Deep Graph Representation Learning and Optimization for Influence Maximization

Chen Ling [*1] Junji Jiang [*2] Junxiang Wang [3] My Thai [4] Lukas Xue [1] James Song [1] Meikang Qiu [5]
Liang Zhao [1]

Abstract

Influence maximization (IM) is formulated as selecting a set of initial users from a social network to maximize the expected number of influenced users. Researchers have made great progress in designing various *traditional* methods, and their theoretical design and performance gain are close to a limit. In the past few years, learning-based IM methods have emerged to achieve stronger generalization ability to unknown graphs than traditional ones. However, the development of learning-based IM methods is still limited by fundamental obstacles, including 1) the difficulty of effectively solving the objective function; 2) the difficulty of characterizing the diversified underlying diffusion patterns; and 3) the difficulty of adapting the solution under various node-centrality-constrained IM variants. To cope with the above challenges, we design a novel framework DeepIM to generatively characterize the latent representation of seed sets, and we propose to learn the diversified information diffusion pattern in a data-driven and end-to-end manner. Finally, we design a novel objective function to infer optimal seed sets under flexible node-centrality-based budget constraints. Extensive analyses are conducted over both synthetic and real-world datasets to demonstrate the overall performance of DeepIM. The code and data are available at: `https://github.com/triplej0079/DeepIM`.

[*]Equal contribution [1]Emory University, Atlanta, GA [2]Fudan University, Shanghai, CHINA [3]NEC Labs America, Princeton, NJ [4]University of Florida, Gainesville, FL [5]Dakota State University, Madison, SD. Correspondence to: Chen Ling <chen.ling@emory.edu>, Liang Zhao <liang.zhao@emory.edu>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

1. Introduction

As one of the fundamental research problems in network analysis, the objective of Influence Maximization (IM) is to find a set of seed nodes that maximizes the spread of influence in a social network. IM has been extensively studied in recent years due to its large commercial value. For example, consider the case of viral marketing (Chen et al., 2010) for promoting a commercial product, where a company may wish to spread the adoption of a new product from some initially selected users, the selected initial users are expected to spread the information about the product on their respective social networks. This cascading process will be continued, and ultimately, a significant fraction of the users will try the product. Besides viral marketing, IM is also the cornerstone in many other critical applications such as network monitoring (Wang et al., 2017), misinformation containment (Yang et al., 2020), and friend recommendation (Ye et al., 2012).

As a typical type of combinatorial optimization problem, retrieving a (near) optimal seed set to maximize the influence in a network is challenging due to the stochastic nature of information diffusion and the hardness of the problem. Traditional (non-learning-based) methods for IM (Leskovec et al., 2007; Kempe et al., 2003; Tang et al., 2014; 2015; Nguyen et al., 2016; Saito et al., 2012) have made great progress in the last decade, and Li et al. (2019b) have even achieved exact solutions under specific diffusion models. The commonality of traditional methods is the explicit requirement of the information diffusion model as the model input. However, the real-world information diffusion process is complex and cannot be simply modeled by prescribed diffusion models. With the recent development of machine/deep learning, it is natural to consider a learning-based way to characterize the underlying diffusion process.

While great progress has been made in the field, current efforts on learning-based IM solutions are still in the infancy stage due to fundamental obstacles as follows. *1). The difficulty of efficiently optimizing the objective function.* Learning-based IM methods tend to solve the discrete problem in continuous space by mostly leveraging deep network representations (Zhang et al., 2022; Kumar et al., 2022) and deep reinforcement learning (Tian et al., 2020; Li et al., 2022). Even though they could attain a competitive performance with traditional methods, their scalability and execution efficiency are problematic due to *(a)* the need to iteratively update all node embeddings at each action and *(b)* the #P-hardness of computing the influence spread (Lin et al., 2017). *2). The difficulty of automatically identifying and modeling the actual diffusion process.* To maximize the influence spread in a network, the underlying information diffusion pattern is an imperative part as it determines the overall information propagation outcome. However, both traditional and learning-based methods cannot characterize the underlying diffusion process without heuristics. To work around this, both traditional and current learning-based methods have been leveraging pre-defined diffusion models (e.g., Linear Threshold (LT) and Independent Cascade (IC)) as the input to solve the combinatorial optimization problem. Although they could work well only for the process following their heuristics, the real-world network process is way more complex than the heuristics and largely unknown. *3). the difficulty of adapting solutions to various node-centrality-constrained IM problems.* There are a lot of variants of IM that relate to node centrality, e.g., the constraint on the number of seed nodes, the constraint on the total degree of seed nodes, etc. Current learning-based IM solutions do not have a well-defined paradigm for solving different node-centrality-constrained IM problems, which poses another challenge to their solution adaptivity.

To address the above challenges, we propose a novel framework - DeepIM, to solve the IM problem by developing a novel strategy that embeds the initial discrete optimization domain into a larger continuous space. Remarkably, we propose to learn the latent representation of seed sets by retaining their expressiveness and directly optimizing in the continuous space to reduce the problem's hardness. We further design a learning-based diffusion model to characterize the underlying diffusion dynamics in an end-to-end manner. Moreover, we develop a generic seed set inference framework to directly optimize and generate set embeddings under a uniform budget constraint. Finally, we summarize our contributions as follows:

- **Problem.** We formulate the learning-based IM problem as embedding the initial discrete optimization domain into continuous space for easing the optimization and identify its unique challenges arising from real applications.
- **Framework.** We propose modeling the representation of the seed set in a latent space, and the representation is jointly trained with the model that learns the underlying graph diffusion process in an end-to-end manner.
- **Adaptivity.** We propose a novel constrained optimization objective function to infer the optimal seed set by leveraging deep graph embeddings, which can be applied under arbitrary node-centrality-related constraints.
- **Evaluation.** We conduct extensive experiments over four real-world datasets to demonstrate the performance of the proposed method. Compared with other state-of-the-art in various application scenarios, DeepIM achieves the best results in finding a seed set to maximize the influence.

2. Related Work

2.1. Learning-based Influence Maximization

Influence Maximization (IM) was first formulated as a combinatorial optimization problem by Kempe et al. (2003), which has inspired extensive research and applications in the next decade. Most of the traditional (i.e., non-learning-based) IM methods can be categorized as simulation-based, proxy-based, and heuristic-based. Traditional methods have achieved near or exact solutions under specific diffusion models with efficiency. Note that Du et al. (2014); Vaswani et al. (2017) have alluded to the possibility of learning the influence from the cascade data; however, they still assumed a prescribed model guides the diffusion pattern, specifically the Coverage function. We refer readers to recent surveys (Li et al., 2018; Banerjee et al., 2020) for more detailed reviews of traditional methods.

Learning-based methods use deep learning to address the drawbacks of traditional IM methods, namely the lack of generalization ability. Pioneered works (Lin et al., 2015; Ali et al., 2018) first combined reinforcement learning with IM, and they triggered extensive works that leveraged deep reinforcement learning to solve the IM problem. Existing state-of-the-art solutions (Li et al., 2019a; Tian et al., 2020; Manchanda et al., 2020; Li et al., 2022; Chen et al., 2022) follow a similar paradigm: learning latent embeddings of nodes or networks, and taking the current node embedding as the state of the agent in order to choose the next seed node as action, where the reward is its marginal influence gain. Other than reinforcement learning-based IM methods, there also exist methods (Kumar et al., 2022; Kamarthi et al., 2019; Panagopoulos et al., 2020) that solely leverage graph neural networks to encode social influence into node embedding and guide the node selection process. The crux of current learning-based IM methods is also obvious, the model complexity and adaptivity of learning-based IM methods are still not comparable to traditional methods. Particularly, current ML-based algorithms can neither handle the diversified diffusion patterns nor guarantee the quality of the solution as well as the model scalability.

2.2. Graph Neural Network

Graph Neural Networks (GNNs) (Wu et al., 2020) are a class of deep learning methods designed to perform inference on data described by graphs. The general paradigm of GNNs alternates between node feature transformation and neighbor nodes' information aggregation. For a K-layer GNN, a node aggregates information within K-hop neighbors. Specifically, the k-th layer transformation is:

$$a^k = \mathcal{A}^k(h^{k-1}; \theta^k), h^k = \mathcal{C}^k(a^k; \theta^k), \forall 1 \leq k \leq K. \quad (1)$$

where a^k is an aggregated feature, and h^k is the k-th layer node feature. The flexibility of aggregation function $\mathcal{A}(\cdot)$ and combine function $\mathcal{C}(\cdot)$ functions induces different GNN models (Veličković et al., 2017; Kipf & Welling, 2016; Xu et al., 2018; Wang et al., 2022b). The high-level representations of nodes or graphs are utilized for different tasks. GNNs have been applied in various learning tasks such as information diffusion estimation (Chamberlain et al., 2021; Xia et al., 2021; Ko et al., 2020), graph source localization (Wang et al., 2022a; Ling et al., 2022b), deep graph generation (Ling et al., 2021; 2023a;b), and graph analogical reasoning (Ling et al., 2022a). In this work, we leverage GNN to characterize the underlying diffusion pattern and construct an end-to-end model for estimating the influence.

3. Problem Formulation

Given a graph $G = \{V, E\}$, the problem of IM aims to maximize the number of influenced nodes in G by selecting an optimal seed node set $\mathbf{x} \subseteq V$. Particularly, the evaluation of IM relies on an influence diffusion model parametrized by θ: $\mathbf{y} = M(\mathbf{x}, G; \theta)$, where θ can be the set of infection probability on each node if $M(\cdot)$ is an independent cascade model or the set of parameters in the aggregation/combine functions if $M(\cdot)$ is GNN-based. We denote $\mathbf{x} \in \{0, 1\}^{|V|}$ as the vector representation of the source node set, where the i-th element $x_i = 1, x_i \in \mathbf{x}$ if $v_i \in \mathbf{x}$ and $x_i = 0$ otherwise. The output $y \in \mathbb{R}_+$ measures the total number of infected nodes (Li et al., 2018). Based on the formalization of the influence spread, the IM problem is defined as follows:

Definition 1 (Influence Maximization). The generic *IM* problem requires selecting a set of k users from V as the seed set to maximize the influence spread:

$$\tilde{\mathbf{x}} = \arg\max_{|\mathbf{x}| \leq k} M(\mathbf{x}, G; \theta), \quad (2)$$

where $\tilde{\mathbf{x}}$ is the optimal seed node set that can produce a maximal influence spread in G.

Intuitively, selecting $\tilde{\mathbf{x}}$ heavily depends on the underlying diffusion process. We have witnessed lots of works that develop algorithms with GNNs and reinforcement learning to tackle the problem. However, the expressiveness and generalization capability of existing learning-based IM frameworks is still limited due to the following challenges.

Challenges. Firstly, most existing learning-based IM frameworks calculate the latent node embedding for selecting high-influential ones. However, their objective functions require iteratively updating the latent embeddings for each node at each action/optimization step no matter whether they are included in the current $\mathbf{x}$. This poses a severe scalability problem if we are dealing with million-scale networks. Secondly, even though we leverage deep node/network embedding and various reward functions to guide the node selection process, existing frameworks are still tailored for specific diffusion models (e.g., they model $M(\cdot)$ as explicit IC and LT model). However, these simple diffusion models cannot meet the needs of real applications. Moreover, to ease the computational overhead of the #P-hard influence estimation, learning-based IM methods rely on techniques from traditional methods, such as proxy-based and sampling-based estimation way, which makes scalability and generalization even worse. Lastly, there are plenty of node-centrality-constrained IM variants. For example, other than regulating the budget of the seed nodes, we may also need to regulate the total cost of selecting seed nodes. Learning-based IM solutions have different objective functions designed according to various application scenarios, and they do not have a well-defined scheme for all of the node-centrality-related constraints.

4. DeepIM

In this section, we propose the DeepIM framework to ease the computational overhead of the learning-based IM methods and automatically identify the underlying diffusion patterns. The framework can be divided into two phases: the *learning phase* is leveraged to characterize the probability of the observed seed set and model the underlying information propagation distribution, and the *inference phase* is employed to optimize the selection of seeds in continuous space to maximize the influence spread.

4.1. Learning Representation of Seed Set

To build an effective and efficient objective function, we propose to characterize the probability of the seed node set $p(\mathbf{x})$ over $\mathbf{x}$ given the graph G since learning $p(\mathbf{x})$ can help depict the seed set's underlying nature. However, learning such probability is not a trivial task because different nodes are inter-connected within each seed set and highly correlated based on the topology of G. These connections make the relationship between nodes very complex and hard to decipher than other similar combinatorial problems.

Learning Probability over Seed Nodes. Instead of directly modeling the highly-intractable probability $p(\mathbf{x})$, we introduce an unobserved latent variable z to represent $\mathbf{x}$ and define a conditional distribution $p(\mathbf{x}|z)$ to quantify the likelihood. These latent variables have much lower dimensions than the observed sub-optimal seed sets, which can yield a compressed representation. Particularly, we marginalize over the latent variables to obtain $p(\mathbf{x}) =$

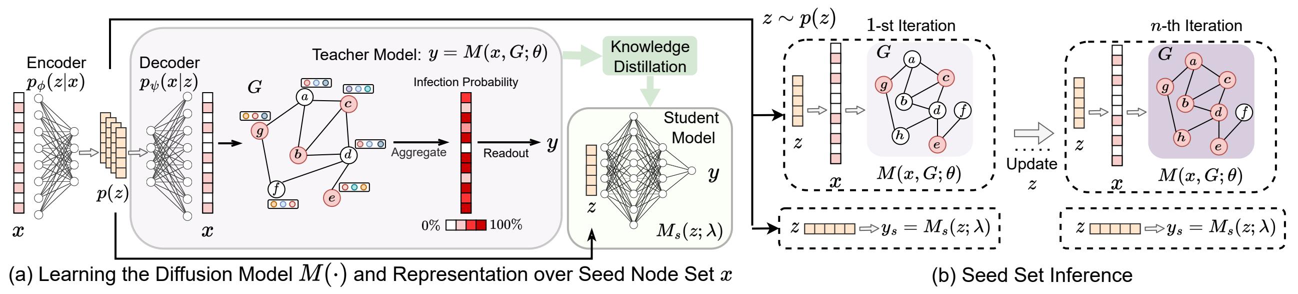


Figure 1. DeepIM consists of two parts. a) we leverage the autoencoder to learn and compress the latent distribution of seed node sets into lower dimension $p(z)$. The lower dimensional $p(z)$ is then leveraged to learn an end-to-end and monotonic diffusion model $M(\mathbf{x}, G; \theta)$ for accurately predicting the spread. In addition, we employ a knowledge distillation module to train a lightweight student model to retain efficiency in predicting the influence spread. b) the seed set inference scheme iteratively optimizes the proposed objective function by updating the latent variable z, initially sampled from the learned $p(z)$, to maximize the influence spread.

$\int p(\mathbf{x}, z)\, dz = \int p(\mathbf{x}|z)p(z)\, dz$. The posterior likelihood $p(z|\mathbf{x}) = p(\mathbf{x}|z) \cdot p(z)/p(\mathbf{x})$ allows us to infer z given the observed seed sets $\mathbf{x}$. In this work, we adopt autoencoder to generatively infer the posterior, where both encoder f_ϕ (parameterized by ϕ) and decoder f_ψ (parameterized by ψ) are used to characterize the likelihood of both posterior and conditional distribution, respectively. The objective of the autoencoder is to maximize the joint likelihood:

$$\max_{\phi,\psi}\ \mathbb{E}\big[p_\psi(\mathbf{x}|z) \cdot p_\phi(z|\mathbf{x})\big]. \qquad (3)$$

Learning the End-to-end Diffusion Model. Once we have learned the latent distribution of seed nodes $p(\mathbf{x})$, the next step is to update the seed node set $\mathbf{x}$ in order to increase the marginal gain of the influence spread. Current learning-based IM solutions still assume the computation of the influence spread (i.e., $M(\mathbf{x}, G; \theta)$) relies on prescribed mathematical models. However, real-world information diffusion is complicated, and it is not easy to determine the most suitable diffusion model in practice. A chosen diffusion model may be misspecified compared to real-world data and lead to large model bias. In addition, the diffusion network structure can also be hidden from us, so we need to learn not only the parameters in the diffusion model but also the diffusion network structure (Du et al., 2014).

In this work, we design a GNN-based diffusion model $M(\cdot)$ for accurate modeling of the relationship between $\mathbf{x}$ and y with considering the overall graph topology. The output of a GNN-based diffusion function $M(\cdot)$ is composed of two functions $M = g_r \circ g_u(\mathbf{x}, G; \theta)$: 1) $\tau = g_u(\mathbf{x}, G; \theta)$, where $g_u(\cdot)$ is a GNN-based aggregation function and $\tau \in [0, 1]^{|V|}$ is an intermediate output after aggregating multi-hop neighborhood information. τ denotes the *infection probability* of each node; and 2) $y = g_r(\tau; \xi), y \in \mathbb{R}_+$ denotes the final information spread, where $g_r(\cdot)$ is a normalization function (e.g., l-1 norm) and ξ is the threshold to transform the probability into discrete value. The GNN-based $M(\cdot)$ is visualized in Fig. 1 (a).

Definition 2 (**Score Monotonicity and Infection Monotonicity**). Given a GNN-based diffusion model $M(\cdot) : 2^{|V|} \rightarrow \mathbb{R}_+$ and any two subsets $S, T \subseteq V$, $M(\cdot)$ is score monotonic if $x_S \preceq x_T$ (i.e. $S \subseteq T$) implies $M(\mathbf{x}_S, G; \theta) \leq M(\mathbf{x}_T, G; \theta)$, where $\mathbf{x}_S, \mathbf{x}_T \in \{0, 1\}^{|V|}$ are vector representations of seed sets S and T, respectively. $M(\cdot)$ is infection monotonic if $x_S \preceq x_T$ (i.e. $S \subseteq T$) implies $\tau_S \preceq \tau_T$, where $\tau_S, \tau_T \in [0, 1]^{|V|}$ denote the infection probability of seed sets S and T, respectively.

Monotonicity is a natural property for us in modeling the overall diffusion network structure. A monotonic diffusion model indicates the spread of influence would continue to increase. Intuitively, if we select a larger community $\mathbf{x}'$ as the seed set, the larger $\mathbf{x}'$ would intrinsically infect no less nodes in the whole network than a smaller seed set $\mathbf{x}$ if $\mathbf{x} \preceq \mathbf{x}'$. Ensuring the property of both monotonicities allows us to better characterize the underlying diffusion network structure and mimic the real-world diffusion pattern (Dolhansky & Bilmes, 2016). Hence, we add constraints to make the GNN-based diffusion model $M(\mathbf{x}, G; \theta)$ monotonic during the influence spread estimation.

Theorem 1 (**Monotonicity of GNN Models**)**.** For any GNN-based $M(\mathbf{x}, G; \theta) = g_r \circ g_u(\mathbf{x}, G; \theta)$, where $g_u(\mathbf{x}, G; \theta)$ is formulated by Eq. (1), M is score and infection monotonic if $\mathcal{A}^k$ and $C^k, k \in [1, K]$, are non-decreasing in Eq. (1), and g_r is also non-decreasing.

We further illustrate that the well-known Graph ATtention network (GAT) can be score and infection monotonic under the constraint we claimed in Theorem 1. Note that we follow the standard network structures of GAT as introduced in the original paper.

Corollary 2 (**Montonicity of GAT**)**.** *M is score and infection monotonic when g_u is GAT if $\theta^k \geq 0$ in Eq. (1) and g_r is also non-decreasing.*

Due to the space limitation, the proof of Theorem 1 and Corollary 2 are provided in Appendix A. According to Theorem 1 and Corollary 2, the GNN-based $M(\mathbf{x}, G; \theta)$ has the theoretical guarantee to retain monotonicity, and the objective of learning the GNN-based $M(\mathbf{x}, G; \theta)$ is given as maximizing the following probability with a constraint:

$$\max_\theta \mathbb{E}\big[p_\theta(y|\mathbf{x}, G)\big], \quad \text{s.t. } \theta \geq 0. \tag{4}$$

Knowledge Distillation for Diffusion Estimation Efficiency. We have learnt the deep representation of seed nodes and an end-to-end diffusion model with a monotonicity guarantee. However, we empirically find the calculation of influence spread $M(\mathbf{x}, G; \theta)$ involves three steps: 1) decoding a node vector $\mathbf{x}$ from the learned posterior $p(\mathbf{x}|z)$; and 2) executing the GNN-based diffusion model $M(\mathbf{x}, G; \theta)$ under the graph G; and 3) normalizing the probabilistic output τ from $M(\mathbf{x}, G; \theta)$ to actual influence spread y. Even though the prediction results are accurate, the computational overhead is still a burden when dealing with million-scale networks. Inspired by recent research on knowledge distillation, we propose to leverage a small yet powerful student model supervised by $M(\mathbf{x}, G; \theta)$ to attain efficiency. Specifically, the student model $M_s(z; \lambda)$ is a lightweight neural network parametrized by λ that directly takes the latent variable z sampled from the learned $p(z)$ as input. $M_s(z; \lambda)$ directly returns the estimated influence spread y_s as output. The distillation loss between the $y = M(\mathbf{x}, G; \theta)$ (teacher model) and $y_s = M_s(z; \lambda)$ can be as simple as $\|y - y_s\|_2^2$.

End-to-end Learning Objective. Finally, in order to bridge the representation learning and the learning of the diffusion model, we propose a unified objective function in an end-to-end manner by putting together Eq. (3) and (4) as:

$$\begin{aligned}\mathcal{L}_{\text{train}} = \max_{\theta,\lambda,\psi,\phi}\mathbb{E}\big[p_\theta(y|\mathbf{x}, G) \cdot p_\lambda(y_s|z) \\ \cdot p_\psi(\mathbf{x}|z) \cdot p_\phi(z|\mathbf{x})\big], \text{ s.t. } \theta \geq 0.\end{aligned} \tag{5}$$

However, optimizing the expectation of joint probabilities could be computationally difficult. We instead derive the negative $\log$ term of Eq. (5) and derive its lower bound as the final learning objective according to Jensen's inequality:

$$\begin{aligned}\mathcal{L}_{\text{train}} &= \min_{\theta,\lambda,\psi,\phi} - \log\Big[\mathbb{E}\big[p_\theta(y|\mathbf{x}, G) \cdot p_\lambda(y_s|z) \\ &\quad \cdot p_\psi(\mathbf{x}|z) \cdot p_\phi(z|\mathbf{x})\big]\Big], \text{ s.t. } \theta \geq 0. \\ &\geq \min_{\theta,\lambda,\psi,\phi} \mathbb{E}\Big[-\log\big[p_\theta(y|\mathbf{x}, G) \cdot p_\lambda(y_s|z) \\ &\quad \cdot p_\psi(\mathbf{x}|z) \cdot (p_\phi(z|\mathbf{x})\big]\Big], \text{s.t. } \theta \geq 0. \\ &= \min_{\theta,\lambda,\psi,\phi} \mathbb{E}\Big[-\log\big[p_\theta(y|\mathbf{x}, G)\big] - \log\big[p_\lambda(y_s|z)\big] \\ &\quad - \log\big[p_\psi(\mathbf{x}|z) \cdot (p_\phi(z|\mathbf{x})\big]\Big], \text{s.t. } \theta \geq 0.\end{aligned} \tag{6}$$

The overall objective consists of minimizing the empirical error $-\log[p_\theta(y|\mathbf{x}, G)]$ of the prediction of y with the reconstructed $\mathbf{x}$ as input and minimizing the reconstruction error. In addition, we minimize the distillation loss $-\log[p_\lambda(y_s|z)]$ to train the student model along with the overall training process. The overall framework for the training of end-to-end diffusion models and the autoencoder for learning the seed set distribution is visualized in Fig. 1 (a).

4.2. Seed Node Set Inference

To infer the high-influential seed node set in the testing domain, we leverage the latent distribution $p(\mathbf{x})$ of the seed node set and the end-to-end diffusion model $M(\cdot)$ jointly from Eq. (6). Firstly, if the autoencoder is well trained and can retain both *continuity* (i.e., two close points in the latent space should not give two completely different contents once decoded) and *completeness* (i.e., for a chosen distribution, a point sampled from the latent space should give "meaningful" content once decoded), the autoencoder in Eq. (3) can generate contents by exploiting the latent feature space $p(z)$ learned from all the examples it was trained from, i.e., $p(\mathbf{x})$. Therefore, we propose to alternatively search the optimal seed node set $\tilde{x}$ in the lower-dimensional and less-noisy latent space $p(z)$. The following corollary demonstrates it is equivalent to estimating the influence spread with the latent variable z rather than high-dimensional $\mathbf{x}$ if the autoencoder retains both continuity and completeness.

Corollary 3 (Influence Estimation Consistency). For any $M(f_\psi(z^{(i)}), G; \theta) > M(f_\psi(z^{(j)}), G; \theta)$, we have $M(x^{(i)}, G; \theta) > M(x^{(j)}, G; \theta)$.

The proof of Corollary 3 can be found in Appendix. According to the corollary, we could find the optimal seed set that can generate the maximal influence by optimizing z in the following joint probability: $\max_z \mathbb{E}\big[p_\theta(y|\mathbf{x}, G) \cdot p_\psi(\mathbf{x}|z)\big]$.

Adaptation to Different IM Variants with Node Centrality Constraints. Since the introduction of IM in (Kempe et al., 2003), IM was studied under various budget constrained settings on nodes in recent years. To enhance the adaptivity of DeepIM, we design a unified constraint that allows inferring seed sets under various budgets on individual nodes. Specifically, the objective $\mathcal{L}_{\text{pred}}$ is given as:

$$\begin{aligned}\mathcal{L}_{\text{pred}} = \max_z \mathbb{E}\big[p_\theta(y|\mathbf{x}, G) \cdot p_\psi(\mathbf{x}|z)\big], \\ \text{s.t. } \sum\nolimits_{i=0}^{|V|} \mathcal{F}(v_i, G) \cdot x_i \leq k,\end{aligned} \tag{7}$$

where $\sum_{i=0}^{|V|} \mathcal{F}(v_i, G) \cdot x_i$ is a generalized budget constraint applied on individual nodes, and k is the actual budget. For the vanilla IM problem that only requires selecting a given number of seed nodes, $\sum_{i=0}^{|V|} \mathcal{F}(v_i, G) \cdot x_i$ can be derived as $\|x \cdot \mathbf{1}\|_1$, where the $\mathbf{1} \in \{1\}^{N\times 1}$ is an all-one vector indicating the price of selecting each node are the same. In addition, for node degree constrained IM problems (Leskovec et al., 2007; Nguyen et al., 2017), $\sum_{i=0}^{|V|} \mathcal{F}(v_i, G) \cdot x_i$ can be derived as $\|x \cdot A\|_1$, where $A \in \{0, 1\}^{N\times N}$ is the adjacency matrix of the network G, and $\|\mathbf{x} \cdot A^i\|_1 \leq k$ represents the $l1$-norm of the total seed node degree is bounded by

Algorithm 1 DeepIM Prediction Framework

```
Input: $\mathcal{L}_{\text{pred}}$; $f_\psi(\cdot)$; $\Phi(\cdot)$; number of training instances $N$;
    the number of iteration $\eta$; learning rate $\alpha$.
1: $z = 1/N \sum_{i=0}^{N} f_\psi(\mathbf{x})$ {$\mathbf{x}$ sampled from training set.}
2: for $i = 0, ..., \eta$ do
3:     $\mathbf{x} \leftarrow f_\psi(z)$ {seed set $\mathbf{x}$.}
4:     $\mathbf{x} \leftarrow \Phi(\mathbf{x})$ {Regularize $\mathbf{x}$ into valid regions.}
5:     $z \leftarrow z - \alpha \cdot \nabla \mathcal{L}_{\text{pred}}(\mathbf{x}, z)$
6: end for
7: $\tilde{\mathbf{x}} \leftarrow \Phi(f_\psi(z))$
```

a budget k. The budget constraint $\mathcal{F}(v_i, G) \cdot x_i \leq k$ can also be easily designed, combined, and adapted to solve the IM variants with even non-uniform prices on each node. With the proposed flexible constraint, the challenge of IM methods' adaptability can be partially addressed.

Implementation Details of the Seed Set Inference. We visualize our inference procedure in Fig. 1 (b). Specifically, the inference framework first samples a latent variable z from the learned latent distribution $p(z)$. The latent variable z is iteratively optimized according to the inference objective function Eq. (7) to attain a larger marginal gain (influence spread). Note that the learning-based diffusion model $p_\theta(y|\mathbf{x}, G)$ can be switched between the student diffusion model $M_s(z; \lambda)$ and the GNN-based diffusion model $M(\mathbf{x}, G; \theta)$ to achieve either efficiency or efficacy. In addition, the constrained objective function Eq. (7) cannot be computed directly so that we provide a practical version of the inference objective function: since the diffused observation y fits the Gaussian distribution and the seed set $\mathbf{x}$ fits the Bernoulli distribution, we can simplify Eq. (7) as:

$$\mathcal{L}_{\text{pred}} = \min_z \Big[-\log \big[\prod\nolimits_{i=0}^{|V|} f_\psi(z_i)^{x_i} (1 - f_\psi(z_i)^{1-x_i} \big] + \big\|\tilde{y} - y\big\|_2^2 \Big] \text{ s.t. } \sum\nolimits_{i=0}^{|V|} \mathcal{F}(v_i, G) \cdot x_i \leq k, \tag{8}$$

where the $\tilde{y}$ is given as the optimal influence spread (i.e., $\tilde{y} = |V|$), and the full derivation of the above equation is provided in Appendix. Furthermore, we utilize the Projected Gradient Descent and propose a regularization function $\Phi(\mathbf{x})$ to keep the predicted seed set $\mathbf{x}$ in a valid region in terms of different constraints. For example, $\Phi(\mathbf{x})$ can be defined as selecting k nodes with the highest probabilities when the price of selecting each node is equal in Eq. (7). $\Phi(\mathbf{x})$ can also be defined as cost-efficiently selecting the top-k nodes from $\mathbf{x}/c(\mathbf{x})$, where $c(\mathbf{x})$ denotes the budget on one node (e.g., node degree). Finally, The optimization procedure is summarized in Algorithm 1. Specifically, we firstly sample an initial latent variable z on Line 1. From Line 2 - 6, we iteratively solve the optimization problem proposed in Eq. (7) via gradient descent optimizer (e.g., Adam) while regularizing the predicted seed set in a valid region with $\Phi(\cdot)$. Fig. 1 (b) illustrates the overall process of the inference objective learning. We provide the derivation details of both Eq. (6) and (8) in Appendix A.

	Digg	Weibo	Power Grid	Network Science	Cora-ML	Jazz	Synthetic
Nodes	279,613	2,251,166	4,941	1,565	2,810	198	50,000
Edges	1,170,689	225,877,808	6,594	13,532	7,981	2,742	250,000

Table 1. The Overview of Dataset

5. Experiment

In this section, we compare the performance of our proposed DeepIM framework across six real networks in maximizing the influence under various settings, following a case study to qualitatively demonstrate the performance of DeepIM. Due to the space limit, more details of the experiment setup, hyperparameter settings, dataset description, and comparison methods can be found in Appendix.

5.1. Experiment Setup

Our primary purpose is to evaluate the expected influence spread as defined in Eq. (2) under various IM application scenarios. Since DeepIM can be easily adapted to different diffusion patterns, we choose two representative models that are commonly used in the IM problem, i.e., LT and IC model. In addition, we also evaluate the IM problem under the susceptible-infectious-susceptible (SIS) epidemic model (Kermack & McKendrick, 1927), where the major difference is activated nodes can be de-activated in SIS. Due to the space limit, the experiment of the non-progressive diffusion model can be found in Appendix.

Data. The proposed DeepIM is compared with other approaches over six real-world datasets, including Cora-ML, Network Science, Power Grid, Jazz, Digg, and Weibo. We also adopt a synthetic dataset that is a random graph with $50,000$ nodes generated by Erdos-Renyi algorithm (Erdős et al., 1960). The statistics of the data are shown in Table 1. We randomly sample seed node set $\mathbf{x}$, and the seed size is proportional to $|V|$ of each network. We then use IC, LT, and SIS models to compute the final influence spread y. The $\{(\mathbf{x}, y)\}$ set then serves as the training set of our algorithm.

5.2. Comparison Method.

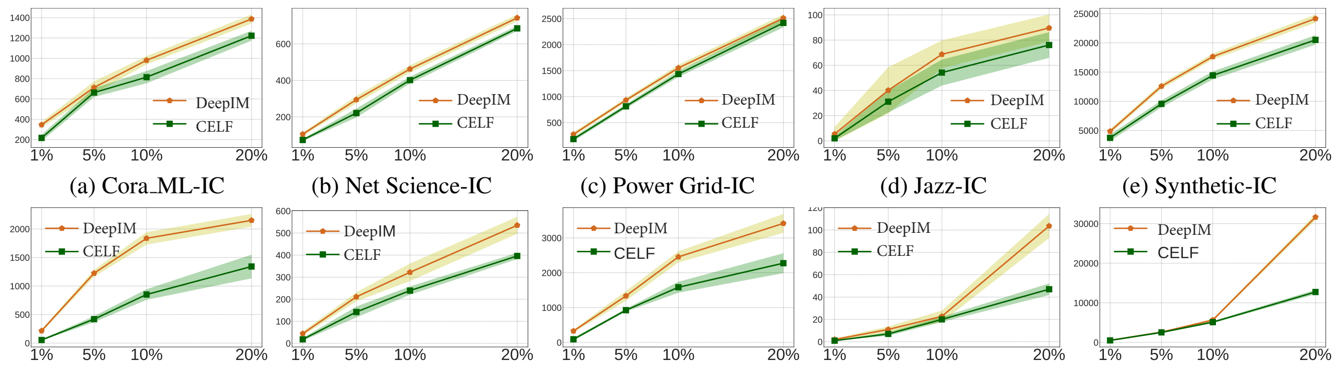
In addition to comparing our model's performance between the GNN-based diffusion model $M(x, G; \theta)$ (denoted as DeepIM) and the student diffusion model $M_s(z; \lambda)$ (denoted as DeepIM$_s$), we also adopt four sets of comparison methods, all of which are outlined as follows. *Traditional IM*: 1) IMM (Tang et al., 2015), 2) OPIM-C (Tang et al., 2018), and 3) SubSIM (Guo et al., 2020). *Learning-based IM*: 1) IMINFECTOR (Panagopoulos et al., 2020), 2) PIANO (Li et al., 2022), and 3) ToupleGDD (Chen et al., 2022). *Online IM*: OIM (Lei et al., 2015). *Budget-constraint IM*:

	Cora-ML				Network Science				Power Grid				Jazz				Synthetic				Digg				Weibo			
Methods	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%
IMM	8.1	26.2	37.3	50.2	5.2	16.8	27.0	45.7	4.3	17.4	31.5	51.1	2.6	20.1	31.4	42.8	9.2	26.2	36.3	51.6	7.4	18.4	32.8	49.6	9.5	23.8	36.4	50.5
OPIM	13.4	26.9	37.4	50.9	6.6	19.4	28.9	48.6	5.7	17.7	29.7	50.1	2.4	20.1	34.4	46.8	9.6	25.3	36.6	51.7	7.6	18.5	32.9	48.9	9.7	23.7	36.6	50.3
SubSIM	10.1	25.7	36.8	51.1	4.8	15.4	27.9	44.8	4.6	19.2	31.7	50.2	3.6	18.8	37.6	44.7	9.5	26.7	36.5	51.5	7.5	18.9	33.3	49.4	9.3	23.1	36.5	50.6
OIM	8.9	27.6	38.0	51.3	4.2	16.7	26.5	48.2	5.7	17.5	31.9	50.8	2.0	18.5	36.3	42.2	9.6	26.2	36.7	51.3	7.8	18.2	33.1	49.6	-	-	-	-
IMINFECTOR	9.6	26.8	37.7	50.6	5.4	17.9	27.8	47.6	5.4	18.2	31.6	50.9	3.6	19.7	37.5	45.9	9.1	26.2	36.1	51.5	7.9	18.6	33.5	49.8	9.4	23.5	36.9	50.3
PIANO	9.8	25.2	37.4	51.1	4.7	16.3	27.1	47.2	5.3	18.1	31.7	50.2	2.2	19.2	36.6	43.2	9.1	26.4	36.2	51.6	-	-	-	-	-	-	-	-
ToupleGDD	10.6	27.5	38.5	51.5	6.3	17.8	28.3	50.5	5.4	19.3	31.6	51.3	3.3	20.4	37.2	45.7	9.5	26.8	37.1	51.4	-	-	-	-	-	-	-	-
DeepIM$_s$	13.6	27.7	38.5	51.8	6.9	19.1	29.3	50.5	5.9	20.2	31.7	51.5	3.8	21.4	38.9	47.1	10.2	26.8	37.5	51.8	7.9	18.8	33.7	50.3	10.1	24.7	36.8	50.8
DeepIM	**14.1**	**28.1**	**39.6**	**52.4**	**7.8**	**20.9**	**31.5**	**51.2**	**6.3**	**21.0**	**32.5**	**52.4**	**4.9**	**23.3**	**41.5**	**49.9**	**11.6**	**27.4**	**38.7**	**52.1**	**8.4**	**19.3**	**34.2**	**51.3**	**11.2**	**26.5**	**37.9**	**51.8**

Table 2. Performance comparison under IC diffusion pattern. – indicates out-of-memory error. (Best is highlighted with bold.)

	Cora-ML				Network Science				Power Grid				Jazz				Synthetic				Digg				Weibo			
Methods	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%
IMM	1.7	34.8	52.2	66.4	2.5	11.9	18.1	33.6	4.6	19.9	31.7	56.9	1.4	5.7	13.4	24.5	1.1	5.2	13.1	66.9	2.4	10.8	37.4	55.6	1.6	6.7	19.3	45.2
OPIM	2.3	36.9	51.2	71.5	1.6	12.0	18.8	34.1	4.4	21.6	29.4	55.5	1.4	6.9	12.6	20.9	1.3	5.2	12.6	62.1	2.1	11.3	38.2	57.1	1.8	6.1	18.7	46.6
SubSIM	1.7	33.6	54.7	70.1	1.8	10.4	19.2	34.1	4.5	21.1	31.2	57.4	1.4	5.9	11.4	21.2	1.4	5.5	13.1	69.6	2.4	11.3	37.9	56.9	1.7	6.7	19.2	46.8
IMINFECTOR	2.1	33.9	51.3	70.6	2.1	11.8	18.7	34.5	4.2	21.3	31.6	56.2	1.4	6.2	13.5	22.8	1.3	5.2	12.9	67.4	2.2	11.1	38.9	58.7	1.8	6.4	18.6	47.5
PIANO	2.1	33.5	53.3	69.8	2.1	11.3	19.1	33,9	4.3	21.3	31.4	57.1	1.1	6.2	12.1	22.4	1.2	5.2	12.9	67.4	-	-	-	-	-	-	-	-
ToubleGDD	2.3	36.2	54.5	70.9	2.8	12.4	19.8	34.6	4.8	21.9	32.6	58.1	1.4	6.5	12.9	23.6	1.3	5.5	13.4	70.2	-	-	-	-	-	-	-	-
DeepIM$_s$	10.7	65.6	75.1	85.2	3.5	14.6	23.8	37.8	5.1	22.9	40.3	65.1	1.4	6.5	14.2	85.3	1.5	6.0	14.2	90.3	3.1	13.3	39.2	67.9	2.5	7.1	32.6	68.4
DeepIM	**13.4**	**69.2**	**83.5**	**94.1**	**4.1**	**16.6**	**26.7**	**41.5**	**6.3**	**24.4**	**46.8**	**71.7**	**1.9**	**6.5**	**16.4**	**99.1**	**1.5**	**6.5**	**15.5**	**99.9**	**3.5**	**15.9**	**41.3**	**76.2**	**3.1**	**7.6**	**39.3**	**72.4**

Table 3. Performance comparison under LT diffusion pattern. – indicates out-of-memory error. (Best is highlighted with bold.)



(a) Cora_ML-IC (b) Net Science-IC (c) Power Grid-IC (d) Jazz-IC (e) Synthetic-IC

(f) Cora_ML-LT (g) Net Science-LT (h) Power Grid-LT (i) Jazz-LT (j) Synthetic-LT

Figure 2. The influence spread (total infected nodes) in the y-axis under the constraint of the budget with the node size growth (x-axis: 1%, 5%, 10%, and 20%). Fig. 2a - 2e and Fig. 2f - 2j are evaluated under the IC and LT model, respectively.

CELF (Leskovec et al., 2007). In addition, we also compare the performance of the student model DeepIM$_s$ (i.e., coupled with the simplified diffusion model $M_s(\cdot)$).

5.3. Quantitative Analysis

We evaluate the performance of DeepIM in maximizing the influence against other approaches under various IM application schemes. Each model selects 1%, 5%, 10%, and 20% nodes in each dataset as seed nodes, and we allow each diffusion model to simulate until the diffusion process stops and record the average influence spread of 100 rounds. We report the percentage of final infected nodes (i.e., the number of infected nodes/the total number of nodes).

IM under IC Model. We first examine the effectiveness of DeepIM against other baseline methods under the IC diffusion pattern. As shown in Table 2, DeepIM can achieve an overall better performance than other methods across all datasets. Compared to traditional methods, IMM, OPIM, and SubSIM are three state-of-the-art methods based on reserve-set sampling and various approximation techniques, which generate similar results across all datasets; however, they rely on different heuristics to guide the node selection for efficiency improvement and fail to decode the underlying distribution of seed sets. OIM achieves better performance than traditional methods in most datasets because it can automatically update the edge weight iteratively. However, the disadvantage of OIM is also obvious: it is tailored for the specific IC diffusion model, which is less applicable in real-world scenarios. Lastly, the learning-based IM methods (IMINFECTOR, PIANO, and ToupleGDD) achieve competitive and generally better performance than traditional ones due to their larger model size and better generalization ability. However, learning-based methods that leverage reinforcement learning experience scalability issues and they cannot be applied in billion-scale networks (e.g., Digg and Weibo), which makes them hard to be applied in real-world scenarios. Compared to learning-based methods, DeepIM proposes a more robust way of learning the end-to-end diffusion model and searching the high-influential node set in latent space directly, which can better capture the underlying diffusion dynamics and resolve the scalability issues. In addition, DeepIM$_s$ incorporates a lightweight end-to-end learning-based diffusion model that could retain both effi-

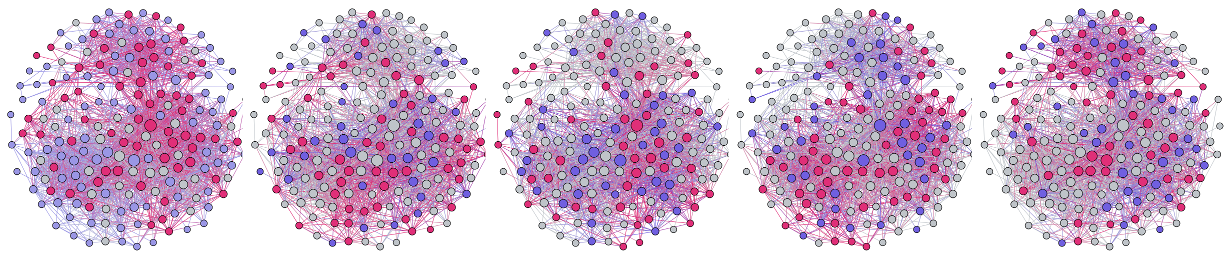

(a) DeepIM-20% (b) OIM-20% (c) OPIM-20% (d) SubSIM-20% (e) ToupleGDD-20%

Figure 3. The visualization of influence spread in Jazz dataset: The size of nodes is determined by the node degree, and the color on nodes determines the infection status: blue means the node is in seed set, red means the node is infected, and grey means the node is not infected.

	10,000	20,000	30,000	50,000	50,000 (Training)
IMINFECTOR	3.478s	7.842s	12.376s	16.492s	4753.67s
PIANO	5.948s	10.532s	16.575s	28.437s	14732.63s
ToupleGDD	10.476s	19.583	32.792s	58.985s	–
DeepIM$_s$	0.312s	0.616s	0.847s	1.275s	503.12s
DeepIM	1.402s	2.798s	5.124s	12.882s	1244.56s

Table 4. The average inference runtime (in seconds) with regard to the increase of node size and the average training time. We select 10% of nodes as the seeds.

cacy and efficiency than other learning-based methods.

IM under LT Model. We then evaluate the final influence spread with respect to the initial seed set size by assuming LT as the diffusion model. As shown in Table 3, DeepIM can generate more superior seed sets to infect the most number of nodes and excel other methods by an evident margin across all datasets. Notably, DeepIM demonstrates its superiority in the Synthetic dataset that can effectively spread the influence to the whole network when choosing 20% of the node as the initial seed set, yet other methods can infect at most 70% nodes of the network. Specifically, DeepIM and DeepIM$_s$ excel other methods by on average 200% in the Jazz dataset and 30% in the Synthetic dataset. The reason is largely due to the lack of generalization capability of other methods under various diffusion models.

IM with Budget Constraint. We then compare the quality of the seed sets generated by DeepIM and CELF under the IC and LT model with the budget constraint, and such a budget is explicitly defined as the node degree in this paper. As can be seen from Fig. 2, our proposed method generally performs better than CELF across all networks of different sizes, and the margins are more evident under the LT model (Fig. 2f - 2j). In addition, compared to CELF, the growths of influence spread in DeepIM have fewer fluctuations across all datasets, which also demonstrates the stability of DeepIM because of its capability of identifying the latent distribution seed sets while considering the budget constraint.

5.4. Scalability Analysis

We record the runtime of the seed set inference with regard to the increase in node size against other learning-based IM solutions. As can be clearly seen in Table 4, DeepIM demonstrates near-linear growth of runtime as the graph size increases. In addition, it achieves a generally shorter inference time (on average 20% faster inference time than the second-fast IMINFECTOR) compared to other learning-based methods. In addition, our DeepIM$_s$ coupled with a lightweight end-to-end diffusion model can greatly reduce the computational cost of estimating the expected influence spread, and achieves even 90% improvement in the inference time on average than our DeepIM model.

5.5. Case Study: Graph Diffusion Visualization

Finally, we conduct a case study to demonstrate the distribution of selected 20% seed nodes as well as the final infection status of all nodes in Fig. 3, where blue nodes indicate the initial seed node, red nodes indicate the infected node during the influence spread, and grey nodes represent uninfected nodes. Due to the ease of representation, we only visualize the result of the Jazz dataset because of its overall smaller graph size. Overall, DeepIM demonstrates better performance in terms of spreading influence. Due to the space limit, we compare the influence spread result between different initial seed set sizes, namely 10% and 20%, in the appendix and provide more discussions there.

6. Conclusion

In this paper, we propose a novel framework to tackle the IM problem in a more robust and generalized way than existing learning-based IM methods. Particularly, to characterize the complex nature of the seed set, we propose to character the probability of the seed set and directly search for a more optimal seed set in continuous space. Furthermore, to solve the challenge of modeling the underlying diffusion pattern, we offer two different learning-based diffusion models to characterize the diversified diffusion dynamics with efficiency and efficacy guarantee. Finally, we propose a novel objective function that can be coupled with multiple constraints for seed node set inference, which can adapt to different IM application schemes. Extensive experiments and case studies on both synthetic and real-world datasets demonstrate the advantages of DeepIM over existing state-of-the-art methods to maximize the influence spread.

References

Ali, K., Wang, C.-Y., and Chen, Y.-S. Boosting reinforcement learning in competitive influence maximization with transfer learning. In *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pp. 395–400. IEEE, 2018.

Banerjee, S., Jenamani, M., and Pratihar, D. K. A survey on influence maximization in a social network. *KAIS*, 62(9): 3417–3455, 2020.

Chamberlain, B., Rowbottom, J., Gorinova, M. I., Bronstein, M., Webb, S., and Rossi, E. Grand: Graph neural diffusion. In *Proc. of the ICML*, pp. 1407–1418, 2021.

Chen, T., Yan, S., Guo, J., and Wu, W. Touplegdd: A fine-designed solution of influence maximization by deep reinforcement learning. *arXiv preprint arXiv:2210.07500*, 2022.

Chen, W., Wang, C., and Wang, Y. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *Proc. of the KDD*, pp. 1029–1038, 2010.

Dolhansky, B. W. and Bilmes, J. A. Deep submodular functions: Definitions and learning. *Advances in Neural Information Processing Systems*, 29, 2016.

Du, N., Liang, Y., Balcan, M., and Song, L. Influence function learning in information diffusion networks. In *ICML*, pp. 2016–2024, 2014.

Erdős, P., Rényi, A., et al. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960.

Guo, Q., Wang, S., Wei, Z., and Chen, M. Influence maximization revisited: Efficient reverse reachable set generation with bound tightened. In *Proc. of the SIGMOD*, pp. 2167–2181, 2020.

Kamarthi, H., Vijayan, P., Wilder, B., Ravindran, B., and Tambe, M. Influence maximization in unknown social networks: Learning policies for effective graph sampling. *arXiv preprint arXiv:1907.11625*, 2019.

Kempe, D., Kleinberg, J., and Tardos, É. Maximizing the spread of influence through a social network. In *Proc. of the KDD*, 2003.

Kermack, W. O. and McKendrick, A. G. A contribution to the mathematical theory of epidemics. *Proceedings of the royal society of london. Series A, Containing papers of a mathematical and physical character*, 115(772):700–721, 1927.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Ko, J., Lee, K., Shin, K., and Park, N. Monstor: An inductive approach for estimating and maximizing influence over unseen networks. In *Proc. of the ASONAM*, pp. 204–211, 2020.

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

Lei, S., Maniu, S., Mo, L., Cheng, R., and Senellart, P. Online influence maximization. In *Proc. of the KDD*, 2015.

Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., and Glance, N. Cost-effective outbreak detection in networks. In *Proc. of the KDD*, 2007.

Li, H., Xu, M., Bhowmick, S. S., Sun, C., Jiang, Z., and Cui, J. Disco: Influence maximization meets network embedding and deep learning. *arXiv preprint arXiv:1906.07378*, 2019a.

Li, H., Xu, M., Bhowmick, S. S., Rayhan, J. S., Sun, C., and Cui, J. Piano: Influence maximization meets deep reinforcement learning. *IEEE Transactions on Computational Social Systems*, 2022.

Li, X., Smith, J. D., Dinh, T. N., and Thai, M. T. Tiptop:(almost) exact solutions for influence maximization in billion-scale networks. *IEEE/ACM Transactions on Networking*, 27(2):649–661, 2019b.

Li, Y., Fan, J., Wang, Y., and Tan, K.-L. Influence maximization on social graphs: A survey. *TKDE*, 30(10): 1852–1872, 2018.

Lin, S.-C., Lin, S.-D., and Chen, M.-S. A learning-based framework to handle multi-round multi-party influence maximization on social networks. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 695–704, 2015.

Lin, Y., Chen, W., and Lui, J. C. Boosting information spread: An algorithmic approach. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, pp. 883–894, 2017.

Ling, C., Yang, C., and Zhao, L. Deep generation of heterogeneous networks. In *2021 IEEE International Conference on Data Mining (ICDM)*, pp. 379–388. IEEE, 2021.

Ling, C., Chowdhury, T., Jiang, J., Wang, J., Zhang, X., Chen, H., and Zhao, L. Deepgar: Deep graph learning for analogical reasoning. In *2022 IEEE International Conference on Data Mining (ICDM)*, pp. 1065–1070, Los Alamitos, CA, USA, Dec 2022a.

Ling, C., Jiang, J., Wang, J., and Zhao, L. Source localization of graph diffusion via variational autoencoders for graph inverse problems. In *Proc. of the KDD*, 2022b.

Ling, C., Cao, H., and Zhao, L. Stgen: Deep continuous-time spatiotemporal graph generation. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2022, Grenoble, France, September 19–23, 2022, Proceedings, Part III*, pp. 340–356, 2023a.

Ling, C., Yang, C., and Zhao, L. Motif-guided heterogeneous graph deep generation. *Knowledge and Information Systems*, pp. 1–26, 2023b.

Manchanda, S., Mittal, A., Dhawan, A., Medya, S., Ranu, S., and Singh, A. Gcomb: Learning budget-constrained combinatorial algorithms over billion-sized graphs. *Advances in Neural Information Processing Systems*, 33: 20000–20011, 2020.

McCallum, A. K., Nigam, K., Rennie, J., and Seymore, K. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000.

Nguyen, H. T., Thai, M. T., and Dinh, T. N. Stop-and-stare: Optimal sampling algorithms for viral marketing in billion-scale networks. In *Proc. of the SIGMOD*, 2016.

Nguyen, H. T., Thai, M. T., and Dinh, T. N. A billion-scale approximation algorithm for maximizing benefit in viral marketing. *IEEE/ACM Transactions On Networking*, 25 (4):2419–2429, 2017.

Panagopoulos, G., Malliaros, F., and Vazirgiannis, M. Multi-task learning for influence estimation and maximization. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

Rossi, R. A. and Ahmed, N. K. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015.

Saito, K., Kimura, M., Ohara, K., and Motoda, H. Efficient discovery of influential nodes for sis models in social networks. *Knowledge and information systems*, 30(3): 613–635, 2012.

Tang, J., Tang, X., Xiao, X., and Yuan, J. Online processing algorithms for influence maximization. In *Proc. of the SIGMOD*, pp. 991–1005, 2018.

Tang, Y., Xiao, X., and Shi, Y. Influence maximization: Near-optimal time complexity meets practical efficiency. In *Proc. of the SIGMOD*, pp. 75–86, 2014.

Tang, Y., Shi, Y., and Xiao, X. Influence maximization in near-linear time: A martingale approach. In *Proc. of the SIGMOD*, 2015.

Tian, S., Mo, S., Wang, L., and Peng, Z. Deep reinforcement learning-based approach to tackle topic-aware influence maximization. *Data Science and Engineering*, 5(1):1–11, 2020.

Vaswani, S., Kveton, B., Wen, Z., Ghavamzadeh, M., Lakshmanan, L. V., and Schmidt, M. Model-independent online learning for influence maximization. In *ICML*, 2017.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Wang, J., Jiang, J., and Zhao, L. An invertible graph diffusion neural network for source localization. In *Proceedings of the ACM Web Conference 2022*, pp. 1058–1069, 2022a.

Wang, J., Li, H., Chai, Z., Wang, Y., Cheng, Y., and Zhao, L. Toward quantized model parallelism for graph-augmented mlps based on gradient-free admm framework. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–11, 2022b. doi: 10.1109/TNNLS.2022.3223879.

Wang, Y., Fan, Q., Li, Y., and Tan, K.-L. Real-time influence maximization on dynamic social streams. *arXiv preprint arXiv:1702.01586*, 2017.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Philip, S. Y. A comprehensive survey on graph neural networks. *IEEE TNNLS*, 32(1):4–24, 2020.

Xia, W., Li, Y., Wu, J., and Li, S. Deepis: Susceptibility estimation on social networks. In *Proc. of the WSDM*, pp. 761–769, 2021.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

Yang, L., Li, Z., and Giua, A. Containment of rumor spread in complex social networks. *Information Sciences*, 506: 113–130, 2020.

Ye, M., Liu, X., and Lee, W.-C. Exploring social influence for recommendation: a generative model approach. In *Proc. of the SIGIR*, pp. 671–680, 2012.

Zhang, C., Li, W., Wei, D., Liu, Y., and Li, Z. Network dynamic gcn influence maximization algorithm with leader fake labeling mechanism. *IEEE Transactions on Computational Social Systems*, 2022.

A. Proofs

The proof of Theorem 1 is demonstrated as follows.

Proof. $g_u(x, G; \theta) = \mathcal{A}^1 \circ (C^1 \circ \mathcal{A}^2 \circ C^2 \cdots \circ \mathcal{A}^K \circ C^K)$ via iterating Eq. (1) recursively. Because $\mathcal{A}^k$ and C^k are non-decreasing, so is $\mathcal{A}^1 \circ C^1 \cdots \circ \mathcal{A}^K \circ C^K$, which is g_u. Therefore, M is infection monotonic. Because g_u and g_r are non-decreasing, M is also non-decreasing and hence score monotonic. □

The proof of Corollary 2 is demonstrated as follows.

Proof. For the GAT model, $a^k_{i,j} = \mathcal{A}^k(h^k_i, h^k_j, \theta^k) = \theta^k_1(\theta^k_2 h^{k-1}_i || \theta^k_2 h^{k-1}_j)$ and $h^k_i = C^k(a^k, \theta^k) = \max(\sum_{j \in N(i)} softmax(LeakyReLU(a^k_{i,j}))\theta^k_1 h^{k-1}_j, 0)$ where $\theta^k = [\theta^k_1; \theta^k_2]$, and $||$ denotes the concatenation of two vectors. $\mathcal{A}^k$ are non-decreasing and non-negative because $\theta^k \geq 0$ (i.e. $\theta^k_1 \geq 0$ and $\theta^k_2 \geq 0$). In other words, $a^k_{i,j}$ is non-negative, and h^k_i, θ^k_1 and the softmax operator are non-negative. Therefore, the LeakyReLU operator and $\max(\bullet, 0)$ can be removed from C^k. That is, $h^k_i = C^k(a^k, \theta^k) = \sum_{j \in N(i)} softmax(a^k_{i,j})\theta^k_1 h^{k-1}_j$. Because the softmax operator is non-decreasing, and θ^k_1 is non-negative. C^k is non-decreasing. Hence, The GAT model satisfies the conditions in Theorem 1 and thus M is score and infection monotonic. □

The proof of Corollary 3 is illustrated as follows.

Proof. According to Theorem 1, the GNN-based $M(x, G; \theta)$ is monotonic. Then for any two $x^{(i)} > x^{(j)}$, we have $M(x^{(i)}, G; \theta) > M(x^{(j)}, G; \theta)$. If the reconstruction error is minimized during the training of $f_\psi(\cdot)$, we also have $f_\psi(z^{(i)}) > f_\psi(z^{(j)})$. Hence, $M(f_\psi(z^{(i)}), G; \theta) > M(f_\psi(z^{(j)}), G; \theta)$ also holds. □

The derivation of Equation 8 is shown as follows.

$$\mathcal{L}_{\text{pred}} = \min_z \mathbb{E}\big[-\log p_\theta(y|x, G) - \log p_\psi(x|z)\big],$$
$$\text{s.t.} \sum\nolimits_{i=0}^{|V|} \mathcal{F}(v_i, G) \cdot x_i \leq k,$$

Since we assume the optimal $\tilde{y} = |V|$ and the predicted $y \in [0, |V|]$, the optimization target is to maximize the y until it reaches the fully infected status. Therefore, the first term in Equation (8) is written as the Mean Squared Loss (MSE): $\|\tilde{y} - M(x, G; \theta)\|^2_2$. For the second term in Equation (8), the value range of x after the autoencoder is $[0, 1]$, indicating the probability of each node being selected to the seed set. $x \in [0, 1]$ fits the binomial distribution so that minimizing the negative log-likelihood is equivalent to minimizing the probability mass function. Therefore, the second term of the above function can be written to $-\log\big[\prod_i^{|V|} f_\psi(z_i)^{x_i}(1 - f_\psi(z_i)^{1-x_i}\big]$. Adding both terms gives us the final expression as shown in Equation (8):

$$\mathcal{L}_{\text{pred}} = \max_z \mathbb{E}\big[p_\theta(y|x, G) \cdot p_\psi(x|z)\big],$$
$$\text{s.t.} \sum\nolimits_{i=0}^{|V|} \mathcal{F}(v_i, G) \leq k.$$

B. More Experiment

B.1. Data

The statistics of datasets are depicted in Table 1, and we also provide a more detailed dataset description as follows. *1) Jazz* (Rossi & Ahmed, 2015). This dataset is a Jazz musicians collaboration network, where each node represents a musician and each edge represents two musicians who have played together in a band. *2) Cora-ML* (McCallum et al., 2000). This network contains computer science research papers, where each node represents a paper and each edge represents one paper cites the other one. *3) Power Grid* (Rossi & Ahmed, 2015). This is a topology network of the US Western States Power Grid. An edge represents a power supply line. A node is either a generator, a transformation, or a substation. *4) Network Science* (Rossi & Ahmed, 2015). This is a coauthorship network between scientists working on network theory, where nodes represent scientists and edges represent two scientists who have collaborated. *5) Digg* (Panagopoulos et al., 2020). A directed network of social media where users follow each other and a vote to a post allows followers to see the post. *6) Weibo* (Panagopoulos et al., 2020). A directed follower network where a cascade is defined by the first tweet and the list of retweets.

B.2. Hyperparameter Setting.

For each baseline, we set hyperparameters according to their original papers and fine-tune them on each dataset. For the configuration of each diffusion model, we use a weighted cascade version of the IC model, i.e., the propagation probability $p_{u,v} = 1/d^{in}_v$ (d^{in}_v denotes the in-degree of node v) for each edge $e = (u, v)$ on graph G; For LT model, the threshold θ is set to be uniformly sampled from $[0.3, 0.6]$ for each node v; the infection probability and recovery probability are set to be 0.001 in the SIS model. For DeepIM, the 2-layer GAT-structured diffusion estimation model that each layer contains 4 attention heads and the dimension of each attention channel is 64. Both encoder and decoder are

	Cora-ML				Network Science				Power Grid				Jazz				Synthetic				Digg				Weibo			
Methods	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%	1%	5%	10%	20%
Greedy	1.6	8.3	14.8	26.1	1.1	5.4	11.6	20.8	1.1	5.0	10.2	21.3	17.4	33.7	49.6	64.2	2.5	12.1	19.5	35.5	1.9	8.6	15.6	31.2	1.5	7.2	13.9	28.7
IMINFECTOR	2.1	9.4	16.1	27.9	1.7	5.8	12.4	22.3	1.3	5.5	12.4	23.1	8.8	35.4	54.8	66.2	2.5	12.4	20.5	36.2	2.3	9.1	16.4	32.4	2.5	8.5	15.5	29.6
IMM	2.0	9.5	15.4	27.6	1.3	5.6	12.2	22.1	1.1	5.6	11.0	22.9	7.6	37.8	55.6	67.1	2.7	12.6	20.9	37.3	2.5	9.4	16.3	32.6	2.3	8.1	15.7	29.4
OPIM	2.3	9.3	16.2	27.2	1.4	5.9	13.0	22.1	1.2	5.9	11.2	22.4	5.7	44.7	58.6	68.3	2.8	12.5	20.2	36.1	2.3	9.3	16.5	32.1	2.3	8.5	15.3	29.7
SubSIM	2.3	9.2	16.9	28.8	1.5	5.6	12.2	23.3	1.2	5.6	11.4	21.9	2.9	30.1	53.8	67.0	2.5	12.6	20.2	36.5	2.5	9.5	16.1	32.3	2.3	8.3	15.6	29.4
DeepIM	**7.1**	**16.1**	**21.9**	**30.8**	**2.7**	**8.7**	**15.1**	**25.1**	**1.9**	**7.6**	**13.3**	**23.8**	**27.1**	**57.1**	**68.1**	**74.1**	**3.2**	**14.4**	**24.5**	**39.1**	**5.6**	**11.4**	**18.8**	**36.3**	**6.5**	**13.1**	**17.1**	**32.3**

Table 5. Performance over comparison methods under SIS diffusion pattern. (Best is highlighted with bold.)

symmetric 4-layer MLP with hidden size 512, 1024, 1024, and 1024 for each layer, respectively. We choose Adam with learning rates 0.001 and 0.0001 for optimizing both Eq. (6) and Eq. (7), respectively.

B.3. IM under Non-progressive Diffusion Model

We demonstrate the performance of each model under the non-progressive SIS model. As shown in Table 5, we can observe a vast number of performance reductions regarding the final influence spread compared to Table 2 and Table 3, which is mainly due to the SIS diffusion model assumes the nodes can switch from activated to de-activated with a certain probability. Existing models all failed to capture the intrinsically more complicated diffusion dynamics. Nevertheless, DeepIM still exhibits better results and excels over others on average 10% on all datasets under such circumstances. To sum up, by jointly learning the seed set representation and the end-to-end diffusion estimation model, DeepIM illustrates its robustness by the capability of adapting to various underlying diffusion patterns and producing a generally competitive and stable influence spread.

B.4. Case Study: Graph Diffusion Visualization

Finally, we conduct a case study to demonstrate the distribution of selected seed nodes as well as the final infection status of all nodes in Fig. 4, where blue nodes indicate the initial seed node, red nodes indicate the infected node during the influence spread, and grey nodes represent uninfected nodes. We compare the influence spread result between different initial seed set sizes, namely 10% and 20%. Due to the ease of representation, we only visualize the result of the Jazz dataset because of its overall smaller graph size.

DeepIM demonstrates an overall better performance in terms of spreading the influence to a great extent. Notably, it can be visually seen from Fig. 4a and 4f that the final influence spread achieved by DeepIM with different initial seed set sizes is also very little, which means DeepIM can attain a better result with a lower cost comparing to others. The visualization demonstrates a consistent result with Table 2.

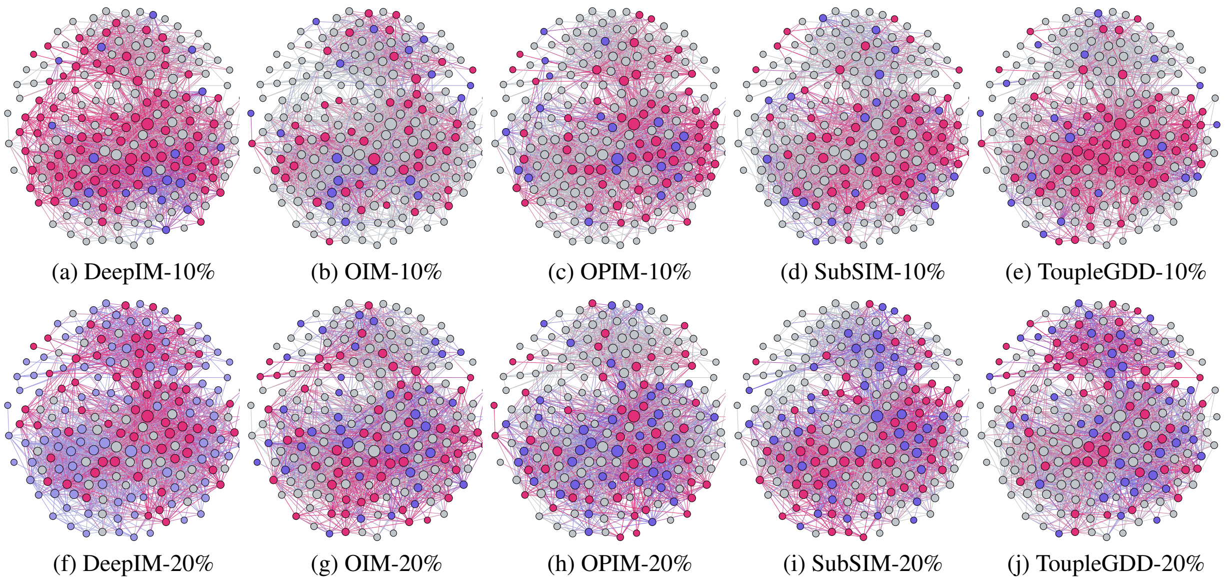


Figure 4. The visualization of influence spread in Jazz dataset: The size of nodes is determined by the node degree, and the color on nodes determines the infection status: blue means the node is in seed set, red means the node is infected, and grey means the node is not infected.