yang et al., 2018)	0.0475	0.1156	0.1472	0.0181	0.0434	0.0544	0.0379	0.0517	0.0539
SNIDSA (Wang et al., 2018)	0.0651	0.2087	0.3493	0.0282	0.0838	0.1288	0.0713	0.1796	0.2315
Inf-VAE (Sankar et al., 2020)	0.0778	0.2558	0.3844	0.0329	0.0927	0.1443	0.1375	0.2372	0.3048
DyHGCN (Yuan et al., 2021)	0.2391	0.4678	0.5914	0.0737	0.1735	0.2585	0.1449	0.2637	0.3318
FOREST (Yang et al., 2019a)	0.2757	0.4676	0.5592	0.0877	0.1741	0.2325	0.1097	0.1975	0.2561
DMT-LIC (Chen et al., 2019a)	0.2779	0.4431	0.5678	0.0943	0.1648	0.2304	0.1474	0.2517	0.3063
Random (Li and Talwalkar, 2020)	0.1944	0.2026	0.2065	0.0688	0.0705	0.0716	0.1153	0.1208	0.1202
Bayesian (White et al., 2021)	0.1966	0.2048	0.2043	0.0666	0.0727	0.0738	0.1131	0.1187	0.1224
GraphNAS (Gao et al., 2021)	0.1933	0.2015	0.2076	0.0699	0.0704	0.0736	0.1120	0.1210	0.1232
HyperIDP (Ours)	**0.3503**	**0.5289**	**0.6461**	**0.1385**	**0.2298**	**0.3057**	**0.2245**	**0.3376**	**0.3952**

Table 3: The experimental results on three datasets are evaluated using $Hits@k$ score for k = 10, 50, and 100, with higher scores representing superior performance. The best-performing human-designed architectures are underlined, while the highest score on each dataset is highlighted in bold.

Models	Christianity			Android			Douban		
	@10	@50	@100	@10	@50	@100	@10	@50	@100
TopoLSTM (Wang et al., 2017)	0.0534	0.0628	0.0646	0.0177	0.0213	0.0224	0.0343	0.0835	0.0873
NDM (Yang et al., 2018)	0.0155	0.0188	0.0193	0.0068	0.0081	0.0093	0.0132	0.0833	0.0875
SNIDSA (Wang et al., 2018)	0.0257	0.0317	0.0335	0.0111	0.0133	0.0141	0.0362	0.0428	0.0157
Inf-VAE (Sankar et al., 2020)	0.0183	0.0265	0.0281	0.0087	0.0114	0.0121	0.0532	0.0579	0.0607
DyHGCN (Yuan et al., 2021)	0.1073	0.1178	0.1195	0.0383	0.0445	0.0457	0.0810	0.0847	0.0854
FOREST (Yang et al., 2019a)	0.1578	0.1667	0.1681	0.0619	0.0678	0.0686	0.0664	0.0703	0.0711
DMT-LIC (Chen et al., 2019a)	0.1658	0.1739	0.1757	0.0633	0.0643	0.0673	0.0821	0.0867	0.0886
Random (Li and Talwalkar, 2020)	0.1578	0.1667	0.1681	0.0639	0.0653	0.0686	0.0664	0.0703	0.0711
Bayesian (White et al., 2021)	0.1587	0.1676	0.1689	0.0630	0.0674	0.0684	0.0644	0.0683	0.0721
GraphNAS (Gao et al., 2021)	0.1596	0.1685	0.1697	0.0641	0.0658	0.0687	0.0653	0.0714	0.0723
HyperIDP (Ours)	**0.1966**	**0.2048**	**0.2065**	**0.0688**	**0.0725**	**0.0736**	**0.1153**	**0.1208**	**0.1224**

Table 4: The experimental results on three datasets are evaluated using $MAP@k$ score for k = 10, 50, and 100, with higher scores representing superior performance. The best-performing human-designed architectures are underlined, while the highest score on each dataset is highlighted in bold.

Model	Christianity	Android	Douban
DeepCas (Li et al., 2017)	1.435	2.113	2.131
DeepHawkes (Cao et al., 2017)	1.102	1.962	1.734
CasCN (Chen et al., 2019b)	1.037	0.972	1.467
CasFlow (Xu et al., 2023)	0.754	1.032	0.456
FOREST (Yang et al., 2019a)	1.715	0.547	0.834
DMT-LIC (Chen et al., 2019a)	1.681	0.212	0.751
Random (Li and Talwalkar, 2020)	1.780	0.223	0.752
Bayesian (White et al., 2021)	1.683	0.220	0.753
GraphNAS (Gao et al., 2021)	1.690	0.213	0.750
HyperIDP	**0.561**	**0.142**	**0.413**

Table 5: The experimental results on three datasets are evaluated using *MSLE*, with lower scores representing superior performance. The best-performing human-designed architectures are underlined, while the highest score on each dataset is highlighted in bold.

hibits the lowest search cost among all NAS baselines. This efficiency is primarily due to the differentiable search algorithm, which transforms the search space from discrete choices to a continuous optimization problem, enabling the use of gradient information during the search process and facilitating faster convergence to the optimal architecture.

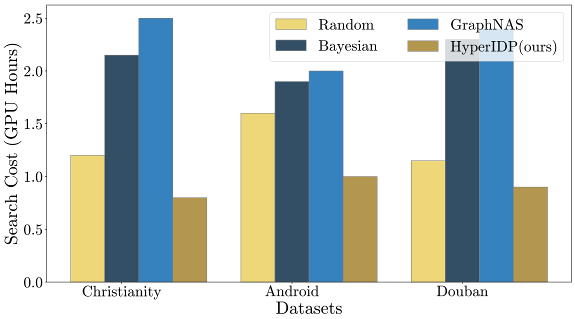


Figure 4: The search cost of each model on all datasets.

Models	Christianity			Douban		
	Hits@100	MAP@100	MSLE	Hits@100	MAP@100	MSLE
w/o Hyper	0.5932	0.2076	1.089	0.3745	0.1221	0.592
w/o Macro	0.5648	0.1938	9.233	0.3729	0.1264	4.674
w/o Micro	0.5946	0.1982	0.852	0.3632	0.1217	0.724
w/o Coop	0.5810	0.1943	0.882	0.3674	0.1206	0.728
w/o Adv	0.5993	0.1967	0.874	0.3625	0.1187	0.739
Vanilla	**0.6315**	**0.2118**	**0.563**	**0.3794**	**0.1259**	**0.532**

Table 6: Ablation studies on the Christianity and Douban datasets assess the contributions of individual submodules within HyperIDP.

3.3 Ablation Study (RQ2)

Table 6 provides the definitions of various model variants: *w/o Hyper* represents the replacement of sequential hypergraphs, *w/o Macro* signifies the exclusion of the macro-level loss function $\mathcal{L}_{\text{macro}}$, *w/o Micro* denotes the omission of the micro-level loss function $\mathcal{L}_{\text{micro}}$, *w/o Coop* indicates the removal of the cooperative loss function $\mathcal{L}_{\text{coop}}$, and *w/o Adv* refers to the elimination of the adversarial loss function $\mathcal{L}_{\text{adv}}$.

The key insights are as follows: 1) The integration of interactive hypergraphs significantly enhances the capture of cascade interactions on a global scale, as shown by the performance of *w/o Hyper*. 2) The macroscopic prediction contributes to refining the microscopic prediction by accurately modeling individual user propagation behaviors, while the microscopic prediction, in turn, sharpens the overall propagation trends captured by the macroscopic prediction. The distinct differences between *w/o Macro*, *w/o Micro*, and HyperIDP in macro and micro indicators underscore the mutual reinforcement between these tasks, leading to superior performance. 3) The cooperative loss (*w/o Coop*) allows the model to adaptively learn complementary representations across different tasks by effectively leveraging inter-task correlations. Furthermore, the adversarial loss (*w/o Adv*) prevents any single task from dominating the shared representation space, thereby maintaining the integrity of the representation space.

4 Conclusion

This paper presents HyperIDP, a multi-scale diffusion prediction model for both microscopic and macroscopic predictions. HyperIDP constructs sequential hypergraphs to capture complex influences and dynamics among cascades from a macro perspective, while simultaneously learning implicit structures and user characteristics within social networks from a micro perspective. To ensure feature integrity, uncertainty-weighted center loss is employed. A search space is developed to tune GNN architectures for both diffusion hypergraphs and social graphs, with optimal designs identified via differentiable search strategies. Experimental results validate the model's effectiveness in predicting next-influenced users and cascade sizes.

5 Limitations

One potential limitation of this work lies in its reliance on the accuracy of the initial hypergraph construction, which might not be robust to noisy or incomplete data. And the computational complexity of the NAS process may limit the scalability of the approach, particularly when applied to large-scale datasets or more complex datasets.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (Nos. U22A2098, 62271411, 62471403, 62261136549, U22B2036), the Fundamental Research Funds for the Central Universities (Nos. G2024WD0151, D5000240309), and the XPLORER PRIZE.

References

Lulwah AlSuwaidan and Mourad Ykhlef. 2016. Toward information diffusion model for viral marketing in business. *International journal of advanced computer science and applications*, 7(2).

Dilyara Baymurzina, Eugene Golikov, and Mikhail Burtsev. 2022. A review of neural architecture search. *Neurocomputing*, 474:82–93.

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

Qi Cao, Huawei Shen, Jinhua Gao, Bingzheng Wei, and Xueqi Cheng. 2020. Popularity prediction on social platforms with coupled graph neural networks. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, page 70–78.

Jingya Chang, Weiyang Ding, Liqun Qi, and Hong Yan. 2018. Computing the p-spectral radii of uniform hypergraphs with applications. *Journal of Scientific Computing*, 75:1–25.

Chaofan Chen, Zelei Cheng, Zuotian Li, and Manyi Wang. 2020. Hypergraph attention networks. In *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 1560–1565.

Xueqin Chen, Kunpeng Zhang, Fan Zhou, Goce Trajcevski, Ting Zhong, and Fengli Zhang. 2019a. Information cascades modeling via deep multi-task learning. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 885–888.

Xueqin Chen, Fan Zhou, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Fengli Zhang. 2019b. Information diffusion prediction via recurrent cascades convolution. In *2019 IEEE 35th international conference on data engineering (ICDE)*, pages 770–781.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3558–3565.

Yang Gao, Hong Yang, Peng Zhang, Chuan Zhou, and Yue Hu. 2021. Graph neural architecture search. In *International joint conference on artificial intelligence*, pages 1403–1409.

Yue Gao, Zizhao Zhang, Haojie Lin, Xibin Zhao, Shaoyi Du, and Changqing Zou. 2020. Hypergraph learning: Methods and practices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(5):2548–2566.

Fuxia Guo, Xiaowen Wang, Yanwei Xie, Zehao Wang, Jingqiu Li, and Lanjun Wang. 2024. A survey of datasets for information diffusion tasks. *arXiv preprint arXiv:2407.05161*.

Sheng Huang, Ahmed Elgammal, and Dan Yang. 2017. On the effect of hyperedge weights on hypergraph learning. *Image and Vision Computing*, 57:89–101.

Mahdi Jalili and Matjaž Perc. 2017. Information cascades in complex networks. *Journal of Complex Networks*, 5(5):665–693.

Jianwen Jiang, Yuxuan Wei, Yifan Feng, Jingxuan Cao, and Yue Gao. 2019. Dynamic hypergraph neural networks. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 2635–2641.

Pengfei Jiao, Hongqian Chen, Qing Bao, Wang Zhang, and Huaming Wu. 2024. Enhancing multi-scale diffusion prediction via sequential hypergraphs and adversarial learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 8571–8581.

Hai Jin, Yao Wu, Hong Huang, Yu Song, Haohui Wei, and Xuanhua Shi. 2022. Modeling information diffusion with sequential interactive hypergraphs. *IEEE Transactions on Sustainable Computing*, 7(3):644–655.

Jisu Kim, Jihwan Aum, SangEun Lee, Yeonju Jang, Eunil Park, and Daejin Choi. 2021. Fibvid: comprehensive fake news diffusion dataset during the covid-19 period. *Telematics and Informatics*, 64:101688.

Cheng Li, Jiaqi Ma, Xiaoxiao Guo, and Qiaozhu Mei. 2017. Deepcas: An end-to-end predictor of information cascades. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, page 577–586, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Liam Li and Ameet Talwalkar. 2020. Random search and reproducibility for neural architecture search. In *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, pages 367–377.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2018. DARTS: Differentiable architecture search. In *International Conference on Learning Representations*, pages 1–13.

Yuqiao Liu, Yanan Sun, Bing Xue, Mengjie Zhang, Gary G. Yen, and Kay Chen Tan. 2023. A survey on evolutionary neural architecture search. *IEEE Transactions on Neural Networks and Learning Systems*, 34(2):550–570.

Babatounde Moctard Oloulade, Jianliang Gao, Jiamin Chen, Tengfei Lyu, and Raeed Al-Sabri. 2021. Graph neural architecture search: A survey. *Tsinghua Science and Technology*, 27(4):692–708.

Zheyi Pan, Songyu Ke, Xiaodu Yang, Yuxuan Liang, Yong Yu, Junbo Zhang, and Yu Zheng. 2021. AutoSTG: Neural architecture search for predictions of spatio-temporal graph. pages 1846–1855.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. 2021a. A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Computing Surveys (CSUR)*, 54(4):1–34.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. 2021b. A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Computing Surveys (CSUR)*, 54(4):1–34.

Aravind Sankar, Xinyang Zhang, Adit Krishnan, and Jiawei Han. 2020. Inf-vae: A variational autoencoder framework to integrate homophily and influence in diffusion prediction. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, WSDM '20, page 510–518, New York, NY, USA. Association for Computing Machinery.

Ling Sun, Yuan Rao, Xiangbo Zhang, Yuqian Lan, and Shuanghe Yu. 2022. Ms-hgat: memory-enhanced sequential hypergraph attention network for information diffusion prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 4156–4164.

Xigang Sun, Jingya Zhou, Ling Liu, and Zhen Wu. 2023. Castformer: A novel cascade transformer towards predicting information diffusion. *Information Sciences*, 648:119531.

Jia Wang, Vincent W Zheng, Zemin Liu, and Kevin Chen-Chuan Chang. 2017. Topological recurrent neural network for diffusion prediction. In *2017 IEEE international conference on data mining (ICDM)*, pages 475–484.

Zhitao Wang, Chengyao Chen, and Wenjie LI. 2018. A sequential neural information diffusion model with structure attention. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, page 1795–1798.

Colin White, Willie Neiswanger, and Yash Savani. 2021. BANANAS: Bayesian optimization with neural architectures for neural architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10293–10301.

Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. 2022. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys*, 55(5):1–37.

Xovee Xu, Fan Zhou, Kunpeng Zhang, Siyuan Liu, and Goce Trajcevski. 2023. CasFlow: Exploring hierarchical structures and propagation uncertainty for cascade prediction. *IEEE Transactions on Knowledge and Data Engineering*, 35(4):3484–3499.

Cheng Yang, Maosong Sun, Haoran Liu, Shiyi Han, Zhiyuan Liu, and Huanbo Luan. 2018. Neural diffusion model for microscopic cascade prediction. *arXiv preprint arXiv:1812.08933*.

Cheng Yang, Jian Tang, Maosong Sun, Ganqu Cui, and Zhiyuan Liu. 2019a. Multi-scale information diffusion prediction with reinforced recurrent networks. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4033–4039. ijcai.org.

Pei Yang, Qi Tan, Jieping Ye, Hanghang Tong, and Jingrui He. 2019b. Deep multi-task learning with adversarial-and-cooperative nets. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence, IJCAI 2019*, pages 4078–4084.

Liu Yu, Xovee Xu, Goce Trajcevski, and Fan Zhou. 2022. Transformer-enhanced hawkes process with decoupling training for information cascade prediction. *Knowledge-Based Systems*, 255:109740.

Chunyuan Yuan, Jiacheng Li, Wei Zhou, Yijun Lu, Xiaodan Zhang, and Songlin Hu. 2021. Dyhgcn: A dynamic heterogeneous graph convolutional network to learn users' dynamic preferences for information diffusion prediction. In *Machine Learning and Knowledge Discovery in Databases*, volume 12459, pages 347–363. Springer International Publishing.

Haoming Zhang, Yiping Yao, Wenjie Tang, Jiefan Zhu, and Yonghua Zhang. 2023. Opinion-aware information diffusion model based on multivariate marked hawkes process. *Knowledge-Based Systems*, 279:110883.

Yu Zhang and Qiang Yang. 2018. An overview of multi-task learning. *National Science Review*, 5(1):30–43.

Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE transactions on knowledge and data engineering*, 34(12):5586–5609.

Huan Zhao, Quanming Yao, and Weiwei Tu. 2021. Search to aggregate neighborhood for graph neural network. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, volume 32, pages 552–563.

Erheng Zhong, Wei Fan, Junwei Wang, Lei Xiao, and Yong Li. 2012. Comsoc: Adaptive transfer of user behaviors over composite social network. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, page 696–704, New York, NY, USA. Association for Computing Machinery.

A Related Works

A.1 Macroscopic Diffusion Prediction

Previous research can be classified into three primary approaches: feature-based, generative process-based, and deep learning-based methods (Guo et al., 2024). Feature-based methods focus on extracting handcrafted features from input data, which are then applied to machine learning algorithms for regression or classification tasks. However, these techniques heavily depend on domain knowledge and tend to lack generalizability. Generative process-based approaches model the spread of infected users as a point process, improving interpretability but often missing implicit information within cascade dynamics (Yu et al., 2022; Zhang et al., 2023). Recently, deep learning-based methods have demonstrated significant effectiveness. For instance, DeepCas (Li et al., 2017) uses recurrent neural networks (RNNs) to encode sampled sequences from social graphs and cascades, while DeepHawkes (Cao et al., 2017) integrates the Hawkes process into an RNN framework. Other approaches, such as CoupledGNN (Cao et al., 2020) and CasCN (Chen et al., 2019b), leverage graph neural networks (GNNs) to capture diffusion patterns across social networks.

A.2 Microscopic Diffusion Prediction

Microscopic diffusion prediction methods are commonly classified into three categories: independent cascade (IC)-model-based approaches, embedding-based approaches, and deep learning-based approaches (Guo et al., 2024). IC-model-based methods assume independent diffusion probabilities between user pairs and use Monte Carlo simulations for prediction. Embedding-based approaches extend the IC model by representing each user as a parameterized vector, modeling diffusion probabilities based on factors such as global user similarity (Jalili and Perc, 2017). However, these methods often fail to account for infection history. Deep learning techniques offer more advanced solutions, with models like TopoLSTM (Wang et al., 2017) structuring hidden states as directed acyclic graphs. NDM (Yang et al., 2018) combines self-attention with convolutional neural networks, while Inf-VAE (Sankar et al., 2020) uses a variational autoencoder framework to capture social homophily and temporal influence. Other methods, such as SNIDSA (Wang et al., 2018) and DyHGCN (Yuan et al., 2021), utilize diffusion paths, social networks, and temporal data for enhanced prediction. Additionally, models like MS-HGAT (Sun et al., 2022) employs hypergraphs to capture global user dependencies.

A.3 Hypergraph Learning

Hypergraph learning is initially introduced as a label propagation method for semi-supervised learning. This approach seeks to minimize label discrepancies among vertices connected by the same hyperedge (Gao et al., 2020). Various methods for constructing hypergraphs, including the k-NN method (Jiang et al., 2019) and the spectral radius method (Chang et al., 2018), have been explored. Recent research has focused on optimizing hyperedge weights, assigning greater weight to hyperedges or sub-hypergraphs of higher significance (Huang et al., 2017). In addition to label propagation, dynamic hypergraph structure learning employs a dual optimization process to learn the hypergraph structure (Chang et al., 2018). Hypergraph neural networks (HGNNs) represent the first deep learning method for hypergraphs, using the hypergraph Laplacian to model hypergraphs from a spectral perspective (Feng et al., 2019). Although HGNNs have achieved notable success, their architecture design typically relies heavily on domain expertise. To address this, the proposed approach employs Neural Architecture Search (NAS) to automatically identify optimal feature aggregation operators for hypergraph learning, eliminating the need for manual design.

A.4 Neural Architecture Search

Contemporary scholarship is increasingly concentrating on NAS, acclaimed for its capacity to independently generate neural architectures that outperform those designed by humans (Ren et al., 2021b; Baymurzina et al., 2022). However, the adaptation of NAS for GNN presents complexities (Oloulade et al., 2021). A pivotal element of NAS involves the delineation of the search space, which substantially influences the effectiveness and efficiency of the search algorithms (Gao et al., 2021). Such a search space typically encompasses all pertinent GNN hyperparameters, including hidden embedding size, aggregation functions, and the number of layers. Traditional approaches involve a trial-and-error methodology, initiating with the sampling of a candidate architecture followed by its training from the ground up (Gao et al., 2021; Pan et al., 2021; Liu et al., 2023). This technique necessitates extensive periods for training a multitude of architectures throughout the search process. Recently, the focus of NAS has shifted towards differentiable methods due to their enhanced efficiency (Zhao et al., 2021). These methods employ an over-parameterized network, or supernet, within a cohesive framework and search space, facilitating the incorporation of existing methodologies.

B Experimental Details

B.1 Datasets

Three datasets are employed in the experiments: Christianity, Android, and Douban.

- **Christianity** (Sankar et al., 2020) comprises a user friendship network and cascading interactions focused on Christian themes, collected from Stack Exchange.
- **Android** (Sankar et al., 2020) is derived from Stack Exchange and includes user interactions across various channels, forming their friendship networks.
- **Douban** (Zhong et al., 2012) is a Chinese social platform where users can update and share their book reading statuses, as well as follow the activities of others.

B.2 Baselines

The following representative baseline models are compared with the proposed models:

Macroscopic Prediction Models:

- **DeepCas** (Li et al., 2017) transforms cascade graphs into node sequences through random walks and learns representations for each cascade using a deep learning framework.
- **DeepHawkes** (Cao et al., 2017) integrates end-to-end deep learning with the Hawkes process for cascade prediction.
- **CasCN** (Chen et al., 2019b) applies graph convolutional networks (GCNs) to capture the structural patterns of information diffusion and utilizes LSTM to learn the sequential dependencies of users' retweeting behaviors in cascades.
- **CasFlow** (Xu et al., 2023) employs normalizing flows to learn node-level and cascade-level latent factors, enabling hierarchical pattern learning in information diffusion.

Microscopic Prediction Models:

- **TopoLSTM** (Wang et al., 2017) extends the standard LSTM model to simulate the information diffusion process in social networks.
- **NDM** (Yang et al., 2018) utilizes CNN to capture users' diffusion representations and employs self-attention for diffusion prediction.
- **SNIDSA** (Wang et al., 2018) jointly learns heterogeneous information representations by exploring diffusion paths and social network structures.
- **Inf-VAE** (Sankar et al., 2020) incorporates social homophily through graph neural networks (GNNs) and employs a co-attentive fusion network to integrate social and temporal variables.
- **DyHGCN** (Yuan et al., 2021) learns the structural and dynamic properties of social and diffusion graphs, encoding temporal information into a heterogeneous graph to capture dynamic user preferences.

Unified Multi-scale Prediction Models:

- **FOREST** (Yang et al., 2019a) integrates macroscopic information into an RNN-based microscopic diffusion model to predict both microscopic and macroscopic diffusion simultaneously.
- **DMT-LIC** (Chen et al., 2019a) employs a shared representation layer to capture the underlying structure of cascade graphs and the node sequence in the diffusion process.

NAS Approaches:

- **Random** (Li and Talwalkar, 2020) NAS is a technique for discovering optimal neural network architectures through a randomized exploration process. Unlike traditional methods that rely on domain expertise or structured algorithms like evolutionary strategies and reinforcement learning, Random NAS generates neural network architectures by randomly sampling from a predefined search space.
- **Bayesian** (White et al., 2021) NAS applies Bayesian optimization (BO) to efficiently explore neural architecture search spaces. Traditional NAS methods can be computationally expensive, as they often require training a large number of architectures to find an optimal design. BANANAS addresses this by integrating BO with a neural predictor model, enabling more sample-efficient search and reducing the computational cost.

- **GraphNAS** (Chen et al., 2019a) is a specialized NAS approach designed to discover optimal architectures for GNNs using reinforcement learning (RL). GNNs are particularly useful for learning over graph-structured data, such as social networks, chemical molecules, and knowledge graphs. GraphNAS automates the design of GNN architectures, which traditionally involves significant manual effort and domain expertise.

B.3 Experimental Settings

Each dataset is randomly sampled, with 80% of the cascades allocated for training, 10% for validation, and 10% for testing. Baseline methods are implemented according to their original settings. The MINDS model is developed using PyTorch, with the Adam optimizer applied at a learning rate of 0.001. The embedding dimension is fixed at 64, and the batch size is 32. The balance parameter λ is set to 0.3, while the hyperparameter γ is configured to 0.05. Social homophily learning is conducted using a 2-layer GCN, and global interaction learning is achieved via a single-layer HGNN. The number of time intervals is specified as 8.

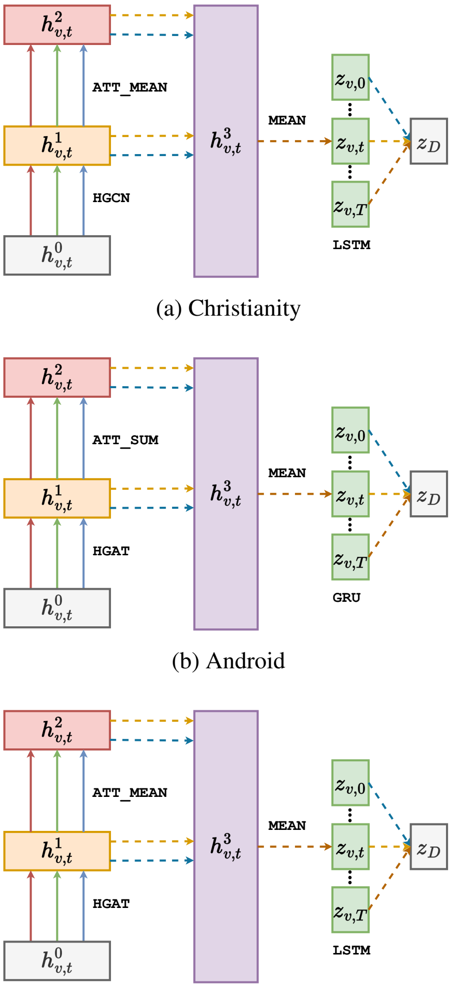
C Additional Experiments

C.1 Searched Architectures

The top-1 architectures identified by HyperIDP across various datasets are visualized in the Figure 5. These architectures demonstrate clear data-dependence and introduce novel designs to the literature. The inclusion of skip-connections proves to significantly impact performance. Moreover, attention-based vertex aggregators, being more expressive than non-attentive counterparts, are more frequently utilized, making HGAT the preferred choice (Chen et al., 2020).

C.2 Parameter Analysis (RQ3)

This subsection investigates the influence of various hyperparameter settings on the model's performance using the Android and Douban datasets. To analyze the sensitivity of embedding size and the number of time intervals, each parameter is varied individually while others remain fixed. Figure 6 illustrates the model's performance in multi-scale prediction across different hyperparameter configurations. During parameter selection, both macro and micro indicators are carefully evaluated; optimal performance is achieved when the macro index is minimized and the micro index is maximized. Remarkably, HyperIDP exhibits stable performance when hyperparameters are adjusted within a reasonable range, indicating strong robustness. Consequently, the optimal hyperparameter configuration is determined to be (λ, γ, number of time intervals) = (0.5, 0.25, 8).

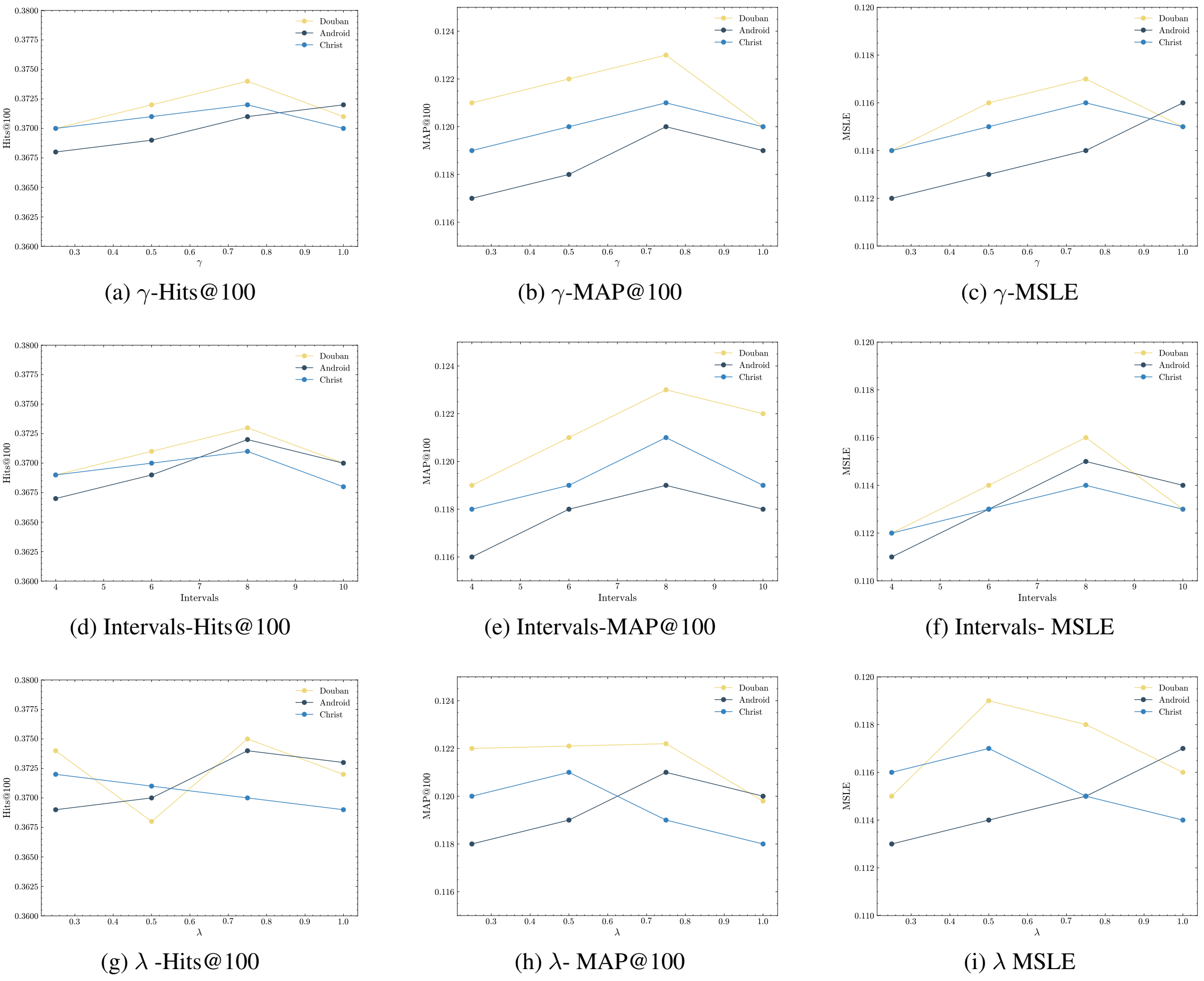


(a) Christianity

(b) Android

(c) Douban

Figure 5: The searched architectures by HyperIDP on different datasets.



(a) γ-Hits@100

(b) γ-MAP@100

(c) γ-MSLE

(d) Intervals-Hits@100

(e) Intervals-MAP@100

(f) Intervals- MSLE

(g) λ -Hits@100

(h) λ- MAP@100

(i) λ MSLE

Figure 6: The parameter sensitivity is evaluated on the Douban and Android datasets. For the maroc-micro balance parameter $\lambda \in \{0.25, 0.5, 0.75, 1.0\}$, the number of time intervals in the range $[2, 12]$, and the cooperative-adversarial balance parameter $\gamma \in \{0.25, 0.5, 0.75, 1.0\}$, the performance metrics considered are MAP@100, MAP@100, and MSLE.