Influence Maximization in Temporal Social Networks with a Cold-Start Problem: A Supervised Approach

Laixin Xie[1]*, Ying Zhang[1], Xiyuan Wang[1], Shiyi Liu[2], Shenghan Gao[1], Xingxing Xing[3], Wei Wan[3], Haipeng Zhang[1], Quan Li[1]†

[1]School of Information Science and Technology, ShanghaiTech University
[2]Arizona State University
[3]UX Center, NetEase Game
{xielx, zhangying12022, gaoshh1,zhanghp,liquan}@shanghaitech.com,
shiyiliu@asu.edu,
{xingxingxing, gzwanwei}@corp.netease.com

Abstract

Influence Maximization (IM) in temporal graphs focuses on identifying influential "seeds" that are pivotal for maximizing network expansion. We advocate defining these seeds through Influence Propagation Paths (IPPs), which is essential for scaling up the network. Our focus lies in efficiently labeling IPPs and accurately predicting these seeds, while addressing the often-overlooked cold-start issue prevalent in temporal networks. Our strategy introduces a motif-based labeling method and a tensorized Temporal Graph Network (TGN) tailored for multi-relational temporal graphs, bolstering prediction accuracy and computational efficiency. Moreover, we augment cold-start nodes with new neighbors from historical data sharing similar IPPs. The recommendation system within an online team-based gaming environment presents subtle impact on the social network, forming multi-relational (i.e., weak and strong) temporal graphs for our empirical IM study. We conduct offline experiments to assess prediction accuracy and model training efficiency, complemented by online A/B testing to validate practical network growth and the effectiveness in addressing the cold-start issue.

Introduction

Virtual social networks like Twitter are pivotal extensions of physical social relationships, connecting billions of users and profoundly shaping human society. Influence Maximization (IM) (Li et al. 2023) aims to expand network scales within a given diffusion model. IM finds diverse applications, from interrupting the spread of COVID-19 (Cheng, Kuo, and Zhou 2020) to enhancing viral marketing strategies (Castiglione et al. 2020) and facilitating rumor control (Vega-Oliveros, da Fontoura Costa, and Rodrigues 2020). At its core, IM identifies active members, or "seeds", with robust propagation capabilities, initiating the diffusion process within the network. Strategically pinpointing and leveraging these influential nodes enables maximal impact within the network, serving various societal and strategic objectives.

When applied to temporal graphs (Yanchenko, Murata, and Holme 2023), IM maintains its primary objective while accounting for the chronological order of edge establishment, introducing complexity due to evolving connections over time. IM is known for its NP-hard nature, and the greedy algorithm is a traditional method for approximating the optimal solution (Yanchenko, Murata, and Holme 2023). In contrast, *SPEX* (Li et al. 2021) motivates us a supervised pipeline aimed at expanding the network scale, which is the ultimate goal of IM. This approach explicitly defines active members, thereby bypassing the need to address the NP-hard problem directly. To be specific, we adopt Influence Propagation Path (IPP) from SPEX, wherein the initial node forms social connections with multi-hop friends over consecutive timestamps. In the context of the IM problem, the initial node of any IPP inherently qualifies as an "active member". In this supervised context, accurately predicting these labeled active members becomes crucial for network expansion. Thus, the challenges of *efficiently labeling IPPs and accurately predicting active members* ($\mathbf{\Gamma 1}$) emerge as central considerations. In addition, the IM problem faces the often-overlooked cold-start issue, characterized by sparse social information at the outset, leading to performance declines for initial nodes. During the diffusion process, the initial network scale is smaller than the final scale, indicating the presence of cold-start nodes. This issue recurs across timestamps in temporal graphs, adds complexity. Essentially, *the IM problem in temporal graphs is significantly affected by a severe and persistent cold-start problem* ($\mathbf{\Gamma 2}$). Despite extensive exploration in social recommendation (Panda and Ray 2022), the cold-start issue has received limited attention in the context of IM.

In this study, we tackle the above two challenges ($\mathbf{\Gamma 1}$ and $\mathbf{\Gamma 2}$) with the following solutions. First, to address the first challenge ($\mathbf{\Gamma 1}$), we introduce Motif-Based Filtering (MF), streamlining the identification of all IPPs by using known initiators as ground truth. Additionally, we implement a Unified Temporal Graph Network (TGN), specifically tailored for multi-relational temporal graphs. This TGN incorporates edge establishment, conversion, and diminishing, ensuring accurate predictions of active members. By transforming op-

*This work was completed during an internship at NetEase.
†Corresponding author.

erations into equivalent tensor operations, our TGN implementation significantly enhances efficiency and scales effectively with larger datasets. Second, to address the cold-start problem ($\mathbf{\Gamma 2}$), we efficiently augment cold-start nodes by adding new neighbors from historical timestamps that share similar IPPs. To expedite the matching of similar IPPs, we encode IPPs as strings and construct a prefix tree based on these strings.

In our experimental design, we conducted an empirical study through A/B testing within an online team-based game, utilizing our IM solution to enhance the game's native social recommendation system's coverage. The game's temporal multi-relational graphs, formed by edge timestamps and strong and weak relationships, perfectly align with our desired scenario. Augmenting the game's network scale emerges as a pivotal factor in boosting the impact of its built-in social recommendations. We selected this game due to its robust socialization features, which facilitated comprehensive data collection and empirical study. The offline experiment systematically highlighted the prediction accuracy of our approach, while the online experiment evaluated real-world network scale growth. Results from both experiments consistently affirm our method's superior performance. In the online experiment, our method demonstrates a $\underline{3.52\%}$ overall improvement and a $\underline{14.32\%}$ improvement in cold-start scenarios for spreading, compared to all other evaluated methods. In summary, our contributions are as follows:

- **Efficient Labeling Methods:** We focus on a subset of active members and introduce efficient labeling methods to enhance IM adaptability. Additionally, our implementation achieves up to a 22-fold speedup in TGN training under our experimental settings, providing a practical alternative to current high-performance GNN frameworks.
- **Addressing the Cold-Start Issue:** We propose a method to efficiently provide neighborhood information for cold-start nodes, effectively tackling this issue in IM.
- **Practical Application and Validation:** We apply our approach to a specific practical problem, demonstrating its effectiveness in expanding network scale through A/B testing in an online experiment.

Related Work

Influence Maximization (IM) and Temporal Graph

IM involves a diffusion model delineating the diffusion process and an algorithm tailored to identify active members. The diffusion simulation typically relies on models like the Independent Cascade (IC) or Linear Threshold (LT) (Li et al. 2023). These models estimate the probability of diffusion based on data from the downstream domain, simulating the network's post-diffusion state. Greedy algorithms are commonly employed to seek suboptimal solution for this NP-hard problem (Feng et al. 2023; Zhang et al. 2023). Our focus lies on addressing the IM problem within temporal graphs, particularly exploring supervised solutions. IM in temporal graphs accounts for temporal variations by considering both the formation and disappearance of edges. Consequently, it grapples with challenges posed by NP-hard complexities and graph dynamics. Recent surveys (Yanchenko, Murata, and Holme 2023) include various traditional solutions, approximation algorithms (Erkol, Mazzilli, and Radicchi 2020), and heuristic node ranking methods (Michalski, Jankowski, and Pazura 2020), which diverge from our proposed solutions.

Deep Learning (DL) for IM

DL solutions for IM have gained prominence due to their effectiveness in handling large-scale networks. In static graphs, Wang et al. (Wang et al. 2021) conducted empirical studies employing graph embedding via unsupervised learning. They initially employed *struc2vec* (Ribeiro, Saverese, and Figueiredo 2017) to generate node embeddings with structural semantics, followed by clustering techniques to identify active members. For temporal graphs, recent approaches in dynamic IM (i.e., IM on temporal graph), as discussed in a survey (Li et al. 2023), draw inspiration from either the reinforcement learning (RL) framework (Meirom et al. 2021; Mendonça, Barreto, and Ziviani 2020) or from updating embeddings based on graph dynamics (Peng 2021). In supervised learning, methods either utilize proxies for active members (Qiu et al. 2018) or generate labels based on diffusion models (Kumar et al. 2022; Kumar, Mallik, and Panda 2023) before applying an improved prediction model. Our supervised learning approach builds on existing research for label generation but introduces a novel element by incorporating multi-relational temporal graphs.

Research Gap

This study addresses several research gaps within the domain of IM on temporal graphs. Notably, there has been limited exploration of supervised learning methods in this area. Our approach adheres to the traditional IM framework but introduces a novel labeling technique based on motif identification, effectively addressing our first set of challenges. Moreover, the cold-start issue has received little recent attention in the IM domain (Panda and Ray 2022). Existing solutions (Ojagh, Malek, and Saeedi 2020; Herce-Zelaya et al. 2020; Bedi et al. 2020; Wang et al. 2020) often rely on supplementary information beyond the primary data. Temporal-aware approaches (Wei et al. 2017; Zhang and Liu 2015; Chalyi, Leshchynskyi, and Leshchynska 2019) primarily focus on user-item scenarios. In contrast, our solution innovatively acquires neighbors from observed timestamps, utilizing solely the inherent temporal social networks. Consequently, this cold-start solution for the IM problem addresses our second set of challenges.

Preliminary and Background

In real-world scenarios, social relationships among people change dynamically over time. At a given timestamp t, a social network is represented as $G_t = (V_t, E_t)$, where V_t denotes the individuals (nodes) and E_t denotes their relationships (edges). For any edge $e^t_{i,j} \in E^t$, it connects nodes v^t_i and v^t_j within V_t. If we consider one week as a time unit, a temporal graph includes all nodes and edges that occur within that week. The neighbors of a node v^t_i at timestamp t are defined as $N^t_i = \bigcup_{\forall e^t_{i,j}} v_j$, which includes all

nodes connected to v_i at timestamp t. Furthermore, we define $N^t(V) = \bigcup_{v_i \in V} N_i^t$ to facilitate subsequent discussions.

Diffusion and Influence Maximization. In a social network, diffusion refers to a recursive propagation process from a node to its neighbors, forming a rapidly expanding diffusion network. Over timestamps from 0 to $T-1$, the network scale is defined as $\sigma(S, T-1) = |\bigcup_{t=0}^{T-1} V_t|$, where $S = V_0$. The goal of IM is to maximize the network scale by selecting seeds S (i.e., active members) as the initial nodes of G_0. For simplicity, we start our timestamp from 0, and the number of seeds remains constant to ensure a fair evaluation. Therefore, the IM problem on a temporal graph can be generally defined as: $S^* = \arg\max_{|S|=K} \sigma(S, T)$, where K is the fixed number of seeds (Yanchenko, Murata, and Holme 2023).

In-game Recommendation and Propagation. On the gaming platform, we focus on team creation through the *in-game teammate recommendation system* to study how its exclusive contribution to teaming behavior. This feature is fundamental across various game modes, supporting both one-on-one team fights and multi-team battles. The recommendation system suggests compatible teammates, thereby enhancing the gaming experience and improving player retention. Subsequently, three types of social relationships are derived: **exposure, invitation and adoption**. Exposure occurs when players appear on someone's recommendation list, allowing invitations to be sent. If an invitation is accepted, adoption occurs, and the invited player joins the inviter's team.

Motivated Insights

Data from a team-versus-team game developed by NetEase Games[1] indicates that only 25.2% of players used the social recommendation system (sending invitations or adopting system recommendations) over a month-long period. This low engagement suggests that conventional social recommendations are inadequate for most players. Additionally, network scale analysis during the same period shows a significant decline in growth rate: an initial 47.89% increase drops to 12.05% after the first week and further reduces to 2.5% by the end of the month, as shown in Figure 1(a). This trend suggests the network scale will eventually stabilize and converge to a constant value. These findings imply that the recommendation system only effectively reaches a limited number of players, resulting in a minimal impact on the overall social network.

Observation 1 ***LIMITED-COVERAGE RECOMMENDATION****: The recommendation system has a limited impact on the entire social network.*

Our investigation into the evolution of neighbor count across timestamps reveals a persistent "cold-start" issue in temporal graphs. As the network scale expands during diffusion in the IM problem, the number of neighbors for each node increases. However, the ratio of nodes with only one neighbor for social relationships ranges from 12.97% to 19.44%, as depicted in Figure 1(b). This underscores that nodes with few neighbors, reflective of the cold-start issue, constitute a substantial portion of the network. Such nodes can hinder performance when methods depend on social information from their neighborhood. Contrary to the assumption that the cold-start issue is confined to initial timestamps, our findings demonstrate its persistence throughout propagation in IM problems, leading to compromised diffusion estimation for affected nodes.

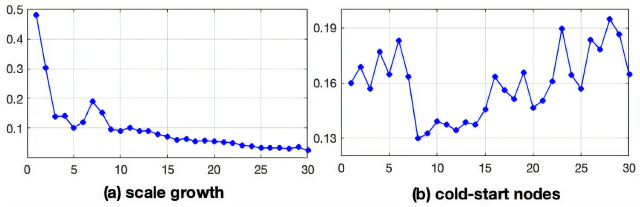

Figure 1: In the graph constructed from deduplicated edges derived from daily invitations and adoptions, the proportion of nodes with only one neighbor over the total number of nodes (y-axis) is presented over a span of 30 days (x-axis).

Observation 2 ***PERSISTENT COLD-START****: The consecutive presence of the cold-start issue across timestamps results in an underestimated propagation process.*

Problem Definition

The observed decline in growth, as highlighted in *Observation 1*, is attributed to recommending acquaintances rather than players with the potential to expand the network. Such recommendations often lead to internal propagation, commonly known as the "information cocoon" (Peng and Liu 2021). Consequently, *Observation 1* prompts us to focus on recommending specific types of active players who can facilitate broader propagation. This supervised approach differs from most existing IM solutions, which often fail to explain the patterns crucial for expanding network scale. To introduce external propagation and break the cocoon, we classify relationships formed through recommendation-driven invitations/adoption as "strong" relationships, while the exposure in the system is deemed "weak" (*Definition 1*). Any strong relationships must occur after exposure, making the weak graph an upper bound (*Constraint 1*) for IM solutions.

Definition 1 *Given a temporal graph $G_t(V_t, E_t)$, the strong graph $G_t^s(V_t^s, E_t^s)$ is a subgraph where $V_t^s \subseteq V_t$ and $E_t^s \subseteq E_t$, such that $\forall e^t(v_i^t, v_j^t) \in E_t^s$, both v_i^t and v_j^t belong to V_t^s. Let $G_t^w(V_t^w, E_t^w) = G_t(V_t, E_t)$ represent the weak graph.*

Constraint 1 $|\sigma(S^*, T)| \leq \left|\bigcup_{t=0}^{T-1} V_t^w\right|, S^* \in V_0^s$*, where S^* is the IM solution.*

The distinction between strong and weak relationships has been extensively studied in previous literature (Granovetter 1973; Onnela et al. 2007), denoting high and low establishment probabilities. Traditional recommendation systems

[1]NetEase Games is a leading global developer and publisher of video game IP across a variety of genres and platforms.

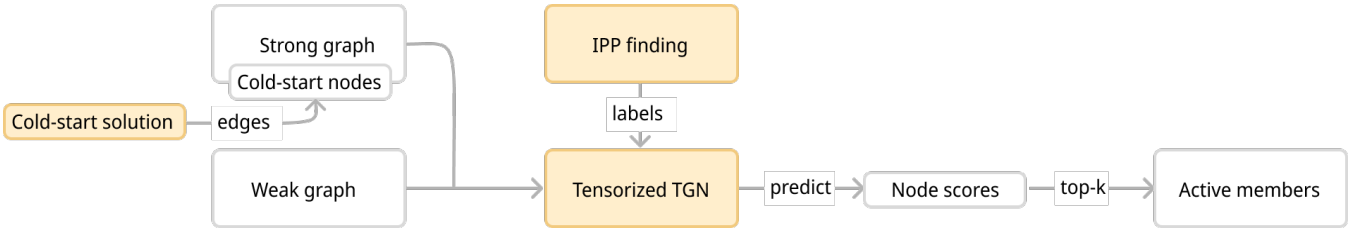

Figure 2: Pipeline for predicting active members. Weak and strong graphs are constructed using exposure edges and invitation and adoption edges derived from in-game teammate recommendations, respectively. These graphs are then input into a Tensorized TGN to generate node scores. During training, node scores with labels from IPP findings are used to calculate the training loss. In the inference phase, nodes with top scores are active members.

typically consider strong relationships as positive edges and weak relationships as negative edges in unsupervised learning (Hamilton, Ying, and Leskovec 2017). However, it's important to recognize that strong relationships can evolve from weak ones through players' invitation/adoption behavior. Weak relationships play a crucial role in setting an upper bound for the network scale achievable by IM solutions.

Inspired by *Observation 2*, we formulate the cold-start issue as another constraint for the IM problem, shaping our approach to addressing this challenge.

Definition 2 *Given a strong graph $G_t^s(V_t^s, E_t^s)$, we define a cold-start node as $\exists c^t \in V_t^s$ such that $|N^t(c_t)| \leq C$, where C is a constant (e.g., 1). Let $\widehat{C}_t$ represent the universal set that includes all c^t at timestamp t.*

Constraint 2 *$\forall t \in [0, T-1], \left|\widehat{C}_t\right| \geqslant \delta$, where δ is the minimal number of cold-start nodes through statistics over T temporal graph.*

These nodes are considered minimal because increasing C would naturally include more nodes, thus broadening the definition of cold-start nodes.

Under these constraints, our IM problem is defined as:

$$\begin{aligned} &\underset{|S|=k}{\text{maximize}} && \sigma(S,T) \\ &\text{subject to} && \textit{Constraint 1}, \\ &&& \textit{Constraint 2} \end{aligned} \tag{1}$$

We argue that traditional metrics, such as adoption rates (Zhang et al. 2023), do not effectively reach players outside the coverage of the recommendation system. Therefore, a primary focus of our research is to expand the limited coverage of the teammate recommendation system. These two objectives are not in direct competition.

Algorithmic Solution & Implementation

In this study, we formulate the IM problem as a supervised learning task, shown in Figure 2. In the following sections, we first introduce the problem formulation and the efficient labeling of IPP findings. Next, we provide an overview of the TGN architecture used for label prediction, highlighting modifications to accelerate training and adapt to large-scale datasets. Lastly, we explore the serialization of IPPs and an efficient retrieval method designed to address the cold-start issue.

Supervised IM and Efficient Labeling

Before formulating IM as a supervised problem, we first introduce IPPs within both the strong and weak graphs:

Definition 3 *An Influence Propagation Path (IPP) is defined as a sequence of edges $< e^{t_0}_{v_0,v_1}, e^{t_1}_{v_1,v_2}, \ldots, e^{t_{N_p-1}}_{v_{N_p-1},v_{N_p}} >$ that adheres to the edge condition $t_0 \leq t_1 \leq \cdots \leq t_{N_p-1}$ and the node condition $\forall i \in [0, N_p-2], v_i \in V^s_{t_i}$, while $v_{i+1}, v_{i+2} \notin V^s_{t_i}$. Here, N_p signifies the length of the IPP.*

Supervised IM. We highlight an important modification from the original definition, which revolves around the node constraint regarding the strong and weak graph, derived from CONSTRAINT 1. This study particularly focuses on the simplest case where $N_p = 2$, aiming to formulate a supervised learning problem using these IPPs as ground truth. We first train our model to predict IPPs, without seeking the IM solution for the observed data. Subsequently, during inference, the inference data are treated as the initial network. The trained model then predicts the seeds for maximizing the subsequent diffusion network.

IPP Findings. Identifying all IPPs consumes a considerable amount of time, prompting the use of a sampling strategy as a compromise to access a subset of labels, as demonstrated in a prior study (Li et al. 2021). To efficiently identify entire labels, we introduce a motif-based filtering (MF) method according to DEFINITION 3. The MF method begins with a motif finding algorithm aimed at detecting all 2-hop paths $< e^{t_0}_{v_0,v_1}, e^{t_1}_{v_1,v_2} >$. Subsequently, these paths undergo filtering based on two conditions: the edge condition ($t_0 \leq t_1$) and the node condition ($v_0 \in V^s_{t_0}$, while $v_1, v_2 \notin V^s_{t_0}$). To efficiently determine a node's inclusion within $V^s_{t_0}$, a strategy is applied where each node initially receives a unique ID in ascending sequential order. The decision on a node's inclusion is based on whether its ID exceeds the maximum ID within $V^s_{t_0}$.

Time Complexity. The time complexity of our algorithm hinges on the motif finding algorithm employed. We argue that any motif finding algorithm's time complexity must be at least linear to $\mathcal{O}(\sum_{t=1}^{T} |Motif_t|)$, where $Motif_t$ represents identified motifs at timestamp t. This assertion holds because validating each edge's eligibility as the desired motif takes at least $\mathcal{O}(1)$, assuming these edges are precisely the final motifs in the best-case scenario. Both edge and node fil-

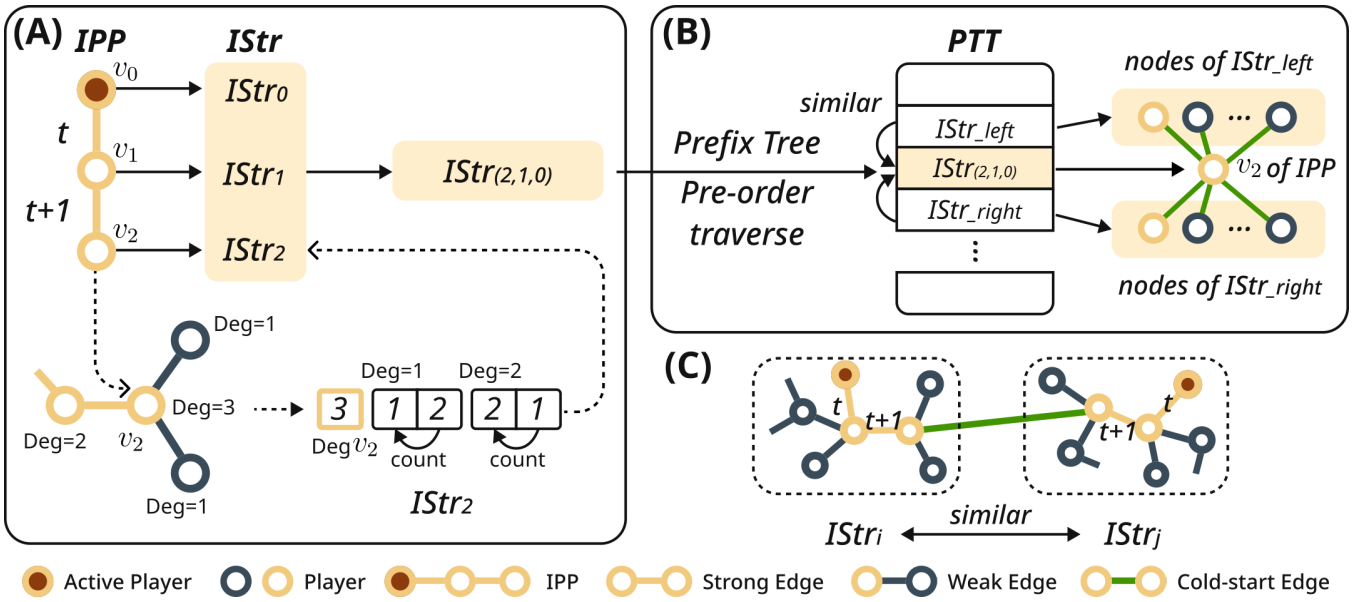

Figure 3: Cold-start solution: (A) IPP Serialization: Serialized strings are generated for each node in an IPP and then concatenate into an *IStr*. (B) Neighbor Retrieval: The *IStr*s of all IPPs are inserted into a prefix tree and positioned by pre-order traversal (PTT), ensuring that similar strings are placed adjacently. Cold-start edges are established between the end node v_2 of an IPP and other nodes belonging to the most similar *IStr*s in the PTT. (C) Demonstration of neighbor retrieval.

tering processes involve iterating through all motifs, resulting in a time complexity of $\mathcal{O}(\sum_{t=1}^{T}|Motif_t|)$. Thus, the time complexity of the filtering process for IPPs is linear to the number of found motifs and must be less than the motif identification process. Our implementation employs *Dotmotif* (Matelsky et al. 2021), an efficient variant of the VF2 algorithm that decouples time complexity from the number of edges. Finally, compared with implementations that recursively examine nodes' neighbors and validate the formation of 2-hop paths, our motif-filtering method excels in handing dense graphs, where the edge count grows exponentially with the node count.

TGN Deployment and Tensorization

To predict IPPs, we implement TGN to fit in our scenario. Our implementation also tensorizes two modules (memory and aggregator) of TGN to improve efficiency.

TGN Architecture. TGN incorporates a specialized storage mechanism termed *memory*, wherein a vector assigned to each node encapsulates the most recent data related to the edges. The training procedure of TGN is demonstrated through a batch comprising $e^{t_0}_{v_0,v_1}, e^{t_1}_{v_1,v_2}$. Initially, node embeddings for $v_0^{t_0}, v_1^{t_0}, v_1^{t_1}, v_2^{t_1}$ are generated by the embedding module, utilizing the temporal graph and the node's *memory*. Subsequently, these embeddings, along with edge information, serve as historical *messages* to update the memory of the nodes. Particularly, information for v_1 at t_0 and t_1 is aggregated, updating its *memory* to t_1. Following this batch, the *memory* stores three vectors for $v_0^{t_0}, v_1^{t_1}, v_2^{t_1}$. For more details, please refer to the paper (Rossi et al. 2020).

Specification. In our scenario, the edge attributes are categorical, with values of 0, 1, 2 representing weak relationships, strong relationships, and the additional connections (introduced in Section) for cold-start nodes, respectively. This notation seamlessly aligns with the requirements of our temporal multi-relational network implemented by TGN. Subsequently, the generated node embeddings are passed through a Multi-layer Perceptron (MLP) decoder, consisting of two Linear-ReLU-Dropout blocks followed by a Linear layer, with an output dimension of 2. The decoder is then trained using a supervised Binary Cross-Entropy (BCE) loss, with the labels identified by IPPs fed into it. Finally, the weighted sum of the supervised BCE loss and the unsupervised TGN loss forms the multi-task loss. During inference, the class with the higher value is considered as the prediction.

Tensorization. In our deployment of TGN, we have identified a key challenge that hampers efficiency. Specifically, concerning message aggregation, TGN explores two alternatives: exclusively utilizing the latest message (LM) and calculating the mean of all messages (MM). An ablation study (Rossi et al. 2020) indicates that while LM slightly outperforms MM in accuracy, it incurs a threefold computational time. Nevertheless, the original implementation is designed for supporting various aggregators. Recognizing this as a potential avenue for efficiency enhancement, we exclusively employ LM as the aggregator and implement a tensorized *memory* and LM aggregator to minimize redundant loops. This tailored implementation results in a significant improvement in efficiency, surpassing the aforementioned threefold acceleration. Moreover, our acceleration on a single operation is orthogonal to the parallel framework designed to accelerate general Temporal Graph Neural Net-

works (TGNNs) (Xia et al. 2024; Sheng et al. 2024; Zhou et al. 2022), which is the primary focus of previous works.

Cold-start Solution

To address the neighbor insufficiency in cold-start nodes, we propose establishing connections among cold-start nodes that share similar IPPs. As shown in Figure 3, this process begins with the serialization of IPPs into fixed-length strings, facilitating subsequent calculation. After serialization, we construct a prefix tree for efficient retrieval of IPPs with similarities. This method adeptly manipulates social relationships within the network data, introducing edges that minimally alter the network's structure. Consequently, our solution to the cold-start problem aligns well with various algorithms designed for temporal graphs. The complete process is outlined in Figure 3.

IPP Serialization. We introduce degree counts for a specific degree k, denoted as:

$$CD(v_i, k, t) = \sum_{nbr_j \in N^t v_i} \mathbb{I}(Deg(nbr_j, t) == k). \quad (2)$$

Here, $Deg(v_i, t)$ denotes the degree of v_i^t, and $\mathbb{I}$ yields 1 when the condition is met. This serialization (Figure 3(A)) method preserves neighborhood information by associating each neighbor's degree with the degree count. To simplify, the serialization retains the node's degree and the three most frequent degree counts. To ensure consistent string length, degrees are truncated to 99, with zeros appended for nodes lacking sufficient neighbors. Subsequently, for three nodes of an IPP $v_0^{t_0}, v_1^{t_1}, v_2^{t_2}$, we generate serialized strings for each involved node, which are then concatenated in reverse order to form $IStr_{v_2,v_1,v_0}^{t_1,t_1,t_0}$. This reverse concatenation is specifically designed for the inference phase, where only historical data are available. The retrieval of IPPs thus focuses on identifying potential neighbors for the latest node v_2 of an IPP, utilizing the serialized IPPs.

Neighbor Retrieval. The Trie, or prefix tree, plays a crucial role in efficiently retrieving IPP strings based on common prefixes. After inserting all IPP strings into the Trie, strings with similarities are positioned adjacently within the sequence generated through the Trie's pre-order traversal, denoted as PTT (construction and traversal details omitted). Subsequently, for a node v_i, we illustrate the retrieval process of an IPP ending with v_i, and its serialization is simplified to $IStr_i$ (illustrated in Figure 3(B) and Algorithm 1). Similar strings are placed adjacently to the position, where $PTT[position] = IStr_i$. These identified strings reflect potential neighbors for the given nodes. A subset of these nodes is randomly selected to augment v_i's neighborhood. Additionally, we filter the identified strings based on similarity to the retrieval string. Specifically, considering each two words as a number within an IPP string (N_I numbers in total), the similarity between any two IPP strings, $IStr_i$ and $IStr_j$, is calculated as:

$$Sim(IStr_i, IStr_j) = \sum_{k=0}^{N_I - 1} \mathbb{I}(IStr_i[k] == IStr_j[k]), \quad (3)$$

Algorithm 1: Neighbor Retrieval (IPPs, PTT, w, h)

$E_c \leftarrow \emptyset$;
for $(v_2, IStr) \in IPPs$ **do**
  $pos \leftarrow \text{Index}(PTT, IStr)$; $Q \leftarrow \emptyset$;
  for $IStr' \in PTT[pos - w, pow + w]$ **do**
    if *Sim*$(IStr, IStr') < h$ **then**
      $V' \leftarrow StringToNode(IStr')$;
      $Q \leftarrow Q \bigcup V'$;
  $Q \leftarrow Sample(Q)$;
  $E_c \leftarrow \{v_2\} \times Q$;
return E_c;

where $IStr[k]$ denotes the k-th number. To boost retrieval efficiency, our implementation utilizes caching, capitalizing on the relatively small size of the PTT compared to the total number of IPPs. An example is demonstrated in Figure 3(C), omitting details.

Time Complexity. If d_{avg} represents the average node degree, generating an $IStr_{v_i}^{t_i}$ requires $\mathcal{O}(d_{avg})$ operations to access the degree of each neighbor. Assuming M IPPs found, the total time complexity for IPP serialization becomes $\mathcal{O}(Md_{avg})$. Neighbor retrieval involves Trie construction, PTT generation, and neighbor retrieval for each cold-start node. Trie construction takes $\mathcal{O}(LM)$, where L denotes the length of serialized string. PTT generation, while worst-case $\mathcal{O}(LM)$, is significantly lower in practical scenarios. The neighbor retrieval exhibits linear time complexity in indexing all strings, assuming a constant number of adjacent positions are considered similar. This indexing requires the length of PTT to populate the cache, with a worst-case complexity of $\mathcal{O}(M)$ where each serialized IPPs is distinct. Similarly, retrieving nodes based on $IStr$ also has a worst-case time complexity of $\mathcal{O}(M)$ to assign all nodes to each $IStr$. In summary, our approach to addressing the cold-start issue has a time complexity of $\mathcal{O}(Md_{avg} + LM)$. In practice, we suggest $\mathcal{O}(Md_{avg})$ due to shared prefixes, reducing the effective length of Trie traversal.

Offline Experiment

The aim of the offline experiment is to validate the predictive accuracy of our approach. We utilize four predictive models detailed in *SPEX* (Li et al. 2021) and five datasets for comparison. Furthermore, we evaluate the efficiency of model training across various scales and computational platforms, conducting experiments in two distinct environments to account for scale variations. The code is opensource [2].

Experimental Setting

Dataset. We utilize two datasets obtained from different platforms. A temporal graph is generated by aggregating interactions over a one-week timeframe, where the ground truth of IPP and addition edges for cold-start nodes are identified. The identified IPP is based on a 2-hop structure un-

[2]https://github.com/laixinn/ICWSM25-Influence-Maximization/

Name	*Twit.*	*NetE.*	*B.C.*	*W.T.*	*S.O.*
nodes	50.4K	1.4M	274	85.9K	0.6M
edges	59.0K	13.4M	809	524.6K	7.5M
weeks	9	8	266	223	86
strong nodes	39.7K	0.2M	218	66.8K	0.5M
strong edges	37.4K	0.5M	201	292.0K	2.8M
ground truth	8.8K	47.9K	77	9.8K	0.1M

Table 1: Overall Statistics ($K = 10^3, M = 10^6$).

der the most relaxed conditions. This is because IPPs with longer hops must be a subset of the 2-hop IPPs, making these 2-hop IPPs representative. Detailed statistics are provided in Table 1, which aggregates data over time.

NetEase (NetE.). This dataset contains interaction data, considered as strong and weak relations, and player profiles obtained from an in-game recommendation system, spanning from January 3 to February 28, 2022. Experiments with this dataset are conducted on a high-performance machine featuring an AMD EPYC 7543 32-Core CPU, 377GB RAM, and an NVIDIA 24G A30 GPU.

Twitter (Twit.) (Lou et al. 2013; Li et al. 2021). This dataset comprises retweet logs from January 1 to March 1, 2010. Strong and weak relational edges are established based on a predefined threshold (0.3 quantile) on edges' accumulated weights. We utilize a BERT-base model[3] to generate tweet embeddings, which are then aggregated to derive user features. Experiments on this dataset are conducted on a system equipped with an Intel(R) Xeon(R) Gold 5218 CPU, 512GB RAM, and an NVIDIA 24G GeForce RTX 3090 GPU.

SNAP Temporal Networks [4]*: BitCoins (B.C.), WikiTalk (W.T.), StackOverflow (S.O.)* (Kumar et al. 2018; Paranjape, Benson, and Leskovec 2017). We leverage publicly available datasets from SNAP, encompassing a diverse range of temporal networks over extended periods. To better capture long-term propagation and ensure the resulted IPP labels, we aggregate each network into 60-day intervals. When no inherent node features are provided, we substitute them with random embeddings.

Baseline. In our evaluation, we consider the following baseline algorithms from two perspectives:

SPEX Competitors. *FuseRec* (Narang et al. 2021), *DiffNet++* (Wu et al. 2020), and their *SPEX* adaptations, without scalable implementation, serve as prediction baselines in Twitter dataset. We adaptively replace items by users and modify supervised loss similar to *SPEX*. The accuracy threshold is set at 0.5. This baseline does not scale effectively to large datasets. Therefore, we intentionally downsample the *S.O.* dataset to evaluate its maximum capable performance, by 50% for FuseRec (&spex) and 40% for DiffNet++ (&spex).

Graph Solutions. *GraphMAE* and *TGN* are representative for the algorithms in static and temporal graphs. Their scalable implementations are utilized in both Twitter and NetEase. dataset. Both approaches employ a 2-class prediction.

For consistency, methods integrating our cold-start solutions or the *SPEX* approach are labeled as "&cold" and "&spex", respectively. To maintain data integrity, datasets are split in a 4:1:1 ratio for training, validation, and testing, preserving temporal sequence integrity. Evaluation metrics including Average Precision (AP), ROC, AUC, and accuracy are aligned with TGN's setting.

Result Analysis

The experimental results, summarized in Table 2, emphasize the highest (**first**) and second highest (second) scores. In most scenarios, *TGN&cold* achieve the best or near-best results, outperforming the *spex* plugin designed for IPP identification. Several significant insights can be found from the results:

***Temporal Information's Impact*: temporal data inclusion markedly improves model performance.** In W.T. and S.O., two *TGN*s consistently outperform in three metrics, averaging $95.46\%, 90.09\%, 97.34\%$. Conversely, moving from the short-term (Twit., NetE.) to long-term datasets (W.T., S.O.) reveals an enhanced performance for *TGN&cold* but a marked decline for two *GraphMAE*s, particularly in AUC (79.15% to 52.82% on average). This contrast underscores the critical role of temporal structure in delivering accurate IPPs for the following empirical study.

***Propagation-insufficient Scenario*: sparse propagation results in insufficient labels and degrades the performance for our cold-start solution.** In the B.C. dataset, 43.5% timestamps include ≤ 1 propagation (i.e., label). Decreasing the accumulation period further increases this proportion and exacerbates label imbalance. These observations illustrate the nature of the rare propagation behavior in the bitcoin scenario. Consequently, *TGN*'s ACC drops to a secondary level, which carries over to the *TGN&cold*. In contrast, W.T. and S.O datasets have 100% and 84.6% of timestamps including more than 10% of propagation, with average propagation proportions of 14.1% and 22.0%. In these datasets, two *TGN*s outperform other baselines in these datasets, demonstrating the strong correlation between propagation and performance. However, in the propagation-insufficient B.C. dataset, our cold-start solution is particularly helpful for *GraphMAE*, with an average improvement of 8.38%. As discussed earlier, the static *GraphMAE* struggles with learning temporal information, and thus the absence of propagation has a neutral or even positive effect on its performance. Overall, *TGN&cold*'s worst performance in B.C. arises from *TGN*'s limitation in propagation-insufficient scenarios, while the cold-start solution itself remains benefit for static GNNs.

***Plugin-perspective Performance*: our cold-start solution (*cold*) outperforms the plugin competitor SPEX (*spex*) regarding overall stability.** These plugins share advantages of requiring no architectural changes to the base models. Integrating *spex* and *cold* delivers a state-of-the-art overall performance improvement of 3.20% and 3.25%. However, *cold* demonstrates a positive effect in 73.3% of

[3]https://github.com/google-research/bert

[4]https://snap.stanford.edu/data/index.html#temporal

	AUC (%)					ACC (%)					AP (%)				
	Twit.	NetE.	B.C.	W.T.	S.O.	Twit.	NetE.	B.C.	W.T.	S.O.	Twit.	NetE.	B.C.	W.T.	S.O.
DiffNet++	64.23	OOM	30.47	52.16	59.84	59.62	OOM	61.49	50.51	58.00	72.81	OOM	31.07	34.06	52.23
DiffNet++&spex	60.68	OOM	57.35	52.16	74.79	53.56	OOM	61.49	80.43	57.90	64.16	OOM	48.21	28.58	65.49
FuseRec	66.09	OOM	**77.46**	57.50	63.40	63.26	OOM	**77.84**	45.57	55.37	69.62	OOM	**80.28**	68.43	76.61
FuseRec&spex	66.30	OOM	77.38	57.46	63.46	62.02	OOM	77.84	45.57	54.94	69.59	OOM	80.12	68.38	76.87
GraphMAE	76.93	81.72	61.35	53.00	50.42	53.52	**79.78**	71.88	72.16	75.67	37.73	76.02	46.48	29.94	24.49
GraphMAE&cold	74.77	83.16	74.44	57.46	50.38	**87.25**	78.33	75.00	72.32	75.68	34.39	75.90	55.41	39.40	24.26
TGN	77.67	81.66	74.53	96.33	91.71	71.38	76.00	51.70	91.61	85.13	**80.72**	80.55	75.96	97.58	95.38
TGN&cold	**78.24**	**83.48**	74.81	**97.30**	**93.62**	73.85	77.20	51.70	**92.99**	**87.19**	80.50	**83.72**	77.05	**98.26**	**96.42**

Table 2: IPP Prediction Performance (OOM: out-of-memory).

cases (22 out of the 30), outperforming *spex*'s 33.33% (8 out of 24 due to OOM). Notably, these results consider the integration to both static (*GraphMAE)* and dynamic (*TGN*) GNN, demonstrating the cold-start solution's broad applicability. Despite *GraphMAE* showing a lower compatibility with *cold* (60% positive effect) compared to *TGN* (93.3%), *cold* still surpasses *spex* when integrating to *GraphMAE*. The observed variation in compatibility is attributed to the absence of temporal information, which affects model's capability in learning cold-start edges.

Computational Efficient

We conduct a comprehensive benchmarking on TGB (Huang et al. 2023) to demonstrate our efficiency improvement, as shown in Table 3. The results indicate a speedup of up to 4× faster than the TGN implementation. The first five datasets are for link prediction, and the subsequent four datasets are for node prediction. The experiment code is available [5]. Additionally, in the Twit. dataset, an ablation study examining the effect of batch size on efficiency enhancement showed that the per-epoch time decreases from 295s to 112s and from 162s to 51s for batch size of 256 and 1024, respectively. This acceleration demonstrates a performance gain increasing from the raise of 1.82 to 2.2 when increasing the batch size fourfold, highlighting our method's potential to scale even further. In the NetE. dataset, due to GPU memory constraints, the most efficient tensorized version of *TGN* (1.54 hours for a batch size of 2048) is 22× faster than its original counterpart (34.27 hours for a batch size of 1024). These observations collectively suggest that our accelerated approach yields potential performance benefits in large-scale scenarios. However, the acceleration provided by our implementation primarily serves as an orthogonal alternative and does not match the hundred-fold speedups offered by specialized parallel architectures (Yu et al. 2024).

Empirical Study

We perform an A/B testing on NetEase Game's team-vs-team game platform to demonstrate the effectiveness of our method in addressing the IM problem. This online experiment ran from January 29 to February 3, 2024. At the beginning of each week, the model was updated with the latest monthly data.

A/B Testing Environment

We start by outlining the fundamental rules of the predefined in-game recommendation system on our platform. Our aim is to target invitations exclusively to online players, requiring daily training of our recommendation system's algorithms to suggest from a pool of real-time online players. We apply additional filtering rules, such as matching preferred game mode with the inviter, to ensure the relevance of recommendations. Subsequently, a portion of online players receive experimental recommendations. We highlight some key settings to ensure test fairness:

Seed Determination. We randomly assign these online players to separate groups, each experiencing a different algorithm to generate an equivalent number of seeds. These algorithms undergo weekly training updates and generate initial predictive scores for player recommendations at the beginning of the week. Nodes with top scores are determined as seeds. The number of seeds is determined by the algorithm with the fewest output predictions.

Seed utilization. We determine our seeds at the beginning to simulate the IM setting. Throughout the week, these seeds receive priority in recommendation, except when offline, reverting to default recommendations in the absence of entire seeds. This strategy of consistent recommendation is also employed for promotion (Zhang et al. 2023; Wang et al. 2021).

Cool-down mechanism. We incorporate a "cool-down" mechanism to prevent recommendation overload: players who receive ten invitations are excluded from recommendations for 24 hours. This helps prevent player annoyance and evenly distributes social interaction among the seeds, avoiding monopolization of attention by the highest-scoring seeds.

Experimental Setting

Baseline. The approaches selected here differ from those in the offline experiments by incorporating a Deep Learning (DL) solution for IM. These SPEX solutions also adhere to our supervised setting for predicting seeds.

IM Competitor with Deep Learning Approach: We use *GE* (Wang et al. 2021) as a validated DL baseline due to

[5]https://github.com/laixinn/TGB

Implementation	Coin	Wiki	Review	Comment	Flight	Trade	Token	Genre	Reddit
Original	472	2	198	1523	472	2	438	71	345
Tensorized	**204**	2	**35**	**326**	**470**	**1**	**260**	**50**	**207**

Table 3: Time consumption (/s per epoch) on TGB.

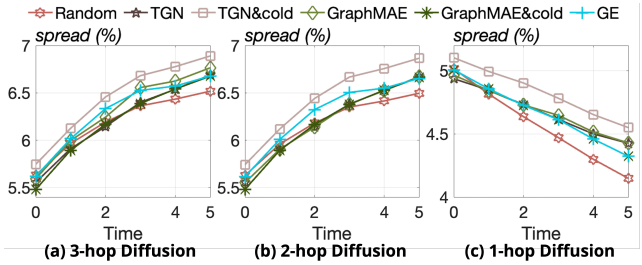

Figure 4: Proportion of network scale over 6 days.

its proven efficacy.

Graph Solutions: In our comparative analysis, we utilize *GraphMAE* and *TGN* to explore static and temporal graph analysis, enriching our insights into various graph-based approaches.

Random: A randomly selected subset of seeds is exposed to default recommendation generated by a LightGBM model (Ke et al. 2017) trained to maximize the click-through rate (CTR) for invitation acceptances. This default recommendation serves as a robust business baseline and undergoes thorough online evaluation.

Metrics. To evaluate the effect of coverage growth, i.e., network scale growth, we track the daily network scale divided by the accumulation of daily active user (DAU). We use *spread* to denote this network scale growth. In this context, the network edges originate exclusively from the recommendation system, and the DAU accurately reflects the number of available players in a test group. Although detailed DAU figures and any data disclosing DAU are kept confidential, it's important to note that both the number of seeds and the scale of the diffused network are relatively small compared to the DAU. This setup ensures that the regular gaming experience of the majority of players remains undisturbed.

Result Analysis

We begin our results analysis with an overview, followed by targeted investigations into cold-start nodes and an ablation study to validate the cold-start solution. In the illustrations, (x, y) coordinates represent positions from the upper-left corner of each figure.

Overall Performance. The diffusion outcomes are detailed in Figure 4. Diffusion beyond 3 hops is excluded since the network scale does not exhibit growth beyond this point. Our method, *TGN&cold*, consistently leads in spreading across both time and hops, achieving a relative improvement of 3.52% in overall statistics. The *random* strategy demonstrates the lowest spread (Time = 4 and 5 in Figure 4), confirming the observation that social recommendation strategies targeting high adoption rates fall short in expanding network scale. A decrease in 1-hop diffusion in Figure 4(a) indicates that the rate of network diffusion is slower than the accumulation of DAU. This is attributed to the cool-down mechanism, which also serves as evidence that recommending our seeds does not introduce bias into the study of the IM problem.

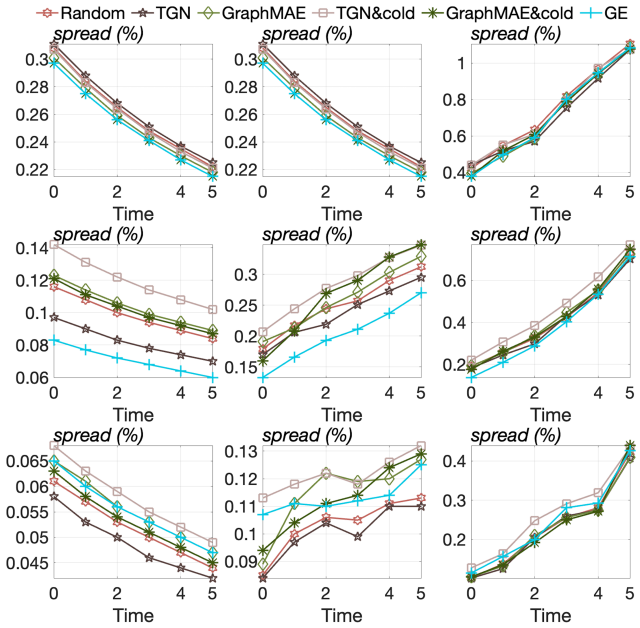

Figure 5: Investigation of degree v.s. hop, where x,y position refers to $\{1, 2, 3\} \times \{1, 2, 6\}$. Each figure presents the proportion of network scale over 6 days.

Cold-Start Treatment. To address the cold-start issue, we evaluate the enhancements for cold-start nodes, as illustrated in Figure 5. The analysis focuses on nodes with fewer than three neighbors at the time of seed prediction (i.e., January 28, 2024). *TGN&cold* either leads or ties in most cases, affirming its effective treatment of cold-start nodes. This is especially evident in cases (2,1), (2,2), (3,1), and (3,2) in Figure 5, where a clear distinction is observed between the default random strategy and *TGN&cold*. The overall statistical improvement here is 14.32%. To explain, we observe that performance rankings remain largely consistent over time and thus the initial performance determines the overall performance. This observation highlights the significance of the players reached initially, which is impacted by the cold-start issue we particularly focus on to improve the IM problem.

Additionally, in Figure 5, *TGN* exhibits lower perfor-

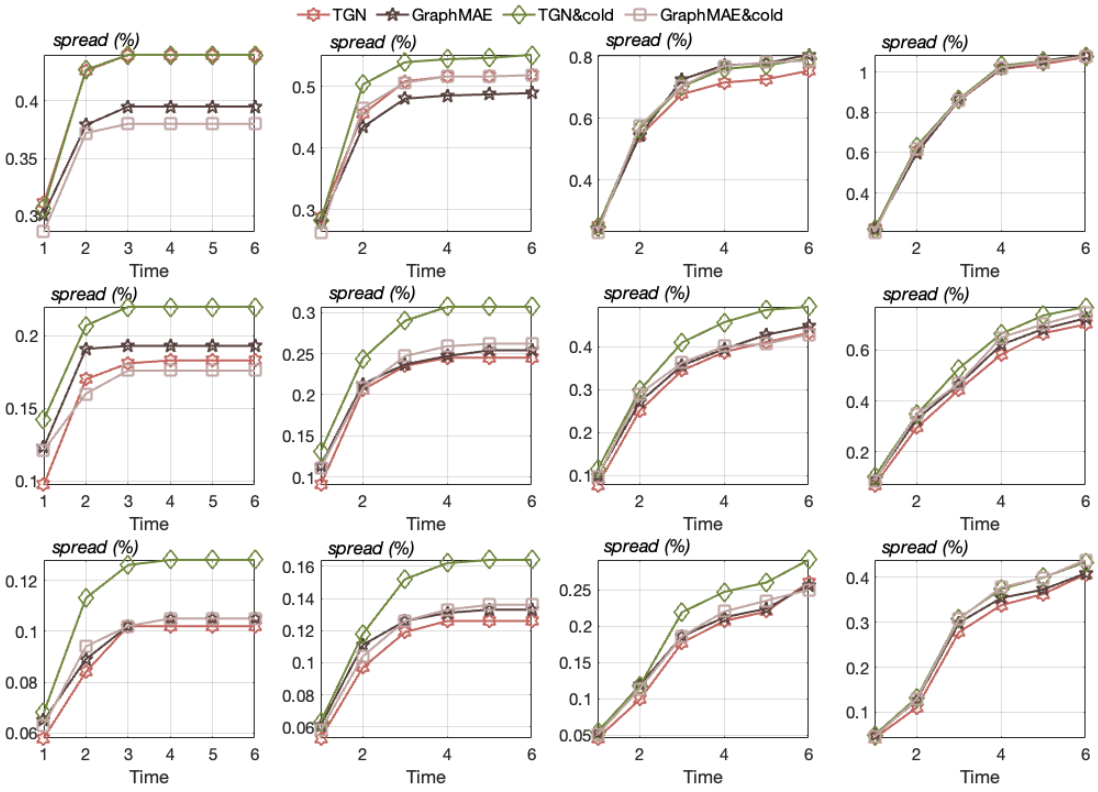

Figure 6: Ablation study of the cold-start solution, where x,y position refers to degrees crossing times, i.e., $\{1, 2, 3\} \times \{1, 2, 4, 6\}$. Each figure presents the proportion of network scale over 6-hop diffusion.

mance without the support of our cold-start solution, particularly evident in cases (2,1), (2,2), (3,1), and (3,2). Although *GE* shows comparable performance in Figure 4, it also struggles to address cold-start challenges, resulting in the lowest performance in cases (2,1) and (2,2). In contrast to its lower performance in Figure 4, the *random* baseline remains comparable throughout Figure 5. We attribute this to its local optimization strategy, which targets the adoption rate without considering long-term spreading. Consequently, its initially comparable performance does not sustain over time, leading to its lowest performance in Figure 4.

Ablation Study Along Hops. An ablation study focusing on diffusion across hops is presented in Figure 6. Overall, *TGN&cold* outperforms *TGN* by 15.59%, whereas *GraphMAE&cold* is marginally surpassed by 0.21%. This finding is consistent with insights from the offline experiment and clearly demonstrates the benefits of our cold-start solution. Additionally, the increase in spread is particularly evident in the first two hops, aligning with our identification of two-hop IPPs. Beyond the second hop, the spread curves become flat and reach saturation, indicating the potential of our supervised framework to scale up IPP hops. During these two hops, *TGN&cold* exhibits significantly enhanced propagation power compared with *TGN*, achieving a higher saturation level. In later days, the impact of our cold-start solution diminishes as the propagation accumulates sufficient neighbors, moving past the initial cold-start period. This suggests that the final performance improvement is primarily driven by the initial propagation power provided by our cold-start solution. This is particularly helpful for nodes that reach propagation saturation quickly. However, it fails to break the propagation ceiling for the cold-start nodes in the long term.

Conclusion and Future Work

In this study, we explore supervised learning techniques to tackle the IM problem in temporal graphs, focusing on mitigating the cold-start issue identified through statistical analysis. We first frame the IM problem as a supervised learning task and propose a method for identifying IPPs within multi-relational temporal graphs. To predict these IPPs, we implement a tensorized TGN model, incorporating an innovative cold-start solution. Our approach is validated through offline evaluations across three key metrics, demonstrating both accuracy in IPP prediction and efficiency based on time consumption analysis. Additionally, online A/B testing with a controlled configuration confirms the practical benefits of our approach for fostering network growth. Ablation studies further highlight the efficacy of our solution in addressing the cold-start issue, offering valuable insights and practical contributions to IM in temporal networks.

Due to our research scope, we exclude diverse recommendation baselines (e.g., adoption rate (Zhang et al. 2023)) and static IM problems (Wang et al. 2024). Recent temporal GNN methods (Poursafaei et al. 2022; Luo and Li 2022) may not scale to our industrial dataset, making TGN a suitable base for advancing temporal networks in scalable IM. Our cold-start solution currently depends on TGN; future work should assess its generalizability across models, explore potential incompatibilities, and enhance adaptability. Nevertheless, future research could address its limited performance in propagation-insufficient domains, such as "who-trusts-whom" networks in Bitcoin trading, and further investigate its behavior in long-hop propagation scenarios.

Ethics Statement

Our empirical study involved conducting online A/B tests. We ensure that our data collection process strictly adhered to privacy and confidentiality standards, and the experimental design is carefully crafted to prevent any negative impacts on players. A potential ethical concern is the abuse of propagation power to influence a large number of players. However, the demonstration of our cool-down mechanism provides a proper guidance to control these unintended propagation effects. Consequently, we believe that our study does not incur any significant ethical issues.

Acknowledgments

This work is supported by grants from the National Natural Science Foundation of China (No. 62372298).

References

Bedi, P.; et al. 2020. Combining trust and reputation as user influence in cross domain group recommender system (CDGRS). *Journal of Intelligent & Fuzzy Systems*, 38(5): 6235–6246.

Castiglione, A.; Cozzolino, G.; Moscato, F.; and Moscato, V. 2020. Cognitive analysis in social networks for viral marketing. *IEEE Transactions on Industrial Informatics*, 17(9): 6162–6169.

Chalyi, S.; Leshchynskyi, V.; and Leshchynska, I. 2019. Method of forming recommendations using temporal constraints in a situation of cyclic cold start of the recommender system. *EUREKA: Physics and Engineering*, (4): 34–40.

Cheng, C.-H.; Kuo, Y.-H.; and Zhou, Z. 2020. Outbreak minimization vs influence maximization: an optimization framework. *BMC Medical Informatics and Decision Making*, 20(1): 1–13.

Erkol, Ş.; Mazzilli, D.; and Radicchi, F. 2020. Influence maximization on temporal networks. *Physical Review E*, 102(4): 042307.

Feng, Y.; Patel, A.; Cautis, B.; and Vahabi, H. 2023. Influence Maximization with Fairness at Scale (Extended Version). *arXiv preprint arXiv:2306.01587*.

FORCE11. 2020. The FAIR Data principles. https://force11.org/info/the-fair-data-principles/.

Gebru, T.; Morgenstern, J.; Vecchione, B.; Vaughan, J. W.; Wallach, H.; Iii, H. D.; and Crawford, K. 2021. Datasheets for datasets. *Communications of the ACM*, 64(12): 86–92.

Granovetter, M. S. 1973. The strength of weak ties. *American journal of sociology*, 78(6): 1360–1380.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Herce-Zelaya, J.; Porcel, C.; Bernabé-Moreno, J.; Tejeda-Lorente, A.; and Herrera-Viedma, E. 2020. New technique to alleviate the cold start problem in recommender systems using information from social media and random decision forests. *Information Sciences*, 536: 156–170.

Huang, S.; Poursafaei, F.; Danovitch, J.; Fey, M.; Hu, W.; Rossi, E.; Leskovec, J.; Bronstein, M.; Rabusseau, G.; and Rabbany, R. 2023. Temporal graph benchmark for machine learning on temporal graphs. *Advances in Neural Information Processing Systems*.

Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; and Liu, T.-Y. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30.

Kumar, S.; Hooi, B.; Makhija, D.; Kumar, M.; Faloutsos, C.; and Subrahmanian, V. 2018. Rev2: Fraudulent user prediction in rating platforms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, 333–341.

Kumar, S.; Mallik, A.; Khetarpal, A.; and Panda, B. 2022. Influence maximization in social networks using graph embedding and graph neural network. *Information Sciences*, 607: 1617–1636.

Kumar, S.; Mallik, A.; and Panda, B. 2023. Influence maximization in social networks using transfer learning via graph-based LSTM. *Expert Systems with Applications*, 212: 118770.

Li, H.; Li, L.; Xv, G.; Lin, C.; Li, K.; and Jiang, B. 2021. Spex: A generic framework for enhancing neural social recommendation. *ACM Transactions on Information Systems (TOIS)*, 40(2): 1–33.

Li, Y.; Gao, H.; Gao, Y.; Guo, J.; and Wu, W. 2023. A survey on influence maximization: From an ml-based combinatorial optimization. *ACM Transactions on Knowledge Discovery from Data*, 17(9): 1–50.

Lou, T.; Tang, J.; Hopcroft, J.; Fang, Z.; and Ding, X. 2013. Learning to predict reciprocity and triadic closure in social networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 7(2): 1–25.

Luo, Y.; and Li, P. 2022. Neighborhood-aware scalable temporal network representation learning. In *Learning on Graphs Conference*, 1–1. PMLR.

Matelsky, J. K.; Reilly, E. P.; Johnson, E. C.; Stiso, J.; Bassett, D. S.; Wester, B. A.; and Gray-Roncal, W. 2021. DotMotif: an open-source tool for connectome subgraph isomorphism search and graph queries. *Scientific Reports*, 11(1).

Meirom, E.; Maron, H.; Mannor, S.; and Chechik, G. 2021. Controlling Graph Dynamics with Reinforcement Learning and Graph Neural Networks. In Meila, M.; and Zhang, T., eds., *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, 7565–7577. PMLR.

Mendonça, M. R. F.; Barreto, A.; and Ziviani, A. 2020. Efficient information diffusion in time-varying graphs through deep reinforcement learning. *World Wide Web*, 25: 2535 – 2560.

Michalski, R.; Jankowski, J.; and Pazura, P. 2020. Entropy-Based Measure for Influence Maximization in Temporal Networks. In *International Conference on Computational Science*, 277–290. Springer.

Narang, K.; Song, Y.; Schwing, A.; and Sundaram, H. 2021. FuseRec: fusing user and item homophily modeling with temporal recommender systems. *Data Mining and Knowledge Discovery*, 35: 837–862.

Ojagh, S.; Malek, M. R.; and Saeedi, S. 2020. A social–aware recommender system based on user's personal smart devices. *ISPRS International Journal of Geo-Information*, 9(9): 519.

Onnela, J.-P.; Saramäki, J.; Hyvönen, J.; Szabó, G.; Lazer, D.; Kaski, K.; Kertész, J.; and Barabási, A.-L. 2007. Structure and tie strengths in mobile communication networks. *Proceedings of the national academy of sciences*, 104(18): 7332–7336.

Panda, D. K.; and Ray, S. 2022. Approaches and algorithms to mitigate cold start problems in recommender systems: a systematic literature review. *Journal of Intelligent Information Systems*, 59(2): 341–366.

Paranjape, A.; Benson, A. R.; and Leskovec, J. 2017. Motifs in temporal networks. In *Proceedings of the tenth ACM international conference on web search and data mining*, 601–610.

Peng, B. 2021. Dynamic Influence Maximization. In *Neural Information Processing Systems*.

Peng, H.; and Liu, C. 2021. Breaking the information cocoon: when do people actively seek conflicting information? *Proceedings of the Association for Information Science and Technology*, 58(1): 801–803.

Poursafaei, F.; Huang, S.; Pelrine, K.; and Rabbany, R. 2022. Towards better evaluation for dynamic link prediction. *Advances in Neural Information Processing Systems*, 35: 32928–32941.

Qiu, J.; Tang, J.; Ma, H.; Dong, Y.; Wang, K.; and Tang, J. 2018. DeepInf: Social Influence Prediction with Deep Learning. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Ribeiro, L. F. R.; Saverese, P. H. P.; and Figueiredo, D. R. 2017. struc2vec: Learning Node Representations from Structural Identity. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Rossi, E.; Chamberlain, B.; Frasca, F.; Eynard, D.; Monti, F.; and Bronstein, M. 2020. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*.

Sheng, G.; Su, J.; Huang, C.; and Wu, C. 2024. Mspipe: Efficient temporal gnn training via staleness-aware pipeline. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2651–2662.

Vega-Oliveros, D. A.; da Fontoura Costa, L.; and Rodrigues, F. A. 2020. Influence maximization by rumor spreading on correlated networks through community identification. *Communications in Nonlinear Science and Numerical Simulation*, 83: 105094.

Wang, L.; Ding, X.; Gu, Y.; and Sun, Y. 2024. Fast and space-efficient parallel algorithms for influence maximization. In *Proceedings of the 2024 ACM Workshop on Highlights of Parallel Computing*, 9–10.

Wang, W.; Yang, H.; Lu, Y.; Zou, Y.; Zhang, X.; Guo, S.; and Lin, L. 2021. Influence maximization in multi-relational social networks. In *Proceedings of the 30th ACM international conference on information & knowledge management*, 4193–4202.

Wang, X.; Peng, Z.; Wang, S.; Yu, P. S.; Fu, W.; Xu, X.; and Hong, X. 2020. CDLFM: cross-domain recommendation for cold-start users via latent feature mapping. *Knowledge and Information Systems*, 62: 1723–1750.

Wei, J.; He, J.; Chen, K.; Zhou, Y.; and Tang, Z. 2017. Collaborative filtering and deep learning based recommendation system for cold start items. *Expert Systems with Applications*, 69: 29–39.

Wu, L.; Li, J.; Sun, P.; Hong, R.; Ge, Y.; and Wang, M. 2020. Diffnet++: A neural influence and interest diffusion network for social recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 34(10): 4753–4766.

Xia, Y.; Zhang, Z.; Yang, D.; Hu, C.; Zhou, X.; Chen, H.; Sang, Q.; and Cheng, D. 2024. Redundancy-free and load-balanced TGNN training with hierarchical pipeline parallelism. *IEEE Transactions on Parallel and Distributed Systems*.

Yanchenko, E.; Murata, T.; and Holme, P. 2023. Influence maximization on temporal networks: a review. *arXiv preprint arXiv:2307.00181*.

Yu, H.; Zhang, Y.; Tan, A.; Lu, C.; Zhao, J.; Liao, X.; Jin, H.; and Liu, H. 2024. RTGA: A Redundancy-free Accelerator for High-Performance Temporal Graph Neural Network Inference. In *Proceedings of the 61st ACM/IEEE Design Automation Conference*, 1–6.

Zhang, S.; Huang, Y.; Sun, J.; Lin, W.; Xiao, X.; and Tang, B. 2023. Capacity Constrained Influence Maximization in Social Networks. *arXiv preprint arXiv:2306.01782*.

Zhang, Z.; and Liu, H. 2015. Social recommendation model combining trust propagation and sequential behaviors. *Applied Intelligence*, 43: 695–706.

Zhou, H.; Zheng, D.; Nisa, I.; Ioannidis, V.; Song, X.; and Karypis, G. 2022. Tgl: A general framework for temporal gnn training on billion-scale graphs. *arXiv preprint arXiv:2203.14883*.

Paper Checklist

1. For most authors...
   (a) Would answering this research question advance science without violating social contracts, such as violating privacy norms, perpetuating unfair profiling, exacerbating the socio-economic divide, or implying disrespect to societies or cultures? Yes, see the Introduction, Conclusion and Future Work.
   (b) Do your main claims in the abstract and introduction accurately reflect the paper's contributions and scope? Yes, see the Abstract and Introduction.
   (c) Do you clarify how the proposed methodological approach is appropriate for the claims made? Yes, see the Introduction, Offline Experiment, Empirical Study, Conclusion and Future Work.
   (d) Do you clarify what are possible artifacts in the data used, given population-specific distributions? Yes, see the Offline Experiment and Empirical Study.
   (e) Did you describe the limitations of your work? Yes, see the Conclusion and Future Work.
   (f) Did you discuss any potential negative societal impacts of your work? Yes, see the Introduction.
   (g) Did you discuss any potential misuse of your work? Yes, see the Introduction.
   (h) Did you describe steps taken to prevent or mitigate potential negative outcomes of the research, such as data and model documentation, data anonymization, responsible release, access control, and the reproducibility of findings? Yes, see the Introduction.
   (i) Have you read the ethics review guidelines and ensured that your paper conforms to them? Yes, see the Ethics Statement.
2. Additionally, if your study involves hypotheses testing...
   (a) Did you clearly state the assumptions underlying all theoretical results? NA
   (b) Have you provided justifications for all theoretical results? NA
   (c) Did you discuss competing hypotheses or theories that might challenge or complement your theoretical results? NA
   (d) Have you considered alternative mechanisms or explanations that might account for the same outcomes observed in your study? Yes, see the Offline Experiment and Empirical Study.
   (e) Did you address potential biases or limitations in your theoretical framework? Yes, see the Offline Experiment and Empirical Study. Our A/B is under rational and fair setting.
   (f) Have you related your theoretical results to the existing literature in social science? Yes, see the Introduction, Motivated Insights and Problem Definition.
   (g) Did you discuss the implications of your theoretical results for policy, practice, or further research in the social science domain? Yes, see the Introduction and Conclusion and Future Work.
3. Additionally, if you are including theoretical proofs...
   (a) Did you state the full set of assumptions of all theoretical results? NA
   (b) Did you include complete proofs of all theoretical results? NA
4. Additionally, if you ran machine learning experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? No, the empirical study is not reproduceable due to the online environment and the confidential industrial data. However, TGB benchmark can be reproduced, see the open-sourced repository https://github.com/laixinn/TGB.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? Yes, see the Offline Experiment and Empirical Study.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? No, because our experiments take same seed with TGN (2020) to ensure the results are fair and reproduced.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? Yes, see the Offline Experiment.
   (e) Do you justify how the proposed evaluation is sufficient and appropriate to the claims made? Yes, see the Offline Experiment and Empirical Study.
   (f) Do you discuss what is "the cost" of misclassification and fault (in)tolerance? Yes, see the Offline Experiment and Empirical Study.
5. Additionally, if you are using existing assets (e.g., code, data, models) or curating/releasing new assets, **without compromising anonymity**...
   (a) If your work uses existing assets, did you cite the creators? Yes, see the Offline Experiment and Empirical Study.
   (b) Did you mention the license of the assets? No, because all the tools, algorithms and data we use are publicly available.
   (c) Did you include any new assets in the supplemental material or as a URL? Yes, see the URLs in footnotes.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? Yes, see the Offline Experiment and Empirical Study. The data is either publicly available or with consent by players.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? Yes, see the Offline Experiment and Empirical Study. The data is either publicly available or undergoes transformation to keep information confidential.

(f) If you are curating or releasing new datasets, did you discuss how you intend to make your datasets FAIR (see FORCE11 (2020))? NA

(g) If you are curating or releasing new datasets, did you create a Datasheet for the Dataset (see Gebru et al. (2021))? NA

6. Additionally, if you used crowdsourcing or conducted research with human subjects, **without compromising anonymity**...

(a) Did you include the full text of instructions given to participants and screenshots? NA

(b) Did you describe any potential participant risks, with mentions of Institutional Review Board (IRB) approvals? NA

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation?

(d) Did you discuss how data is stored, shared, and deidentified? NA