Social-Inverse: Inverse Decision-making of Social Contagion Management with Task Migrations

Guangmo Tong
Department of Computer and Information Sciences
University of Delaware
amotong@udel.edu

Abstract

Considering two decision-making tasks A and B, each of which wishes to compute an effective *decision* Y for a given *query* X, can we solve task B by using query-decision pairs (X, Y) of A without knowing the latent decision-making model? Such problems, called *inverse decision-making with task migrations*, are of interest in that the complex and stochastic nature of real-world applications often prevents the agent from completely knowing the underlying system. In this paper, we introduce such a new problem with formal formulations and present a generic framework for addressing decision-making tasks in social contagion management. On the theory side, we present a generalization analysis for justifying the learning performance of our framework. In empirical studies, we perform a sanity check and compare the presented method with other possible learning-based and graph-based methods. We have acquired promising experimental results, confirming for the first time that it is possible to solve one decision-making task by using the solutions associated with another one.

1 Introduction

Social contagion management. Social contagion, in its most general sense, describes the diffusion process of one or more information cascades spreading between a set of atomic entities through the underlying network [1, 2, 3, 4]. Prototypical applications of social contagion management include deploying advertising campaigns to maximize brand awareness [5, 6], broadcasting debunking information to minimize the negative impact of online misinformation [7, 8, 9], HIV prevention for homeless youth [10, 11], and the prevention of youth obesity [12, 13]. In these applications, a central problem is to launch new information cascades in response to certain input queries, with the goal of optimizing the agents' objectives [14, 15]. In principle, most of these tasks fall into either *diffusion enhancement*, which seeks to maximize the influence of the to-be-generated cascade (e.g., marketing campaign [5, 16, 17] and public service announcement [12, 18, 19]), or *diffusion containment*, which aims to generate positive cascades to minimize the spread of negative cascades (e.g., misinformation [20, 21, 22] and violence-promoting messages [23, 24]).

Inverse decision-making with task migrations. Traditional research on social contagion management often adopts classic operational diffusion models with known parameters, and focuses on algorithmic development in overcoming the NP-hardness [25, 17, 26, 27, 28]. However, real-world contagions are often very complicated, and therefore, perfectly knowing the diffusion model is less realistic [29, 30, 31, 32, 33]. When presented with management tasks defined over unknown diffusion models, one can adopt the learn-and-optimize approach in which modeling methodologies and optimization schemes are designed separately; in such methods, the main issue is that the learning process is guided by model accuracy but not by the optimization effect [34, 35], suggesting that

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

the endeavors dedicated to model construction are neither necessary nor sufficient for successfully handling the downstream optimization problems. This motivates us to explore unified frameworks that can shape the learning pipeline towards effective approximations. Recently, it has been shown that for contagion management tasks like diffusion containment, it is possible to produce high-quality decisions for future queries by using query-decision pairs from the same management task without learning the diffusion model [36]. Such findings point out an interesting and fundamental question: with a fixed latent diffusion model, can we solve a target management task by using query-decision pairs from a different management task? This is of interest because the agents often simultaneously deal with several management tasks while it is less likely that they always have the proper empirical evidence concerning the target task. For example, a network manager may need to work on a rumor blocking task, but they only have historical data collected in solving viral marketing tasks. We call such a setting as *inverse decision-making with task migrations*.

Contribution. This paper presents a formal formulation of inverse decision making where the target task we wish to solve is different from the source task that generates samples, with a particular focus on social contagion management tasks. Our main contribution is a generic framework, called *Social-Inverse*, for handling migrations between tasks of diffusion enhancement and diffusion containment. For Social-Inverse, we present theoretical analysis to obtain insights regarding how different contagion management tasks can be subtly correlated in order for samples from one task to help the optimization of another task. In empirical studies, we have observed encouraging results indicating that our method indeed works the way it is supposed to. Our main observations suggest that the task migrations are practically manageable to a satisfactory extent in many cases. In addition, we also explore the situations where the discrepancy between the target task and the source task is inherently essential, thereby making the samples from the source task less useful.

Roadmap. In Sec. 2, we first provide preliminaries regarding social contagion models, and then discuss how to formalize the considered problem. The proposed method together with its theoretical analysis is presented in Sec. 3. In Sec. 4, we present our empirical studies. We close our paper with a discussion on limitations and future works (Sec. 5). The technical proofs, source code, pre-train models, data, and full experimental analysis can be found in the supplementary material. The data and source code is maintained online[1].

2 Preliminaries

2.1 Stochastic diffusion model

A social network is given by a directed graph $G = (V, E)$, with V and E respectively denoting the user set and the edge set. In modeling the contagion process, let us assume that there are $L \in \mathbb{Z}$ information cascades $\{C_i\}_{i=1}^{L}$, each of which is associated with a seed set $S_i \subseteq V$. Without loss of generality, we assume that $S_i \cap S_j = \emptyset$ for $i \neq j$. A diffusion model $\mathcal{M}$ is governed by two sets of configurations: each node $u \in V$ is associated with a distribution $\mathcal{N}_u$ over $2^{N_u^-}$, where N_u^- is the set of the in-neighbors of u; each edge $(u, v) \in E$ is associated with a distribution $\mathcal{T}_{(u,v)}$ over $(0, +\infty)$ denoting the transmission time. During the diffusion process, a node can be inactive or C_i-active if activated by cascade C_i. Given the seed sets, the diffusion process unfolds as follows:

- **Initialization**: Each node u samples a subset $A_u \subseteq N_u^-$ following $\mathcal{N}_u$, and each edge (u, v) samples a real number $t_{(u,v)} \geq 0$ following $\mathcal{T}_{(u,v)}$.
- **Time** 0: The nodes in S_i become C_i-active at time 0, and other nodes are inactive.
- **Time** t: When a node u becomes C_i-active at time t, for each inactive node v such that u is in A_v, v will be activated by u and become C_i-active at time $t + t_{(u,v)}$. Each node will be activated by the first in-neighbor attempting to activate them and never deactivated. When a node v is activated by two or more in-neighbors at the same time, v will be activated by the cascade with the smallest index.

Remark 1. The considered model is in general highly expressive because $\mathcal{N}_u$ and $\mathcal{T}_{(u,v)}$ can be flexibly designed. For sample, it subsumes the classic independent cascade model [25] by making $\mathcal{N}_u$ sample each in-neighbor independently. When there is only one or two cascades, the above model generalizes a few popular diffusion models, including discrete-time independent cascade model [25],

[1] `https://github.com/cdslabamotong/social_inverse`

discrete-time linear threshold model [25], continuous-time independent cascade model [37], and independent multi-cascade model [38, 36].

Definition 1 (**Realization**). Notice that the initialization phase essentially samples a weighted subgraph, and the diffusion process becomes deterministic after the initialization phase. For an abstraction, we call each of such weighted subgraph a *realization*, and use $\mathcal{R}_G$ to denote the space of weighted subgraphs of G. With the concept of realization, we may abstract a concrete stochastic diffusion model $\mathcal{M}$ as a collection of density functions, i.e., $\mathcal{M} = \{\mathcal{N}_u : u \in V\} \cup \{\mathcal{T}_{(u,v)} : (u,v) \in E\}$. Slightly abusing the notation, we also use $\mathcal{M} : \mathcal{R}_G \to [0,1]$ to denote the distribution over $\mathcal{R}_G$ induced by the density functions specified by $\mathcal{M}$. On top of a diffusion model $\mathcal{M}$, the distribution of the diffusion outcome depends on the seed sets of the cascades. An example for illustrating the diffusion process is given in Appendix A.

2.2 Social contagion management tasks

In this paper, we focus on the following two classes of social contagion management tasks.

Problem 1 (**Diffusion Enhancement** (**DE**)). Given a diffusion model $\mathcal{M}$ and a set of target users $X \subseteq V$, we consider the single-cascade case and let $f_{\mathcal{M}}^{\mathrm{DE}}(X,Y)$ be the expected number of users in X who are activated by a cascade from seed set $Y \subseteq V$. We would like to find a seed set with at most $k \in \mathbb{Z}$ nodes such that the total influence on X can be maximized, i.e.,

$$\operatorname*{arg\,max}_{Y \subseteq V, |Y| \leq k} f_{\mathcal{M}}^{\mathrm{DE}}(X,Y). \tag{1}$$

Problem 2 (**Diffusion Containment** (**DC**)). Given a diffusion model $\mathcal{M}$, we now consider the situation of competitive diffusion where there is a negative cascade C_1 with a seed set $X \subseteq V$ and a positive cascade C_2 with a seed set $Y \subseteq V$. Let $f_{\mathcal{M}}^{\mathrm{DC}}(X,Y)$ be the expected number of users who are *not* activated by the negative cascade. Given the diffusion model $\mathcal{M}$ and the seed set X of the negative cascade, we would like to find a seed set Y for the positive cascade with at most $k \in \mathbb{Z}$ nodes such that the impact of the negative cascade can be maximally limited, i.e.,

$$\operatorname*{arg\,max}_{Y \subseteq V, |Y| \leq k} f_{\mathcal{M}}^{\mathrm{DC}}(X,Y). \tag{2}$$

Contagion management tasks like Problems 1 and 2 might be viewed as decision-making problems aiming to infer an effective decision Y in response to a query X . In such a sense, we may abstract such problems in the following way:

Problem 3 (**Abstract Contagion Management Tasks**). Given a diffusion model $\mathcal{M}$, an abstract management task T is specified by an objective function $f_{\mathcal{M}}^{T}(X,Y) : 2^V \times 2^V \to \mathbb{R}$, a candidate space $\mathcal{X}_T \subseteq 2^V$ of the queries, and a candidate space $\mathcal{Y}_T \subseteq 2^V$ of the decisions, where we wish to compute

$$Y_{\mathcal{M},T,X} \coloneqq \operatorname*{arg\,max}_{Y \in \mathcal{Y}_T} f_{\mathcal{M}}^{T}(X,Y) \tag{3}$$

for each input query $X \in \mathcal{X}_T$. We assume that $\mathcal{X}_T$ and $\mathcal{Y}_T$ are matroids over V, subsuming common constraints such as the cardinality constraint or k-partition [39]. Since such optimization problems are often NP-hard, their approximate solutions are frequently used, and we denote by $Y_{\mathcal{M},T,X}^{\alpha}$ an α-approximation to Equation 3.

Definition 2 (**Linearity over Kernel Functions**). In addressing the above management tasks, it is worth noting that the objective function is calculated over the possible diffusion outcomes, which are determined in the initialization phase. Specifically, denoting by $f_r^T(X,Y)$ the objective value projected to a single realization $r \in \mathcal{R}_G$, the objective function can be expressed as

$$f_{\mathcal{M}}^{T}(X,Y) = \int_{r \in \mathcal{R}_G} \mathcal{M}(r) \cdot f_r^T(X,Y) dr. \tag{4}$$

The function $f_r^T(X,Y)$ is called *kernel function*, in the sense that it transforms set structures into real numbers. For example, $f_r^{\mathrm{DE}}(X,Y)$ denotes the number of users in X who are activated by a cascade generated from Y in a single realization r; $f_r^{\mathrm{DC}}(X,Y)$ denotes the number of users in G who are not activated by the negative cascade (generated from X) in realization r when the positive cascade spreads from Y.

2.3 Inverse decision-making of contagion management with task migrations

Supposing that the diffusion model $\mathcal{M}$ is given, DE and DC are purely combinatorial optimization problems, which have been extensively studied [25, 20]. In the case that the diffusion model is unknown, inverse decision-making is a potential solution, which seeks to solve contagion management tasks by directly learning from query-decision pairs [40]. In particular, with respect to a certain management task $f_{\mathcal{M}}^{T}$ associated with an unknown diffusion model $\mathcal{M}$, the agent receives a collection of pairs (X_i, Y_i) where Y_i is the optimal/suboptimal solution to maximizing $f_{\mathcal{M}}^{T}(X_i, Y)$. Such empirical evidence can be mathematically characterized as

$$S_{\mathcal{M},T,m}^{\overline{\alpha}} = \left\{ (X_i, Y_{\mathcal{M},T,X_i}^{\overline{\alpha}}) : f_{\mathcal{M}}^{T}(X_i, Y_{\mathcal{M},T,X_i}^{\overline{\alpha}}) \geq \overline{\alpha} \cdot \max_{Y \in \mathcal{Y}_T} f_{\mathcal{M}}^{T}(X_i, Y) \right\}_{i=1}^{m} \tag{5}$$

where $\overline{\alpha} \in (0, 1]$ is introduced to measure the optimality of the sample decisions. For the purpose of theoretical analysis, the ratio $\overline{\alpha}$ may be interpreted as the best approximation ratio that can be achieved by a polynomial-time algorithm under common complexity assumptions (e.g., $NP \neq P$). For DE and DC, we have the best ratio as $1 - 1/e$ due to the well-known fact that their objective functions are submodular [25, 20]. Leveraging such empirical evidence, we wish to solve the same or a different management task for future queries:

Problem 4 (**Inverse Decision-making with Task Migrations**). Suppose that there is an underlying diffusion model $\mathcal{M}_{true}$. Consider two management tasks $T_{\textcircled{s}}$ and $T_{\textcircled{t}}$ defined in Problem 3, where $T_{\textcircled{s}}$ is the source task and $T_{\textcircled{t}}$ is the target task. With a collection $S_{\mathcal{M}_{true},T_{\textcircled{s}},m}^{\overline{\alpha}_{\textcircled{s}}}$ of samples concerning the source task $T_{\textcircled{s}}$ for some ratio $\overline{\alpha}_{\textcircled{s}} \in (0, 1]$, we aim to build a framework $A : \mathcal{X}_{T_{\textcircled{t}}} \to \mathcal{Y}_{T_{\textcircled{t}}}$ that can make a prediction $A(X)$ for each future query X of the target task $T_{\textcircled{t}}$. Let l be a loss function $l(X, \hat{Y}) : \mathcal{X}_{T_{\textcircled{t}}} \times \mathcal{Y}_{T_{\textcircled{t}}} \to [0, 1]$ that measures the desirability of $\hat{Y}$ with respect to X. We seek to minimize the generalization error $\mathcal{L}$ with respect to an unknown distribution $\mathcal{D}$ over $\mathcal{X}_{T_{\textcircled{t}}}$:

$$\mathcal{L}(A, \mathcal{D}, l) := \mathbb{E}_{X \sim \mathcal{D}} \Big[l(X, A(X)) \Big]. \tag{6}$$

Since $\mathcal{M}_{true}$ and m are fixed, we denote $S_{\mathcal{M}_{true},T_{\textcircled{s}},m}^{\overline{\alpha}_{\textcircled{s}}}$ as $S_{T_{\textcircled{s}}}^{\overline{\alpha}_{\textcircled{s}}}$ for conciseness. We will focus on the case where the source task and the target task are selected from DE and DC.

Remark 2. In general, the above problem appears to be challenging because the query-decision pairs of one optimization problem do not necessarily shed any clues on effectively solving another optimization problem. What makes our problem tractable is that the source task and the target task share the same underlying diffusion model $\mathcal{M}_{true}$. With the hope that the query-decision pairs of the source task can identify $\mathcal{M}_{true}$ to a certain extent, we may solve the target task with statistical significance, as evidenced in experiments. In such a sense, our setting is called *inverse* as it implicitly infers the structure of the underlying model from solutions, in contrast to the forward decision-making pipeline that seeks solutions based on given models.

3 Social-Inverse

In this section, we present a learning framework called Social-Inverse for solving Problem 4. Our method is inspired by the classic structured prediction [41] coupled with randomized kernels [40], which may be ultimately credited to the idea of Random Kitchen Sink [42]. Social-Inverse starts by selecting an empirical distribution $\mathcal{M}_{em}$ over $\mathcal{R}_G$ and a hyperparameter $K \in \mathbb{Z}$, and then proceeds with the following steps:

- **Hypothesis design.** Sample K iid realizations $R_K = \{r_1, ..., r_K\} \subseteq \mathcal{R}_G$ following $\mathcal{M}_{em}$, and obtain the hypothesis space $\mathcal{F}_{R_K} := \{H_{R_K,\mathbf{w}}^{\textcircled{t}}(X, Y) : \mathbf{w} = (w_1, ..., w_K) \in \mathbb{R}^K\}$ where $H_{R_K,\mathbf{w}}^{\textcircled{t}}$ is the affine combination of $f_r^{T_{\textcircled{t}}}$ over the realizations in R_K:

$$H_{R_K,\mathbf{w}}^{\textcircled{t}}(X, Y) := \sum_{i=1}^{K} w_i \cdot f_{r_i}^{T_{\textcircled{t}}}(X, Y). \tag{7}$$

- **Training.** Compute a prior vector $\widetilde{\mathbf{w}}$ using the training set $S_{T_{\textcircled{s}}}^{\overline{\alpha}_{\textcircled{s}}}$ (Sec. 3.2), and sample the final parameter $\overline{\mathbf{w}}$ from an isotropic Gaussian $Q(\gamma \cdot \widetilde{\mathbf{w}}, \mathcal{I})$ with a mean of $\widetilde{\mathbf{w}}$ scaled by γ. The selection of γ will be discussed in Sec. 3.1.

- **Inference.** Given a future query $X \in \mathcal{X}_{T_{\textcircled{t}}}$ of the target task $T_{\textcircled{t}}$, the prediction is made by solving the inference problem over the hypothesis in $\mathcal{F}_{R_K}$ associated with final weight $\overline{\mathbf{w}}$:

$$\arg\max_{Y \in \mathcal{Y}_{T_{\textcircled{t}}}} H^{\textcircled{t}}_{R_K, \overline{\mathbf{w}}}(X, Y). \tag{8}$$

  It is often NP-hard to solve the above inference problem in optimal, and therefore, we assume that an $\widetilde{\alpha}_{\textcircled{t}}$-approximation to Equation 8 – denoted by $Y^{\widetilde{\alpha}_{\textcircled{t}}}_{R_K, \overline{\mathbf{w}}, X}$ – is employed for some $\widetilde{\alpha}_{\textcircled{t}} \in (0, 1]$. Notice that the ratio $\widetilde{\alpha}_{\textcircled{t}}$ herein represents the inference hardness, while the ratio $\overline{\alpha}_{\textcircled{s}}$ associated with the training set measures the hardness of the source task.

In completing the above procedure, it is left to determine a) the prior distribution $\mathcal{M}_{em}$, b) the hyperparameter K, c) the scale factor γ, d) the training method for computing the prior vector $\widetilde{\mathbf{w}}$, and e) the inference algorithm for computing $Y^{\widetilde{\alpha}_{\textcircled{t}}}_{R_K, \overline{\mathbf{w}}, X}$. In what follows, we will first discuss how they may influence the generalization performance in theory, and then present methods for their selections. For the convenience of reading, the notations are summarized in Table 2 in Appendix B.

3.1 Generalization analysis

For Social-Inverse, given the fact the generation of $\overline{\mathbf{w}}$ is randomized, the generalization error is further expressed as

$$\mathcal{L}(\text{Social-Inverse}, \mathcal{D}, l) \coloneqq \mathbb{E}_{X \sim \mathcal{D}, \overline{\mathbf{w}} \sim Q} \left[l(X, Y^{\widetilde{\alpha}_{\textcircled{t}}}_{R_K, \overline{\mathbf{w}}, X}) \right]. \tag{9}$$

In deriving an upper bound with respect to the prior vector $\widetilde{\mathbf{w}}$, let us notice that the empirical risk is given by $l(X_i, Y^{\widetilde{\alpha}_{\textcircled{t}}}_{R_K, \overline{\mathbf{w}}, X_i})$, which is randomized by $\overline{\mathbf{w}} \sim Q(\gamma \cdot \widetilde{\mathbf{w}}, \mathcal{I})$. Thus, the prediction $Y^{\widetilde{\alpha}_{\textcircled{t}}}_{R_K, \overline{\mathbf{w}}, X_i}$ associated with a training input X_i is most likely one of those centered around $Y^{\widetilde{\alpha}_{\textcircled{t}}}_{R_K, \widetilde{\mathbf{w}}, X_i}$, and we will measure such concentration by their difference in terms of a fraction of the empirical risk associated with $\widetilde{\mathbf{w}}$. More specifically, controlled by a hyperparameter $\beta \in (0, \widetilde{\alpha}_{\textcircled{t}})$, for an input query X_i, the potential predictions are those within the margin:

$$\begin{aligned} &\mathrm{I}^{\textcircled{t}}_{R_K, \widetilde{\mathbf{w}}, X_i, \beta} \coloneqq \\ &\left\{ Y \in \mathcal{Y}_{\textcircled{t}} : \widetilde{\alpha}_{\textcircled{t}} \cdot H^{\textcircled{t}}_{R_K, \widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{t}}}_{R_K, \widetilde{\mathbf{w}}, X_i}) - H^{\textcircled{t}}_{R_K, \widetilde{\mathbf{w}}}(X_i, Y) \leq \beta \cdot H^{\textcircled{t}}_{R_K, \widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{t}}}_{R_K, \widetilde{\mathbf{w}}, X_i}) \right\}. \end{aligned} \tag{10}$$

The empirical risk is therefore given via the above margin:

$$\mathcal{L}_{em}(\widetilde{\mathbf{w}}, R_K, S^{\overline{\alpha}_{\textcircled{s}}}_{T_{\textcircled{s}}}, l) \coloneqq \frac{1}{m} \sum_{i=1}^{m} \max_{Y \in \mathcal{Y}_{\textcircled{t}}} l(X_i, Y) \cdot \mathbb{1}_{\mathrm{I}^{\textcircled{t}}_{R_K, \widetilde{\mathbf{w}}, X_i, \beta}}(Y), \tag{11}$$

where $\mathbb{1}_S(x) \in \{0, 1\}$ is the indicator function: $\mathbb{1}_S(x) = 1 \iff x \in S$. With the above progressions, we have the following result concerning the generalization error.

Theorem 1. *For each $\widetilde{\mathbf{w}} = (\widetilde{\mathrm{w}}_1, ..., \widetilde{\mathrm{w}}_K)$, $R_K \subseteq \mathcal{R}_G$, $\beta \in (0, \widetilde{\alpha}_{\textcircled{t}})$, and $\delta > 0$, with probability at least $1 - \delta$, we have*

$$\mathcal{L}(\textit{Social-Inverse}, \mathcal{D}, l) \leq \mathcal{L}_{em}(\widetilde{\mathbf{w}}, R_K, S^{\overline{\alpha}_{\textcircled{s}}}_{T_{\textcircled{s}}}, l) + \frac{\|\widetilde{\mathbf{w}}\|^2}{m} + \sqrt{\frac{\gamma^2 \|\widetilde{\mathbf{w}}\|^2 / 2 + \ln(m/\delta)}{2(m-1)}}$$

provided that

$$\gamma = \frac{\widetilde{\alpha}^2_{\textcircled{t}} + 1}{\min_p |\, \widetilde{\mathrm{w}}_p \,| \cdot \beta \cdot \widetilde{\alpha}_{\textcircled{t}}} \sqrt{2 \ln \frac{2mK}{\|\widetilde{\mathbf{w}}\|^2}} \tag{12}$$

The proof follows from the standard analysis of the PAC-Bayesian framework [43] coupled with the approximate inference [44] based on a multiplicative margin [40]; the extra caution we need to handle is that our margin (Equation 10) is parameterized by β. Notice that when β decreases, the regularization term becomes larger, while the margin set $\mathrm{I}^{\textcircled{t}}_{R_K, \widetilde{\mathbf{w}}, X_i, \beta}$ becomes smaller – implying a low empirical risk (Equation 11). In this regard, Theorem 1 presents an intuitive trade-off between the estimation error and the approximation error controlled by β.

Having seen the result for a general loss, we now seek to understand the best possible generalization performance in terms of the approximation loss l_{approx}:

$$l_{\text{approx}}(X, \hat{Y}) \coloneqq 1 - \frac{f^{T_{①}}_{\mathcal{M}_{true}}(X, \hat{Y})}{\max_{Y \in \mathcal{Y}_{①}} f^{T_{①}}_{\mathcal{M}_{true}}(X, Y)} \in [0, 1]. \tag{13}$$

Such questions essentially explore the realizability of the hypothesis space $\mathcal{F}_{R_K}$, which is determined by the empirical distribution $\mathcal{M}_{em}$ and the number of random realizations used to construct $\mathcal{F}_{R_K}$. We will see shortly how these factors may impact the generalization performance. By Theorem 1, when infinite samples are available, the empirical risk approaches to

$$\mathbb{E}_{X \sim \mathcal{D}} \Big[\max_{Y \in \mathcal{Y}_{T_{①}}} l_{\text{approx}}(X, Y) \cdot \mathbb{1}_{\mathrm{I}^{①}_{R_K, \widetilde{\mathbf{w}}, X, \beta}}(Y) \Big]. \tag{14}$$

The next result provides an analytical relationship between the complexity of the hypothesis space and the best possible generalization performance in terms of l_{approx}.

Theorem 2. *Let $\Delta \coloneqq \sup_r \frac{\mathcal{M}_{true}(r)}{\mathcal{M}_{em}(r)} \cdot \frac{\max f^{T_{①}}_r(X,Y)}{\min f^{T_{①}}_r(X,Y)}$ measure the divergence between $\mathcal{M}_{true}$ and $\mathcal{M}_{em}$ scaled by the range of the kernel function. For each $\epsilon > 0, \delta_1 > 0, \delta_2 > 0$, and $\mathcal{M}_{em}$, when K is $O(\frac{\Delta^2}{\epsilon^2 \cdot \delta_2^2}(\ln |\mathcal{Y}_{T_{①}}| + \ln \frac{1}{\delta_1}))$, with probability at least $1 - \delta_1$ over the selection of R_K, there exists a desired weight $\widetilde{\mathbf{w}}$ such that*

$$\Pr_{X \sim \mathcal{D}} \Big[\max_{Y \in \mathcal{Y}_{T_{①}}} l_{\text{approx}}(X, Y) \cdot \mathbb{1}_{\mathrm{I}^{①}_{R_K, \widetilde{\mathbf{w}}, X, \beta}}(Y) \leq 1 - \frac{\widetilde{\alpha}_{①} \cdot (\widetilde{\alpha}_{①} - \beta) \cdot (1 - \epsilon)}{(1 + \epsilon)} \Big] \geq 1 - \delta_2. \tag{15}$$

Remark 3. The above result has the implication that the best possible ratio in generalization is essentially bounded by $O(\widetilde{\alpha}_{①} \cdot (\widetilde{\alpha}_{①} - \beta))$. On the other hand, one can easily see that the target task (Equation 3) and the inference problem (Equation 8) suffer the same approximation hardness, and therefore, one would not wish for a true approximation error that is better than $\widetilde{\alpha}_{①}$; in this regard, the result in Theorem 2 is not very loose.

The results in this section demonstrate how the selections of K, $\mathcal{M}_{em}$, $\widetilde{\mathbf{w}}$, and β may affect the generalization performance in theory. Since the true model $\mathcal{M}_{true}$ is unknown, the prior distribution $\mathcal{M}_{em}$ can be selected to be uniform or Gaussian distribution. K and β can be taken as hyperparameters determining the model complexity. In addition, since the true loss l_{approx} is not accessible, one can take general loss functions. Given the fact that we are concerned with set structures rather than real numbers, we employ the zero-one loss, which is adopted also for the convenience of optimization. Therefore, it remains to figure out how to compute the prior vector $\widetilde{\mathbf{w}}$ from training samples as well as how to solve the inference problem (Equation 8), which will be discussed in the next part.

3.2 Training method

In computing the prior vector $\widetilde{\mathbf{w}}$, the main challenge caused by the task migration is that the target task on which we performance inference is different from the source task that generates training samples. Theorem 1 suggests that, ignoring the low-order terms, one may find the prior vector by minimizing the regularized empirical risk $\mathcal{L}_{em}(\widetilde{\mathbf{w}}, R_K, S^{\overline{\alpha}_{ⓢ}}_{T_{ⓢ}}, l) + \frac{\|\widetilde{\mathbf{w}}\|^2}{m}$. Directly minimizing such a quantity would be notoriously hard because the optimization problem is bilevel: optimizing over $\widetilde{\mathbf{w}}$ involves the term $Y^{\widetilde{\alpha}_{①}}_{R_K, \widetilde{\mathbf{w}}, X_i}$ which is obtained by solving another optimization problem depending on $\widetilde{\mathbf{w}}$ (Equations 10 and 11). Notably, since $H^{①}_{R_K, \widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{①}}_{R_K, \widetilde{\mathbf{w}}, X_i})$ is lower bounded by $\widetilde{\alpha}_{①} \cdot H^{①}_{R_K, \widetilde{\mathbf{w}}}(X_i, Y_{\mathcal{M}_{true}, T_{①}, X_i})$, replacing $Y^{\widetilde{\alpha}_{①}}_{R_K, \widetilde{\mathbf{w}}, X_i}$ with $Y_{\mathcal{M}_{true}, T_{①}, X_i}$ would allow for us to optimize an upper bound of the empirical risk. Seeking a large-margin formulation, this amounts to solving the following mathematical program under the zero-one loss [41, 45]:

$$\begin{aligned} \min \quad & \|\mathbf{w}\|^2 + C \sum_{i=1}^{m} \xi_i / m \\ \text{s.t.} \quad & \widetilde{\alpha}_{①}(\widetilde{\alpha}_{①} - \beta) \cdot H^{①}_{R_K, \mathbf{w}}(X_i, Y_{\mathcal{M}_{true}, T_{①}, X_i}) - H^{①}_{R_K, \mathbf{w}}(X_i, Y) \geq \xi_i, \ \forall i \in [m], \ \forall\, Y \in \mathcal{Y}_{①} \\ & \mathbf{w} \geq 0 \end{aligned} \tag{16}$$

where ξ_i is the slack variable and C is a hyperparameter [46]. However, our dataset concerns only about the source task $T_{\textcircled{s}}$ without informing $Y_{\mathcal{M}_{true},T_{\textcircled{t}},X_i}$ or its approximation. In order to see where we could feed the training samples into the training process, let us notice that the constraints in Equation 16 have an intuitive meaning: with respect to the target the task $T_{\textcircled{t}}$, a desired weight $\widetilde{\mathbf{w}}$ should lead to a score function $H^{\textcircled{t}}_{R_K,\widetilde{\mathbf{w}}}$ that can assign highest scores to the optimal solutions $Y_{\mathcal{M}_{true},T_{\textcircled{t}},X_i}$. Similar arguments also apply to the source task $T_{\textcircled{s}}$, as the weight $\widetilde{\mathbf{w}}$ implicitly estimates the true model $\mathcal{M}_{true}$, which is independent of the management tasks. This enables us to reformulate the optimization problem with respect to the source task $T_{\textcircled{s}}$ by using the following constraints:

$$\widetilde{\alpha}_{\textcircled{t}}(\widetilde{\alpha}_{\textcircled{t}} - \beta) \cdot H^{\textcircled{s}}_{R_K,\mathbf{w}}(X_i, Y^{\overline{\alpha}_{\textcircled{s}}}_{\mathcal{M}_{true},T_{\textcircled{s}},X_i}) - H^{\textcircled{s}}_{R_K,\mathbf{w}}(X_i, Y) \geq \xi_i,\ \forall i \in [m],\ \forall Y \in \mathcal{Y}_{\textcircled{t}} \quad (17)$$

where $H^{\textcircled{s}}_{R_K,\mathbf{w}}(X,Y) := \sum_{i=1}^{K} w_i \cdot f^{T_{\textcircled{s}}}_{r_i}(X,Y)$ is the score function corresponding to the source task $T_{\textcircled{s}}$. As desired, pairs of $(X_i, Y^{\overline{\alpha}_{\textcircled{s}}}_{\mathcal{M}_{true},T_{\textcircled{s}},X_i})$ are the exactly the information we have in the training data $S^{\overline{\alpha}_{\textcircled{s}}}_{T_{\textcircled{s}}}$. One remaining issue is that the acquired program (Equation 17) has an exponential number of constrains [45], which can be reduced to linear (in sample size) if the following optimization problem can be solved for each $\mathbf{w}$ and X_i:

$$\max_{Y \in \mathcal{Y}_{\textcircled{s}}} H^{\textcircled{s}}_{R_K,\mathbf{w}}(X_i, Y). \quad (18)$$

Provided that the above problem can be addressed, the entire program can be solved by several classic algorithms, such as the cutting plane algorithm [47] and the online subgradient algorithm [48]. Therefore, in completing the entire framework, it remains to solve Equations 8 and 18. For tasks of DE and DC, we delightedly have the following results.

Theorem 3. *When $T_{\textcircled{s}}$ and $T_{\textcircled{t}}$ are selected from DE and DC, Equations 8 and 18 are both NP-hard to solve in optimal, but both can be approximated within a ratio of $1 - 1/e$ in polynomial time.*

A concrete example of using Social-Inverse to solve Problem 4 is provided in Appendix C.

4 Empirical studies

Although some theoretical properties of our framework can be justified (Sec. 3.1), it remains open whether or not the proposed method is practically effective, especially given the fact that no prior work has attempted to solve one optimization problem by using the solutions to another one. In this section, we present our empirical studies.

4.1 Experimental settings

We herein present the key logic of our experimental settings and provide details in Appendix E.1.

The latent model $\mathcal{M}_{true}$ and samples (Appendix E.1.1). To generate a latent diffusion model $\mathcal{M}_{true}$, we first determine the graph structure and then fix the distributions $\mathcal{N}_u$ and $\mathcal{T}_{(u,v)}$ by generating random parameters. We adopt four graphs: a Kronecker graph [49], an Erdős-Rényi graph [50], a Higgs graph [51], and a Hep graph [52]). Given the underlying diffusion model $\mathcal{M}_{true}$, for each of DE and DC, we generate a pool of query-decision pairs (X_i, Y_i) for training and testing, where X_i is selected randomly from V and Y_i is the approximate solution associated with X_i (Theorem 3). As for Problem 4, there are four possible target-source pairs: DE-DE, DC-DE, DE-DC, and DC-DC.

Social-Inverse (Appendix E.1.2). With K and β being hyperparameters, to set up Social-Inverse, we need to specify the empirical distribution $\mathcal{M}_{em}$. We construct the empirical distribution by building three diffusion models $\mathcal{M}_q$ with $q \in \{0.1, 0.5, 1\}$, where a smaller q implies that $\mathcal{M}_q$ is closer to $\mathcal{M}_{true}$. In addition, we construct an empirical distribution $\mathcal{M}_\infty$ which is totally random and not close to $\mathcal{M}_{true}$ anywhere. For each empirical distribution, we generate a pool of realizations.

Competitors (Appendix E.1.3). Given the fact that Problem 4 may be treated as a supervised learning problem with the ignorance of task migration, we have implemented Naive Bayes (NB) [53], graph neural networks (GNN) [54], and a deep set prediction network (DSPN) [55]. In addition, we consider the High-Degree (HD) method, which is a popular heuristic believing that selecting the high-degree users as the seed nodes can decently solve DE and DC. A random (Random) method is also used as the baseline.

Table 1: **Results on Kronecker (Kro) and Erdős-Rényi (ER).** Each cell shows the mean of performance ratio with std. For Social-Inverse under each empirical distribution, the table shows the results with $K \in \{15, 30, 60\}$ and $\beta = 1$. The training size is 270, and the testing size is 540.

			K = 15	K = 30	K = 60		K = 15	K = 30	K = 60
Kro	**DC | DE**	$\mathcal{M}_\infty$	0.541 (0.014)	0.567 (0.016)	0.584 (0.006)	$\mathcal{M}_1$	0.748 (0.022)	0.794 (0.016)	0.811 (0.011)
		$\mathcal{M}_{0.5}$	0.767 (0.028)	0.837 (0.004)	0.851 (0.004)	$\mathcal{M}_{0.1}$	0.770 (0.025)	0.850 (0.018)	0.853 (0.040)
		NB: 0.64 (0.01)	**GNN**: 0.51 (0.05)	**DSPN**: 0.46 (0.14)	**HD**: 0.59 (0.01)	**Random**: 0.15 (0.01)			
	DE | DC	$\mathcal{M}_\infty$	0.846 (0.005)	0.913 (0.005)	0.985 (0.014)	$\mathcal{M}_1$	0.796 (0.056)	0.845 (0.040)	0.899 (0.030)
		$\mathcal{M}_{0.5}$	0.850 (0.036)	0.937 (0.058)	1.021 (0.040)	$\mathcal{M}_{0.1}$	0.862 (0.013)	0.953 (0.018)	1.041 (0.026)
		NB: 0.88 (0.01)	**GNN**: 0.76 (0.05)	**DSPN**: 0.57 (0.14)	**HD**: 0.89 (0.01)	**Random**: 0.26 (0.01)			
ER	**DC | DE**	$\mathcal{M}_\infty$	0.541 (0.010)	0.585 (0.018)	0.577 (0.007)	$\mathcal{M}_1$	0.744 (0.005)	0.752 (0.003)	0.754 (0.003)
		$\mathcal{M}_{0.5}$	0.825 (0.005)	0.830 (0.005)	0.830 (0.005)	$\mathcal{M}_{0.1}$	0.829 (0.002)	0.833 (0.004)	0.833 (0.004)
		NB: 0.78 (0.01)	**GNN**: 0.05 (0.02)	**DSPN**: 0.06 (0.03)	**HD**: 0.32 (0.01)	**Random**: 0.06 (0.01)			
	DE | DC	$\mathcal{M}_\infty$	0.526 (0.020)	0.677 (0.016)	0.750 (0.010)	$\mathcal{M}_1$	0.773 (0.016)	0.796 (0.010)	0.800 (0.009)
		$\mathcal{M}_{0.5}$	0.879 (0.002)	0.900 (0.002)	0.892 (0.002)	$\mathcal{M}_{0.1}$	0.886 (0.005)	0.902 (0.003)	0.895 (0.003)
		NB: 0.04 (0.01)	**GNN**: 0.05 (0.02)	**DSPN**: 0.04 (0.01)	**HD**: 0.09 (0.01)	**Random**: 0.04 (0.01)			

Training, testing, and evaluation (Appendix E.1.4) The testing size is 540, and the training size m is selected from $\{90, 270, 1350\}$. Given the training size and the testing size, the samples are randomly selected from the pool we generate; similarly, given K, the realizations used in Social-Inverse are also randomly selected from the pool we generate. For each method, the entire process is repeated five times, and we report the average performance ratio together with the standard deviation. The performance ratio is computed by comparing the predictions with the decisions in testing samples; larger is better.

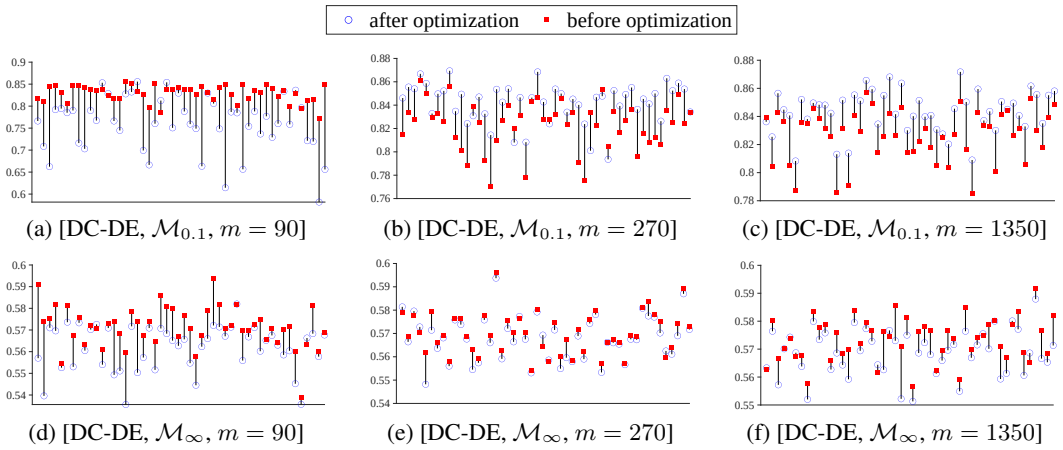
4.2 Main observations

The main results on the Kronecker graph and the Erdős-Rényi graph are provided in Table 1. According to Table 1, it is clear that Social-Inverse performs better when K becomes larger or when the discrepancy between $\mathcal{M}_{em}$ and $\mathcal{M}_{true}$ becomes smaller (i.e., q is small), which suggests that Social-Inverse indeed works the way it is supposed to. In addition, while all the methods are non-trivially better than Random, one can also observe that Social-Inverse easily outperforms other methods by an evident margin as long as sufficient realizations are provided. We also see that learning-based methods do not perform well in many cases; this is not very surprising because the effectiveness of learning-based methods hinges on the assumption that different tasks share similar decisions for the same query, which however may not be the case especially on the Erdős-Rényi graph. Furthermore, Social-Inverse appears to be more robust than other methods in terms of standard deviation. Finally, the performance of standard learning methods (e.g., NB and GNN) are sensitive to graph structures; they are relatively good on the Kronecker graph but less satisfactory on the Erdős-Rényi graph, while Social-Inverse is consistently effective on all datasets.

4.3 An in-depth investigation on task migration

Notably, the effectiveness of Social-Inverse depends not only on the training samples $S_{T_{①}}^{\overline{\alpha}_{①}}$ (for tuning the weight $\overline{\mathbf{w}}$) but also on the expressiveness of the hypothesis space (determined by $\mathcal{M}_{em}$ and K). Therefore, with solely the results in Table 1, we are not ready to conclude that samples of DE (resp., DC) are really useful for solving DC (resp., DE). In fact, when $\mathcal{M}_{em}$ is identical to $\mathcal{M}_{true}$, no samples are needed because setting $\overline{\mathbf{w}} = 1$ can allow for us to perfectly recover the best decisions as long as K is sufficiently large. As a result, the usefulness of the samples should be assessed by examining how much they can help in delivering a high-quality $\overline{\mathbf{w}}$. To this end, for each testing query, we report the quality of two predictions made based, respectively, on the initial weight (before optimization) and on the final weight (after optimization).

Such results for DC-DE on the Kronecker graph are provided in Figure 1. As seen from Figure 1b, the efficacy of DC samples in solving DE is statistically significant under $\mathcal{M}_{0.1}$, which might be the first



(a) [DC-DE, $\mathcal{M}_{0.1}$, $m = 90$] (b) [DC-DE, $\mathcal{M}_{0.1}$, $m = 270$] (c) [DC-DE, $\mathcal{M}_{0.1}$, $m = 1350$]

(d) [DC-DE, $\mathcal{M}_{\infty}$, $m = 90$] (e) [DC-DE, $\mathcal{M}_{\infty}$, $m = 270$] (f) [DC-DE, $\mathcal{M}_{\infty}$, $m = 1350$]

Figure 1: Each subgraph shows the results of 50 testing queries on the Kronecker graph with $K = 30$. For each query, we report the quality (i.e. performance ratio) of the predictions made based on the initial weight (before optimization) and final weight (after optimization).

piece of experimental evidence confirming that it is indeed possible to solve one decision-making task by using the query-decision pairs of another one. In addition, according to Figures 1a, 1b, and 1c, the efficacy of the samples is limited when the sample size is too small, and it does not increase too much after sufficient samples are provided. On the other hand, with Figure 1e, we also see that the samples of DC are not very useful when the empirical distribution (e.g., $\mathcal{M}_{\infty}$) deviates too much from the true model, and in such a case, providing more samples can even cause performance degeneration (Figure 1f), which is an interesting observation calling for further investigations.

The full experimental study can be found in Appendix E.2, E.3 and E.4, including the results on Higgs and Hep, results with more realizations, results of DE-DE and DC-DC, results of GNN under different seeds, a discussion on the impact of β, and a discussion of training efficiency.

5 Further discussion

We close our paper with a discussion on the limitations and future directions of the presented work, with the related works being discussed in Appendix F.

Future directions. Following the discussion on Figure 1, one immediate direction is to systemically investigate the necessary or sufficient condition in which the task migrations between DE and DC are manageable. In addition, settings in Problem 4 can be carried over to other contagion management tasks beyond DE and DC, such as effector detection [56] and diffusion-based community detection [57]. Finally, in its most general sense, the problem of inverse decision-making with task migrations can be conceptually defined over any two stochastic combinatorial optimization problems [40] sharing the same underlying model (e.g., graph distribution). For instance, considering the stochastic shortest path problem [58] and the minimum Steiner tree problem [59], with the information showing the shortest paths between some pairs of nodes, can we infer the minimum Steiner tree of a certain group of nodes with respect to the same graph distribution? Such problems are interesting and fundamental to many complex decision-making applications [60, 61].

Limitations. While our method offers promising performance in experiments, it is possible that deep architectures can be designed in a sophisticated manner so as to achieve improved results. In addition, while we believe that similar observations also hold for graphs that are not considered in our experiment, more experimental studies are required to support the universal superiority of Social-Inverse. In another issue, the assumption that the training set contains approximation solutions is the minimal one for the purpose of theoretical analysis, but in practice, such guarantees may never be known. Therefore, experimenting with samples of heuristic query-decision pairs is needed to further justify the practical utility of our method. Finally, we have not experimented with graphs of extreme scales (e.g., over 1M nodes) due to the limit in memory. We wish to explore the above issues in future work.

Acknowledgments and Disclosure of Funding

We thank the reviewers for their time and insightful comments. This work is supported in part by a) National Science Foundation under Award IIS-2144285 and b) the University of Delaware.

References

[1] N. A. Christakis and J. H. Fowler, "Social contagion theory: examining dynamic social networks and human behavior," *Statistics in medicine*, vol. 32, no. 4, pp. 556–577, 2013.

[2] R. S. Burt, "Social contagion and innovation: Cohesion versus structural equivalence," *American journal of Sociology*, vol. 92, no. 6, pp. 1287–1335, 1987.

[3] N. O. Hodas and K. Lerman, "The simple rules of social contagion," *Scientific reports*, vol. 4, no. 1, pp. 1–7, 2014.

[4] I. Iacopini, G. Petri, A. Barrat, and V. Latora, "Simplicial models of social contagion," *Nature communications*, vol. 10, no. 1, pp. 1–9, 2019.

[5] A. De Bruyn and G. L. Lilien, "A multi-stage model of word-of-mouth influence through viral marketing," *International journal of research in marketing*, vol. 25, no. 3, pp. 151–163, 2008.

[6] M. R. Subramani and B. Rajagopalan, "Knowledge-sharing and influence in online social networks via viral marketing," *Communications of the ACM*, vol. 46, no. 12, pp. 300–307, 2003.

[7] G. Tong, W. Wu, L. Guo, D. Li, C. Liu, B. Liu, and D.-Z. Du, "An efficient randomized algorithm for rumor blocking in online social networks," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 2, pp. 845–854, 2017.

[8] R. Yan, Y. Li, W. Wu, D. Li, and Y. Wang, "Rumor blocking through online link deletion on social networks," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 13, no. 2, pp. 1–26, 2019.

[9] M. Del Vicario, A. Bessi, F. Zollo, F. Petroni, A. Scala, G. Caldarelli, H. E. Stanley, and W. Quattrociocchi, "The spreading of misinformation online," *Proceedings of the National Academy of Sciences*, vol. 113, no. 3, pp. 554–559, 2016.

[10] A. Yadav, B. Wilder, E. Rice, R. Petering, J. Craddock, A. Yoshioka-Maxwell, M. Hemler, L. Onasch-Vera, M. Tambe, and D. Woo, "Bridging the gap between theory and practice in influence maximization: Raising awareness about hiv among homeless youth." in *IJCAI*, 2018, pp. 5399–5403.

[11] L. E. Young, J. Mayaud, S.-C. Suen, M. Tambe, and E. Rice, "Modeling the dynamism of hiv information diffusion in multiplex networks of homeless youth," *Social Networks*, vol. 63, pp. 112–121, 2020.

[12] M. H. Oostenbroek, M. J. van der Leij, Q. A. Meertens, C. G. Diks, and H. M. Wortelboer, "Link-based influence maximization in networks of health promotion professionals," *PloS one*, vol. 16, no. 8, p. e0256604, 2021.

[13] E. Rice, B. Wilder, L. Onasch-Vera, C. Hill, A. Yadav, S.-J. Lee, and M. Tambe, "A peer-led, artificial intelligence-augmented social network intervention to prevent hiv among youth experiencing homelessness."

[14] S. Aral, "Commentary—identifying social influence: A comment on opinion leadership and social contagion in new product diffusion," *Marketing Science*, vol. 30, no. 2, pp. 217–223, 2011.

[15] S. Aral and D. Walker, "Creating social contagion through viral product design: A randomized trial of peer influence in networks," *Management science*, vol. 57, no. 9, pp. 1623–1639, 2011.

[16] C. I. Dăniasă, V. Tomiţă, D. Stuparu, and M. Stanciu, "The mechanisms of the influence of viral marketing in social media," *Economics, Management & Financial Markets*, vol. 5, no. 3, pp. 278–282, 2010.

[17] W. Chen, C. Wang, and Y. Wang, "Scalable influence maximization for prevalent viral marketing in large-scale social networks," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010, pp. 1029–1038.

[18] J. Seymour, "The impact of public health awareness campaigns on the awareness and quality of palliative care," *Journal of Palliative Medicine*, vol. 21, no. S1, pp. S–30, 2018.

[19] T. W. Valente and P. Pumpuang, "Identifying opinion leaders to promote behavior change," *Health education & behavior*, vol. 34, no. 6, pp. 881–896, 2007.

[20] C. Budak, D. Agrawal, and A. El Abbadi, "Limiting the spread of misinformation in social networks," in *Proceedings of the 20th international conference on World wide web*, 2011, pp. 665–674.

[21] M. Farajtabar, J. Yang, X. Ye, H. Xu, R. Trivedi, E. Khalil, S. Li, L. Song, and H. Zha, "Fake news mitigation via point process based intervention," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1097–1106.

[22] X. He, G. Song, W. Chen, and Q. Jiang, "Influence blocking maximization in social networks under the competitive linear threshold model," in *Proceedings of the 2012 siam international conference on data mining*. SIAM, 2012, pp. 463–474.

[23] J. Fagan, D. L. Wilkinson, and G. Davies, "Social contagion of violence." in *Urban Seminar Series on Children's Health and Safety, John P. Kennedy School of Government, Harvard University, Cambridge, MA, US; An earlier version of this paper was presented at the aforementioned seminar*. Cambridge University Press, 2007.

[24] W. A. Knight and T. Narozhna, "Social contagion and the female face of terror: New trends in the culture of political violence," *Canadian Foreign Policy Journal*, vol. 12, no. 1, pp. 141–166, 2005.

[25] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2003, pp. 137–146.

[26] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 199–208.

[27] A. Goyal, W. Lu, and L. V. Lakshmanan, "Celf++ optimizing the greedy algorithm for influence maximization in social networks," in *Proceedings of the 20th international conference companion on World wide web*, 2011, pp. 47–48.

[28] Y. Tang, X. Xiao, and Y. Shi, "Influence maximization: Near-optimal time complexity meets practical efficiency," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, 2014, pp. 75–86.

[29] N. Du, Y. Liang, M. Balcan, and L. Song, "Influence function learning in information diffusion networks," in *International Conference on Machine Learning*. PMLR, 2014, pp. 2016–2024.

[30] X. He, K. Xu, D. Kempe, and Y. Liu, "Learning influence functions from incomplete observations," *arXiv preprint arXiv:1611.02305*, 2016.

[31] A. Goyal, F. Bonchi, and L. V. Lakshmanan, "Learning influence probabilities in social networks," in *Proceedings of the third ACM international conference on Web search and data mining*, 2010, pp. 241–250.

[32] D. Kalimeris, Y. Singer, K. Subbian, and U. Weinsberg, "Learning diffusion using hyperparameters," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2420–2428.

[33] M. Gomez-Rodriguez, L. Song, N. Du, H. Zha, and B. Schölkopf, "Influence estimation and maximization in continuous-time diffusion networks," *ACM Transactions on Information Systems (TOIS)*, vol. 34, no. 2, pp. 1–33, 2016.

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

[35] J. Ko, K. Lee, K. Shin, and N. Park, "Monstor: an inductive approach for estimating and maximizing influence over unseen networks," in *2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2020, pp. 204–211.

[36] G. Tong, "Stratlearner: Learning a strategy for misinformation prevention in social networks," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[37] N. Du, L. Song, M. Gomez Rodriguez, and H. Zha, "Scalable influence estimation in continuous-time diffusion networks," *Advances in neural information processing systems*, vol. 26, 2013.

[38] G. Tong, R. Wang, and Z. Dong, "On multi-cascade influence maximization: Model, hardness and algorithmic framework," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1600–1613, 2021.

[39] J. G. Oxley, *Matroid theory*. Oxford University Press, USA, 2006, vol. 3.

[40] G. Tong, "Usco-solver: Solving undetermined stochastic combinatorial optimization problems," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[41] I. Tsochantaridis, T. Joachims, T. Hofmann, Y. Altun, and Y. Singer, "Large margin methods for structured and interdependent output variables." *Journal of machine learning research*, vol. 6, no. 9, 2005.

[42] A. Rahimi and B. Recht, "Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning," *Advances in neural information processing systems*, vol. 21, 2008.

[43] D. McAllester, "Simplified pac-bayesian margin bounds," in *Learning theory and Kernel machines*. Springer, 2003, pp. 203–215.

[44] Y. Wu, M. Lan, S. Sun, Q. Zhang, and X. Huang, "A learning error analysis for structured prediction with approximate inference," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 6131–6141.

[45] B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin, "Learning structured prediction models: A large margin approach," in *Proceedings of the 22nd international conference on Machine learning*, 2005, pp. 896–903.

[46] A. C. Müller and S. Behnke, "Pystruct: learning structured prediction in python." *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 2055–2060, 2014.

[47] J. E. Kelley, Jr, "The cutting-plane method for solving convex programs," *Journal of the society for Industrial and Applied Mathematics*, vol. 8, no. 4, pp. 703–712, 1960.

[48] A. Lucchi, Y. Li, and P. Fua, "Learning for structured prediction using approximate subgradient descent with working sets," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1987–1994.

[49] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani, "Kronecker graphs: an approach to modeling networks." *Journal of Machine Learning Research*, vol. 11, no. 2, 2010.

[50] A. Hagberg, P. Swart, and D. S Chult, "Exploring network structure, dynamics, and function using networkx," Los Alamos National Lab.(LANL), Los Alamos, NM (United States), Tech. Rep., 2008.

[51] M. De Domenico, A. Lima, P. Mougel, and M. Musolesi, "The anatomy of a scientific rumor," *Scientific reports*, vol. 3, no. 1, pp. 1–9, 2013.

[52] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graph evolution: Densification and shrinking diameters," *ACM transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, pp. 2–es, 2007.

[53] I. Rish *et al.*, "An empirical study of the naive bayes classifier," in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, no. 22, 2001, pp. 41–46.

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

[55] Y. Zhang, J. Hare, and A. Prugel-Bennett, "Deep set prediction networks," in *NeurIPS*, 2019, pp. 3207–3217.

[56] T. Lappas, E. Terzi, D. Gunopulos, and H. Mannila, "Finding effectors in social networks," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010, pp. 1059–1068.

[57] M. Ramezani, A. Khodadadi, and H. R. Rabiee, "Community detection using diffusion information," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 12, no. 2, pp. 1–22, 2018.

[58] D. P. Bertsekas and J. N. Tsitsiklis, "An analysis of stochastic shortest path problems," *Mathematics of Operations Research*, vol. 16, no. 3, pp. 580–595, 1991.

[59] A. Gupta and M. Pál, "Stochastic steiner trees without a root," in *International Colloquium on Automata, Languages, and Programming*. Springer, 2005, pp. 1051–1063.

[60] R. Ramanathan, "Stochastic decision making using multiplicative ahp," *European Journal of Operational Research*, vol. 97, no. 3, pp. 543–549, 1997.

[61] E. Celik, M. Gul, M. Yucesan, and S. Mete, "Stochastic multi-criteria decision-making: an overview to methods and applications," *Beni-Suef University Journal of Basic and Applied Sciences*, vol. 8, no. 1, pp. 1–11, 2019.

[62] S. Fujishige, *Submodular functions and optimization*. Elsevier, 2005.

[63] G. Calinescu, C. Chekuri, M. Pal, and J. Vondrák, "Maximizing a monotone submodular function subject to a matroid constraint," *SIAM Journal on Computing*, vol. 40, no. 6, pp. 1740–1766, 2011.

[64] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions—i," *Mathematical programming*, vol. 14, no. 1, pp. 265–294, 1978.

[65] C. Seshadhri, A. Pinar, and T. G. Kolda, "An in-depth study of stochastic kronecker graphs," in *2011 IEEE 11th International Conference on Data Mining*. IEEE, 2011, pp. 587–596.

[66] E. Bodine-Baron, B. Hassibi, and A. Wierman, "Distance-dependent kronecker graphs for modeling social networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 4, pp. 718–731, 2010.

[67] C. Seshadhri, T. G. Kolda, and A. Pinar, "Community structure and scale-free collections of erdős-rényi graphs," *Physical Review E*, vol. 85, no. 5, p. 056109, 2012.

[68] T. Hong and Q. Liu, "Seeds selection for spreading in a weighted cascade model," *Physica A: Statistical Mechanics and its Applications*, vol. 526, p. 120943, 2019.

[69] T. Joachims, T. Finley, and C.-N. J. Yu, "Cutting-plane training of structural svms," *Machine learning*, vol. 77, no. 1, pp. 27–59, 2009.

[70] Q. E. Dawkins, T. Li, and H. Xu, "Diffusion source identification on networks with statistical confidence," in *International Conference on Machine Learning*. PMLR, 2021, pp. 2500–2509.

[71] W. Dong, W. Zhang, and C. W. Tan, "Rooting out the rumor culprit from suspects," in *2013 IEEE International Symposium on Information Theory*. IEEE, 2013, pp. 2671–2675.

[72] D. Shah and T. Zaman, "Rumors in a network: Who's the culprit?" *IEEE Transactions on information theory*, vol. 57, no. 8, pp. 5163–5181, 2011.

[73] P.-D. Yu, C. W. Tan, and H.-L. Fu, "Rumor source detection in finite graphs with boundary effects by message-passing algorithms," in *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. Springer, 2018, pp. 175–192.

[74] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*. SIAM, 2014, pp. 946–957.

[75] X. Fang, P. J.-H. Hu, Z. Li, and W. Tsai, "Predicting adoption probabilities in social networks," *Information Systems Research*, vol. 24, no. 1, pp. 128–145, 2013.

[76] K. Saito, M. Kimura, K. Ohara, and H. Motoda, "Selecting information diffusion models over social networks for behavioral analysis," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2010, pp. 180–195.

[77] F. Bonchi, "Influence propagation in social networks: A data mining perspective." *IEEE Intell. Informatics Bull.*, vol. 12, no. 1, pp. 8–16, 2011.

[78] H. Wang, H. Xie, L. Qiu, Y. R. Yang, Y. Zhang, and A. Greenberg, "Cope: Traffic engineering in dynamic networks," in *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, 2006, pp. 99–110.

[79] F. Fang, T. H. Nguyen, R. Pickles, W. Y. Lam, G. R. Clements, B. An, A. Singh, M. Tambe, and A. Lemieux, "Deploying paws: Field optimization of the protection assistant for wildlife security," in *Twenty-eighth IAAI conference*, 2016.

[80] A. Mukhopadhyay and Y. Vorobeychik, "Prioritized allocation of emergency responders based on a continuous-time incident prediction model," in *International Conference on Autonomous Agents and MultiAgent Systems*, 2017.

[81] Y. Xue, I. Davies, D. Fink, C. Wood, and C. P. Gomes, "Avicaching: A two stage game for bias reduction in citizen science," in *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, 2016, pp. 776–785.

[82] A. Beygelzimer and J. Langford, "The offset tree for learning with partial labels," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 129–138.

[83] B. Ford, T. Nguyen, M. Tambe, N. Sintov, and F. Delle Fave, "Beware the soothsayer: From attack prediction accuracy to predictive reliability in security games," in *International Conference on Decision and Game Theory for Security*. Springer, 2015, pp. 35–56.

[84] P. L. Donti, B. Amos, and J. Z. Kolter, "Task-based end-to-end model learning in stochastic optimization," *arXiv preprint arXiv:1703.04529*, 2017.

[85] B. Wilder, B. Dilkina, and M. Tambe, "Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 1658–1665.

[86] D. Bertsimas and N. Kallus, "From predictive to prescriptive analytics," *Management Science*, vol. 66, no. 3, pp. 1025–1044, 2020.

[87] Y. Bengio, "Using a financial training criterion rather than a prediction criterion," *International journal of neural systems*, vol. 8, no. 04, pp. 433–443, 1997.

[88] J. Roland, Y. De Smet, and J. R. Figueira, "Inverse multi-objective combinatorial optimization," *Discrete applied mathematics*, vol. 161, no. 16-17, pp. 2764–2771, 2013.

[89] J. Zhang and Z. Ma, "Solution structure of some inverse combinatorial optimization problems," *Journal of Combinatorial Optimization*, vol. 3, no. 1, pp. 127–139, 1999.

[90] T. C. Chan, R. Mahmood, and I. Y. Zhu, "Inverse optimization: Theory and applications," *arXiv preprint arXiv:2109.03920*, 2021.

[91] Y. Tan, D. Terekhov, and A. Delong, "Learning linear programs from optimal decisions," *Advances in Neural Information Processing Systems*, vol. 33, pp. 19 738–19 749, 2020.

[92] Z. Shahmoradi and T. Lee, "Quantile inverse optimization: Improving stability in inverse linear programming," *Operations Research*, 2021.

[93] F. Ahmadi, F. Ganjkhanloo, and K. Ghobadi, "Inverse learning: A data-driven framework to infer optimizations models," *arXiv preprint arXiv:2011.03038*, 2020.

[94] C. Dong and B. Zeng, "Wasserstein distributionally robust inverse multiobjective optimization," *arXiv preprint arXiv:2009.14552*, 2020.

[95] C. Dong, Y. Chen, and B. Zeng, "Generalized inverse optimization through online learning," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Sec. 5
   (c) Did you discuss any potential negative societal impacts of your work? [No] This work focuses on learning and algorthimic foundations of social contagion management, without being tied to specific applications
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] They can be found in the appendix.
3. If you ran 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)? [Yes] They are provided in the supplemental material. Due to the 100M limitation, we only provide partial data at the submission time and will release the complete dataset (> 6GB) after the review period.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] The details are provided in Appendix E.1.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Yes, we report standard deviations.
   (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 Appendix E.2.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] We use public dataset and source code following their licenses.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] Our dataset is purely numerical and does not have any personally identifiable information or offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Social-Inverse: Inverse Decision-making of Social Contagion Management with Task Migrations (Appendix)

A Diffusion process

An example of the diffusion process can be found in Figure 2.

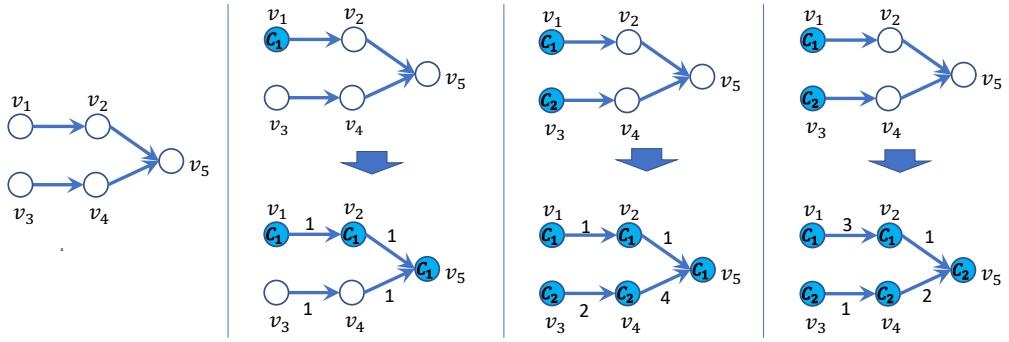


Figure 2: The first subgraph shows an example social network. The second subgraph presents a possible realization as well as a diffusion process of one cascade spreading from $\{v_1\}$. The third and fourth subgraphs depict a competitive diffusion under different realizations, where the cascades C_1 and C_2 start to spread from $\{v_1\}$ and $\{v_3\}$, respectively; node v_5 is activated by cascade C_1 at time 2 in the third subgraph, while it is activated by cascade C_2 at time 3 in the fourth subgraph.

B Notations

Table 2: Notations.

Symbol	Description	Reference
$G = (V, E)$	social graph	Sec. 2.1
N_u^-	the set of in-neighbors of node u	Sec. 2.1
$\mathcal{N}_u$	a distribution over $2^{N_u^-}$ for a node u	Sec. 2.1
$\mathcal{T}_{(u,v)}$	a distribution over $(0, +\infty]$ for a pair (u, v)	Sec. 2.1
$\mathcal{R}_G$	the set of all possible retaliations (i.e., weighted subgraphs)	Def. 1
$\mathcal{M} : \mathcal{R}_G \to [0, 1]$	the distribution over $\mathcal{R}_G$ specified by model $\mathcal{M}$	Def. 1
$f_{\mathcal{M}}^{\mathrm{DE}}$ and $f_{\mathcal{M}}^{\mathrm{DC}}$	the objective functions of DE and DC associated with model $\mathcal{M}$	Problems 1 and 2
$f_{\mathcal{M}}^{T}$	the objective function of an abstract task T associated model $\mathcal{M}$	Problem 3
$Y_{\mathcal{M},T,X}^{\alpha}$	an α-approximation to $\arg\max_{Y \in \mathcal{Y}_T} f_{\mathcal{M}}^{T}(X, Y)$	Problem 3
f_r^T	the objective function projected to a realization r	Def. 2
$S_{\mathcal{M},T,m}^{\alpha}$	dataset associated with task T and a ratio α	Equation 5
$T_{ⓢ}$ and $T_{ⓣ}$	source task and target task	Problem 4
$\mathcal{M}_{true}$	the true model that defines the optimization tasks	Problem 4
$\overline{\alpha}_{ⓢ}$	approximation ratio of the samples regarding the source task	Problem 4
$\mathcal{M}_{em}$	an empirical distribution over $\mathcal{R}_G$	Sec. 3
$\mathcal{F}_{R_K}$	the hypothesis space defined over a set R_K of K realizations	Sec. 3
$H_{R_K,\mathbf{w}}^{ⓣ}(X, Y)$	the hypothesis associated with vector $\mathbf{w}$ and R_K	Sec. 3
$\widetilde{\mathbf{w}}$ and $\overline{\mathbf{w}}$	the prior weight and final weight	Sec. 3
γ	the parameter controlling the distribution for sampling $\overline{\mathbf{w}}$	Sec. 3
$Y_{R_K,\overline{\mathbf{w}},X}^{\widetilde{\alpha}_{ⓣ}}$	an $\widetilde{\alpha}_{ⓣ}$-approximation for inference	Equation 8
$\mathrm{I}_{R_K,\widetilde{\mathbf{w}},X,\beta}^{ⓣ}$	the margin set specifying potential predictions associated with X	Equation 10
β	the parameter that controls the strictness of margin	Equation 10
$H_{R_K,\mathbf{w}}^{ⓢ}(X, Y)$	the score function associated with the source task	Equation 17

C An illustration of Social-Inverse

In what follows, we present concrete steps in Social-Inverse for solving Problem 4, using an example with $T_{\textcircled{s}} = \text{DE}$, $T_{\textcircled{t}} = \text{DC}$, $K = 30$, and $M_{em} = \mathcal{M}_\infty$, where $\mathcal{M}_\infty$ samples random subgraphs with unit weights. In such a case, the best ratio one can have is $\overline{\alpha}_{\textcircled{s}} = \widetilde{\alpha}_{\textcircled{t}} = 1 - 1/e \approx 0.618$.

- **Instance setting.** A social graph $G = (V, E)$ and an underlying diffusion model $\mathcal{M}_{true}$.
- **Input.** A collection of pairs $S_{\text{DE}}^{\overline{\alpha}_{\textcircled{s}}} = \left\{(X_i, Y_{\mathcal{M}_{true},\text{DE},X_i}^{\overline{\alpha}_{\textcircled{s}}})\right\}_{i=1}^m$, where X_i is a subset of V and $Y_{\mathcal{M}_{true},\text{DE},X_i}^{\overline{\alpha}_{\textcircled{s}}}$ is a $(1-1/e)$-approximation to $\arg\max_{Y \subseteq V, |Y| \leq k} f_{\mathcal{M}_{true}}^{\text{DE}}(X_i, Y)$ for some predefined budget $k \in \mathbb{Z}$.
- **Step 1.** Sample a collection of $K = 30$ iid realizations $R_K = \{r_1, ..., r_{30}\} \subseteq \mathcal{R}_G$ following $\mathcal{M}_\infty$.
- **Step 2.** With $S_{\text{DE}}^{\overline{\alpha}_{\textcircled{s}}}$ and R_K, calculate a prior vector $\widetilde{\mathbf{w}}$ by solving the following program using, for example, the n-slack cutting plane algorithm [41]:

$$\begin{aligned} \min \quad & \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i / m \\ \text{s.t.} \quad & \widetilde{\alpha}_{\textcircled{t}}(\widetilde{\alpha}_{\textcircled{t}} - \beta) \cdot H_{R_K,\mathbf{w}}^{\textcircled{s}}(X_i, Y_{\mathcal{M}_{true},\text{DE},X_i}^{\overline{\alpha}_{\textcircled{s}}}) - H_{R_K,\mathbf{w}}^{\textcircled{s}}(X_i, Y) \geq \xi_i, \ \forall i \in [m], \ \forall Y \in \mathcal{Y}_{\textcircled{s}} \\ & \mathbf{w} \geq 0 \end{aligned}$$

  In solving the above program, we need to repeatedly deal with some intermediate weight $\mathbf{w} = (w_1, ..., w_{30})$ and solve

$$\arg\max_{Y \subseteq V, |Y| \leq \left|Y_{\mathcal{M}_{true},\text{DE},X_i}^{\overline{\alpha}_{\textcircled{s}}}\right|} H_{R_K,\mathbf{w}}^{\textcircled{s}}(X_i, Y) = \sum_{i=1}^{30} w_i \cdot f_{r_i}^{\text{DE}}(X_i, Y),$$

  which can be approximated within in a ratio of $1 - 1/e$ (Theorem 3).
- **Step 3.** Acquire γ through Equation 12, and sample the final vector $\overline{\mathbf{w}}$ following $Q(\gamma \cdot \widetilde{\mathbf{w}}, \mathcal{I})$. Notice that this step is optional due to the facts that a) every entry of $\widetilde{\mathbf{w}}$ is scaled by the same factor and b) the prediction is invariant to scaling $\widetilde{\mathbf{w}}$.
- **Step 4.** With the final vector $\overline{\mathbf{w}} = (\overline{w}_1, ..., \overline{w}_{30})$ and R_K, given a new query $X^* \subseteq V$ of the target task DC and a budget k, make a prediction by solving the following problem using the greedy algorithm (Theorem 3):

$$\arg\max_{Y \subseteq V, |Y| \leq k} H_{R_K,\overline{\mathbf{w}}}^{\textcircled{t}}(X^*, Y) = \sum_{i=1}^{K} \overline{w}_i \cdot f_{r_i}^{\text{DC}}(X^*, Y).$$

D Proofs

D.1 Proof of Theorem 1

The proof of Theorem 1 is derived based on the following two results. The first lemma is the standard PAC-Bayesian bound for structured prediction [43]. The second lemma (proved in Sec. D.1.1) states that with a high chance, the prediction based on the sampled vector $\overline{\mathbf{w}}$ will fall into the margin defined by the prior vector $\widetilde{\mathbf{w}}$.

Lemma 1. *For each R_K, let $\mathcal{P}$ be a prior distribution over the hypothesis space (i.e., $\mathbb{R}^K$) and let $\delta \in (0, 1)$. Then, with probability of at least $1 - \delta$ over the choice of an iid training set $S = \{z_1, ..., z_m\}$ sampled according to $\mathcal{D}$, for all distributions $\mathcal{Q}$ over $\mathbb{R}^K$, we have*

$$\mathbb{E}_{\overline{\mathbf{w}} \sim \mathcal{Q}, X \sim \mathcal{D}} \left[l(X, Y_{R_K,\overline{\mathbf{w}},X}^{\widetilde{\alpha}_{\textcircled{t}}}) \right] \leq \frac{1}{m} \sum_{i=1}^m \mathbb{E}_{\overline{\mathbf{w}} \sim \mathcal{Q}} \left[l(X_i, Y_{R_K,\overline{\mathbf{w}},X}^{\widetilde{\alpha}_{\textcircled{t}}}) \right] + \sqrt{\frac{D(\mathcal{Q} \,||\, \mathcal{P}) + \ln(m/\delta)}{2(m-1)}}$$

where $D(\mathcal{Q} \,||\, \mathcal{P})$ is the KL divergence.

Lemma 2. *For each $\widetilde{\mathbf{w}}$, R_K and ϕ_{true}, with probability at least $1-\frac{\|\widetilde{\mathbf{w}}\|^2}{m}$ over the selection of $\overline{\mathbf{w}}$, we have*

$$l(X, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\overline{\mathbf{w}},X}) \leq \max_{Y\in\mathcal{Y}_{\textcircled{1}}} l(X,Y)\cdot \mathbb{1}_{\mathrm{I}^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}},X,\beta}}(Y) \tag{19}$$

holding simultaneously for all $X\in\mathcal{X}_{\textcircled{1}}$, provided that $\gamma = \frac{\widetilde{\alpha}^2_{\textcircled{1}}+1}{\min_p|\widetilde{\mathrm{w}}_p|\cdot\beta\cdot\widetilde{\alpha}_{\textcircled{1}}}\sqrt{2\ln\frac{2mK}{\|\widetilde{\mathbf{w}}\|^2}}$.

With the above two lemmas, we now prove Theorem 1. Let the prior distribution $\mathcal{P}$ in Lemma 1 be the isotropic Gaussian with an identity covariance and a zero mean, and $\mathcal{Q}$ be $Q(\gamma\cdot\widetilde{\mathbf{w}},\mathcal{I})$. We have the following inequality,

$$\mathbb{E}_{\overline{\mathbf{w}}\sim Q, X\sim\mathcal{D}}\left[l(X, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\overline{\mathbf{w}},X}\right] \leq \frac{1}{m}\sum_{i=1}^{m}\mathbb{E}_{\overline{\mathbf{w}}\sim Q}\left[l(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{\overline{\mathbf{w}},R_K,X_i})\right] + \sqrt{\frac{\frac{\gamma^2\|\widetilde{\mathbf{w}}\|^2}{2}+\ln\frac{m}{\delta}}{2(m-1)}}. \tag{20}$$

According to Lemma 2, for each X_i, we have

$$\begin{aligned}
&\mathbb{E}_{\overline{\mathbf{w}}\sim Q}\left[l(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{\overline{\mathbf{w}},R_K,X_i})\right]\\
\leq&(1-\frac{\|\widetilde{\mathbf{w}}\|^2}{m})\max_{Y\in\mathcal{Y}_{\textcircled{1}}} l(X_i,Y)\cdot\mathbb{1}_{\mathrm{I}^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}},X_i,\beta}}(Y) + \frac{\|\widetilde{\mathbf{w}}\|^2}{m}\max_{Y\in\mathcal{Y}_{\textcircled{1}}} l(X_i,Y)\\
\leq&\max_{Y\in\mathcal{Y}} l(X_i,Y)\cdot\mathbb{1}_{\mathrm{I}^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}},X_i,\beta}}(Y) + \frac{\|\widetilde{\mathbf{w}}\|^2}{m}.
\end{aligned} \tag{21}$$

Combining Equations 20 and 21, we have

$$\begin{aligned}
&\mathbb{E}_{\overline{\mathbf{w}}\sim Q, X\sim\mathcal{D}_{\mathcal{X}}}\left[l(X, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\overline{\mathbf{w}},X}\right]\\
\leq&\frac{1}{m}\sum_{i=1}^{m}\max_{Y\in\mathcal{Y}_{\textcircled{1}}} l(X_i,Y)\cdot\mathbb{1}_{\mathrm{I}^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}},X_i,\beta}}(Y) + \frac{\|\widetilde{\mathbf{w}}\|^2}{m} + \sqrt{\frac{\frac{\gamma^2\|\widetilde{\mathbf{w}}\|^2}{2}+\ln\frac{m}{\delta}}{2(m-1)}}.
\end{aligned}$$

D.1.1 Proof of Lemma 2

Let $\widetilde{\mathbf{w}} = (\widetilde{\mathrm{w}}_1, ..., \widetilde{\mathrm{w}}_K)$, $\overline{\mathbf{w}} = (\overline{\mathrm{w}}_1, ..., \overline{\mathrm{w}}_K)$ and $\epsilon := \frac{\beta\cdot\widetilde{\alpha}_{\textcircled{1}}}{\widetilde{\alpha}^2_{\textcircled{1}}+1}$. We first prove the following result:

$$|\overline{w}_p - \gamma\cdot\widetilde{w}_p| \leq \epsilon\cdot\gamma\cdot\widetilde{w}_p, \forall p\in\{1,...,K\} \implies Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\overline{\mathbf{w}},X} \in \mathrm{I}^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}},X,\beta} \implies \text{Equation 19.} \tag{22}$$

The second $\implies$ holds by definition, so it is left to prove the first part. With the LHS of Equation22, by rearrangements, we have

$$\begin{aligned}
&\widetilde{\alpha}_{\textcircled{1}}\cdot H^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X_i}) - H^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\overline{\mathbf{w}},X_i})\\
=&\frac{1}{\gamma}\Big(\widetilde{\alpha}_{\textcircled{1}}\cdot H^{\textcircled{1}}_{R_K,\gamma\cdot\widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X_i}) - H^{\textcircled{1}}_{R_K,\gamma\cdot\widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\overline{\mathbf{w}},X_i})\Big)\\
=&\frac{1}{\gamma}\Big(\widetilde{\alpha}_{\textcircled{1}}\cdot H^{\textcircled{1}}_{R_K,\gamma\cdot\widetilde{\mathbf{w}}-\overline{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X_i}) + \widetilde{\alpha}_{\textcircled{1}}\cdot H^{\textcircled{1}}_{R_K,\overline{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X_i})\\
&\qquad - H^{\textcircled{1}}_{R_K,\gamma\cdot\widetilde{\mathbf{w}}-\overline{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\overline{\mathbf{w}},X_i}) - H^{\textcircled{1}}_{R_K,\overline{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\overline{\mathbf{w}},X_i})\Big)\\
&\{\text{by the LHS of Equation22}\}\\
=&\frac{1}{\gamma}\Big(\widetilde{\alpha}_{\textcircled{1}}\cdot H^{\textcircled{1}}_{R_K,\epsilon\cdot\gamma\cdot\widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X_i}) + \widetilde{\alpha}_{\textcircled{1}}\cdot H^{\textcircled{1}}_{R_K,\overline{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X_i})\\
&\qquad + H^{\textcircled{1}}_{R_K,\epsilon\cdot\gamma\cdot\widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\overline{\mathbf{w}},X_i}) - H^{\textcircled{1}}_{R_K,\overline{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\overline{\mathbf{w}},X_i})\Big).
\end{aligned}$$

Since $H^{\textcircled{1}}_{R_K,\overline{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\overline{\mathbf{w}},X_i}) \geq \widetilde{\alpha}_{\textcircled{1}}\max_{Y\in\mathcal{Y}} H^{\textcircled{1}}_{R_K,\overline{\mathbf{w}}}(X_i,Y) \geq \widetilde{\alpha}_{\textcircled{1}}\cdot H^{\textcircled{1}}_{R_K,\overline{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X_i})$, the above inequality is further upper bounded by

$$\frac{1}{\gamma}\Big(\widetilde{\alpha}_{\textcircled{1}}\cdot H^{\textcircled{1}}_{R_K,\epsilon\cdot\gamma\cdot\widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X_i}) + H^{\textcircled{1}}_{R_K,\epsilon\cdot\gamma\cdot\widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\overline{\mathbf{w}},X_i})\Big),$$

which equals to

$$\widetilde{\alpha}_{\textcircled{1}} \cdot H^{\textcircled{1}}_{R_K,\epsilon\cdot\widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X_i}) + H^{\textcircled{1}}_{R_K,\epsilon\cdot\widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\overline{\mathbf{w}},X_i}). \tag{23}$$

By the definition of $Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X_i}$, Equation 23 is no more than

$$\begin{aligned}&\widetilde{\alpha}_{\textcircled{1}} \cdot H^{\textcircled{1}}_{R_K,\epsilon\cdot\widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X_i}) + \frac{1}{\widetilde{\alpha}_{\textcircled{1}}} \cdot H^{\textcircled{1}}_{R_K,\epsilon\cdot\widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X_i}) \\ &= (\epsilon \cdot \widetilde{\alpha}_{\textcircled{1}} + \frac{\epsilon}{\widetilde{\alpha}_{\textcircled{1}}}) \cdot H^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X_i})\end{aligned} \tag{24}$$

By the selection of ϵ, the RHS of Equation 24 is upper bounded by $\beta \cdot H^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}}}(X_i, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X_i})$. which implies $Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\overline{\mathbf{w}},X} \in \mathrm{I}^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}},X,\beta}$ by the definition of $\mathrm{I}^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}},X,\beta}$ (Equation 10). Therefore, Equation 22 is proved.

With Equation 22, to prove Lemma 2, it suffices to prove

$$\Pr_{\widetilde{\mathbf{w}}\sim Q}\left[|\overline{w}_p - \gamma \cdot \widetilde{w}_p| \leq \epsilon \cdot \gamma \cdot \widetilde{w}_p, \forall p \in [K]\right] \geq 1 - \frac{\|\widetilde{\mathbf{w}}\|^2}{m} \tag{25}$$

By the property of Gaussian, we have

$$\Pr\left[|\overline{w}_p - \gamma \cdot \widetilde{w}_p| \geq \epsilon \cdot \gamma \cdot \widetilde{w}_p\right] \leq 2\exp(\frac{-(\gamma \cdot \widetilde{w}_p \cdot \epsilon)^2}{2}), \tag{26}$$

which is no more than $\frac{\|\widetilde{\mathbf{w}}\|^2}{K\cdot m}$ by the selection of ϵ and γ. This implies Equation 25 due to the union bound over $p \in [K]$.

D.2 Proof of Theorem 2

The next lemma (proved in Sec. D.2.1) relates the margin (Equation 10) with the latent objective function (i.e., $f^{T_{\textcircled{1}}}_{\mathcal{M}_{true}}$) by stating that our choice of K ensures that the hypothesis space contains a score function that can approximate the latent objective function to a sufficient extent.

Lemma 3. *For each $\epsilon, \delta_1, \delta_2 \in \mathbb{R}$ and $\mathcal{M}_{em}$, when $K = \frac{4\Delta^2}{\epsilon^2\cdot\delta_2^2}(\ln|\mathcal{Y}_{T_{\textcircled{1}}}| + \ln\frac{1}{\delta_1} + 1)$, with probability at least $1-\delta_1$ over the selection of R_K, there exist a $\widetilde{\mathbf{w}}$ such that we have*

$$\Pr_{X\sim\mathcal{D}}\left[\left|H^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}}}(X,Y) - f^{T_{\textcircled{1}}}_{\mathcal{M}_{true}}(X,Y)\right| \leq \epsilon \cdot f^{T_{\textcircled{1}}}_{\mathcal{M}_{true}}(X,Y), \forall\, Y \in \mathcal{Y}\right] \geq 1-\delta_2. \tag{27}$$

To prove Theorem 2, it suffices to prove that the event in Equation 27 implies the event in Equation 15. With the event in Equation 27, we have for each $Y \in \mathrm{I}^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}},X,\beta}$ that

$$\begin{aligned}&\widetilde{\alpha}_{\textcircled{1}} \cdot H^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}}}(X, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X}) - H^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}}}(X,Y) \leq \beta \cdot H^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}}}(X, Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X}) \\ &\{\text{by the definition of } Y^{\widetilde{\alpha}_{\textcircled{1}}}_{R_K,\widetilde{\mathbf{w}},X}\} \\ \Longrightarrow\ &\widetilde{\alpha}_{\textcircled{1}} \cdot (\widetilde{\alpha}_{\textcircled{1}} - \beta) \cdot H^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}}}\big(X, \arg\max_{Y\in\mathcal{Y}_{\textcircled{1}}} f^{T_{\textcircled{1}}}_{\mathcal{M}_{true}}(X,Y)\big) - H^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}}}(X,Y) \leq 0 \\ &\{\text{by the event in Equation 27}\} \\ \Longrightarrow\ &\widetilde{\alpha}_{\textcircled{1}} \cdot (\widetilde{\alpha}_{\textcircled{1}} - \beta)(1-\epsilon) \cdot f^{T_{\textcircled{1}}}_{\mathcal{M}_{true}}\big(X, \arg\max_{Y\in\mathcal{Y}_{\textcircled{1}}} f^{T_{\textcircled{1}}}_{\mathcal{M}_{true}}(X,Y)\big) - (1+\epsilon) \cdot f^{T_{\textcircled{1}}}_{\mathcal{M}_{true}}(X,Y) \leq 0 \\ \Longrightarrow\ &\widetilde{\alpha}_{\textcircled{1}} \cdot (\widetilde{\alpha}_{\textcircled{1}} - \beta)(1-\epsilon) \cdot \max_{Y\in\mathcal{Y}_{\textcircled{1}}} f^{T_{\textcircled{1}}}_{\mathcal{M}_{true}}\big(X,Y\big) - (1+\epsilon) \cdot f^{T_{\textcircled{1}}}_{\mathcal{M}_{true}}(X,Y) \leq 0.\end{aligned}$$

This implies the desired results as follows:

$$\max_{Y\in\mathcal{Y}_{\textcircled{1}}} l_{\text{approx}}(X,Y) \cdot \mathbb{1}_{\mathrm{I}^{\textcircled{1}}_{R_K,\widetilde{\mathbf{w}},X,\beta}}(Y) \leq 1 - \frac{\widetilde{\alpha}_{\textcircled{1}} \cdot (\widetilde{\alpha}_{\textcircled{1}} - \beta) \cdot (1-\epsilon)}{(1+\epsilon)}$$

D.2.1 Proof of Lemma 3

The proof basically concerns about the concentration of averages towards the mean in a Hilbert space, which is similar to the proof of Lemma 4 in [42]. For completeness, we give the proof below.

For each $X \in \mathcal{X}_{\textcircled{1}}$ and $Y \in \mathcal{Y}_{\textcircled{1}}$, we have the observation that

$$\mathbb{E}_{r\sim\mathcal{M}_{em}}\left[\frac{M_{true}(r)}{\mathcal{M}_{em}(r)} f_r^{T_{\textcircled{1}}}(X,Y)\right] = f_{\mathcal{M}_{true}}^{T_{\textcircled{1}}}(X,Y) \tag{28}$$

Noticing that $R_K = \{r_1, ..., r_K\}$ are iid samples from $\mathcal{M}_{em}$, we consider its normalized average function

$$\hat{h}_{(X,Y)}(r_1, ..., r_K) = \frac{1}{f_{\mathcal{M}_{true}}^{T_{\textcircled{1}}}(X,Y)} \sum_{i=1}^{K} \frac{\mathcal{M}_{true}(r_i)}{K \cdot \mathcal{M}_{em}(r_i)} f_{r_i}^{T_{\textcircled{1}}}(X,Y).$$

For each $Y \in \mathcal{Y}_{\textcircled{1}}$, we denote by $g_Y(r_1, ..., r_K)$ the L2 norm of $|\hat{h}_{(X,Y)}(r_1, ..., r_K) - 1|$ with respect to $X \sim \mathcal{D}$:

$$g_Y(r_1, ..., r_K) = \sqrt{\int_{X\in\mathcal{X}_{\textcircled{1}}} \left(\hat{h}_{(X,Y)}(r_1, ..., r_K) - 1\right)^2 d\,\mathcal{D}(X)}. \tag{29}$$

Taking $g_Y(r_1, ..., r_K)$ as a function of the variables r_i, its stability is given by the fact that for each $\{r_1, ..., r_K, r_i^*\} \subseteq \mathcal{R}_G$, we have

$$\left|g_Y(r_1, ..., r_i, ..., r_K) - g_Y(r_1, ..., r_i, ..., r_K)\right| \leq \sup_r \frac{\mathcal{M}_{true}(r)}{\mathcal{M}_{em}(r)} \frac{2\max f_r^{T_{\textcircled{1}}}(X,Y)}{K \min f_r^{T_{\textcircled{1}}}(X,Y)} = 2\Delta/K.$$

In addition, an upper bound of the expectation of $g_Y(r_1, ..., r_K)$ is found by

$$\begin{aligned}
&\mathbb{E}_{R_K\sim\mathcal{M}_{em}}[g_y(r_1, ..., r_K)] \leq \sqrt{\mathbb{E}_{R_K\sim\mathcal{M}_{em}}[g_y(r_1, ..., r_K)^2]} \\
=&\sqrt{\int_{X\in\mathcal{X}_{\textcircled{1}}} \mathbb{E}_{R_K\sim\mathcal{M}_{em}}[\left(\hat{h}_{(X,Y)}(r_1, ..., r_K) - 1\right)^2] d\,\mathcal{D}(X)} \\
=&\sqrt{\int_{X\in\mathcal{X}_{\textcircled{1}}} \frac{1}{K}\Big(\mathbb{E}_{r\sim\mathcal{M}_{em}}[\Big(\frac{1}{f_{\mathcal{M}_{true}}^{T_{\textcircled{1}}}(X,Y)} \frac{\mathcal{M}_{true}(r)}{\mathcal{M}_{em}(r)} f_r^{T_{\textcircled{1}}}(X,Y)\Big)^2] - 1\Big) d\,\mathcal{D}(X)} \\
\leq&\sqrt{\int_{X\in\mathcal{X}_{\textcircled{1}}} \frac{1}{K}\Big(\sup_r \frac{\mathcal{M}_{true}(r)}{\mathcal{M}_{em}(r)} \frac{\max f_r^{T_{\textcircled{1}}}(X,Y)}{\min f_r^{T_{\textcircled{1}}}(X,Y)}\Big)^2 d\,\mathcal{D}(X)} = \frac{\Delta}{\sqrt{K}}
\end{aligned} \tag{30}$$

With the stability result and a bounded expectation, we have the following results:

$$\begin{aligned}
&\Pr_{R_K\sim\mathcal{M}_{em}}[g_Y(r_1, ..., r_K) - \frac{\epsilon\cdot\delta_2}{2} \geq \frac{\epsilon\cdot\delta_2}{2}] \\
\leq&\Pr_{R_K\sim\mathcal{M}_{em}}[g_Y(r_1, ..., r_K) - \frac{\Delta}{\sqrt{K}} \geq \frac{\epsilon\cdot\delta_2}{2}] \\
\leq&\Pr_{R_K\sim\mathcal{M}_{em}}[g_Y(r_1, ..., r_K) - \mathbb{E}_{R_K\sim\mathcal{M}_{em}}[g_y(r_1, ..., r_K)] \geq \frac{\epsilon\cdot\delta_2}{2}] \\
\leq&\exp(\frac{-2(\frac{\epsilon\cdot\delta_2}{2})^2}{K(2\Delta/K)^2}) = \delta_1/|\,\mathcal{Y}_{\textcircled{1}}\,|,
\end{aligned}$$

where the first and the third inequalities follow from selection the selection of K and the second inequality follows from Equation 30. Taking the union bound over $Y \in \mathcal{Y}_{\textcircled{1}}$, we have the followings holding simultaneously for each $Y \in \mathcal{Y}_{\textcircled{1}}$ with probability at least $1 - \delta_1$,

$$\begin{aligned}
&\mathbb{E}_{X\sim\mathcal{D}}\left[|\hat{h}_{(X,Y)}(r_1, ..., r_K) - 1|\right] \\
\leq&\sqrt{\mathbb{E}_{X\sim\mathcal{D}}\left[|\hat{h}_{(X,Y)}(r_1, ..., r_K) - 1|^2\right]} = g_Y(r_1, ..., r_K) \leq \epsilon\cdot\delta_2
\end{aligned}$$

By the Markov's inequality, we have

$$\Pr_{X\sim\mathcal{D}}\left[|\hat{h}_{(X,Y)}(r_1,...,r_K)-1|\leq\epsilon,\forall\, Y\in\mathcal{Y}_{\textcircled{t}}\right]\leq\frac{\epsilon\cdot\delta_2}{\epsilon}\implies$$

$$\Pr_{X\sim\mathcal{D}}\left[|\sum_{i=1}^{K}\frac{\mathcal{M}_{true}(r_i)}{K\cdot\mathcal{M}_{em}(r_i)}f_{r_i}^{T_{\textcircled{t}}}(X,Y)-f_{\mathcal{M}_{true}}^{T_{\textcircled{t}}}(X,Y)|\leq\epsilon f_{\mathcal{M}_{true}}^{T_{\textcircled{t}}}(X,Y),\forall\, Y\in\mathcal{Y}_{\textcircled{t}}\right]\leq\delta_2,$$

which has suggested that $(\frac{\mathcal{M}_{true}(r_1)}{K\cdot\mathcal{M}_{em}(r_1)},...,\frac{\mathcal{M}_{true}(r_K)}{K\cdot\mathcal{M}_{em}(r_K)})$ is the desired weight.

D.3 Proof of Theorem 3

A set function f over the subsets of a set U is submodular if it has a diminishing marginal return [62]:

$$f(\{v\}\cup U_1)-f(U_1)\geq f(\{v\}\cup U_2)-f(U_2),\ \forall\, U_1\subseteq U_2\subseteq\ U, v\notin U_2.$$

Maximizing a nonnegative monotone submodular function over a matroid can be approximated within a ratio of $1-1/e$ [62, 63]. It is well-known that for DE and DC, the kernel function is Np-hard to optimize in optimal over a general realization $r\in\mathcal{R}_G$, but it is submodular [25, 20]. Since submodularity is preserved under non-negative linear combination, $H_{R_K,\mathbf{w}}^{\textcircled{s}}(X_i,Y)$ and $H_{R_K,\mathbf{w}}^{\textcircled{t}}(X,Y)$ are both submodular for any $\mathbf{w}\geq 0$, thereby admitting $(1-1/e)$-approximations. In particular, when the matroid reduces to the cardinality constraint, the simple greedy algorithm provides an $(1-1/e)$-approximation [64].

Table 3: **Datasets.**

	Kronecker	Erdős-Rényi	Higgs	Hep
Nodes	1024	512	10000	15233
Edges	2655	650	22482	32235
Size of the pool of query-decision pairs	2700	2700	810	810
Size of the pool of realizations	10000	10000	1000	1000
Default training size	270	270	270	270
Default testing size	540	540	540	540
Graph structure source	[49]	[50]	[51]	[52]

E Empirical studies

In this section, we first provide details in experimental settings (Sec. E.1), and then present the full experimental results (Secs. E.2, E.3 and E.4). The data and source code can be found in the supplementary material, including the sample pools and pre-train models. Due to the 100M limit, we only include the realizations of Kronecker and Erdős-Rényi under $\mathcal{M}_\infty$. The complete dataset (>6GB) will be released after the review period.

E.1 Detailed experimental settings

E.1.1 The latent model $\mathcal{M}_{true}$ and samples

To set up a latent diffusion model $\mathcal{M}_{true}$, we first fix the graph structure and then generate the distributions $\mathcal{N}_u$ and $\mathcal{T}_{(u,v)}$.

- **Graphs.** We adopt four graphs: Kronecker, Erdős-Rényi, Higgs, and Hep. Kronecker and Erdős-Rényi are two representative graphs for studying social networks [65, 66, 67]. Higgs is a subgraph of the network built after monitoring the spreading process of the messages on Twitter regarding the discovery of the elusive Higgs boson [51]. Hep is a collaboration network of Arxiv High Energy Physics category. More information can be found in Table 3.
- **Distribution $\mathcal{N}_u$.** Following the independent cascade model [25], for each node u, the distribution $\mathcal{N}_u$ is designed by assuming that each in-neighbor v of u will be selected independently with a probability of $p_{(v,u)}$. In particular, following the weighted cascade model [68], we have $p^{true}_{(v,u)} = 1/|N^-_u|$ in $\mathcal{M}_{true}$.
- **Distribution $\mathcal{T}_{(u,v)}$.** For each edge e, the distribution $\mathcal{T}_{(u,v)}$ is selected to be a Weibull distribution with two parameters $\alpha_{(u,v)}$ and $\beta_{(u,v)}$. In the underlying model $\mathcal{M}_{true}$, $\alpha^{true}_{(u,v)}$ and $\beta^{true}_{(u,v)}$ are randomly selected from $\{1, ..., 10\}$, which is standard practice for having a diverse transmission time [29, 36].

With the latent diffusion model $\mathcal{M}_{true}$, the objective functions of DE and DC become concrete. There are four possible settings of Problem 4: DE-DE, DE-DC, DC-DE, and DC-DC.

Samples. Given the latent diffusion model $\mathcal{M}_{true}$, we generate random samples for DE and DC, as follows. In generating one query-decision pair $(X, Y^\alpha_{\mathcal{M}_{true},T,X})$ where T is DE (resp., DC), the size of X is sampled from a power law distribution with a parameter of 2.5 scaled by 40 (resp., 10), and the elements in X are then randomly selected from V; the size of $Y^\alpha_{\mathcal{M}_{true},T,X}$ is sampled from a power law distribution with a parameter of 2.5 scaled by 10, and the elements in $Y^\alpha_{\mathcal{M}_{true},T,X}$ are computed by solving Equation 3 using the greedy algorithm, which is a $(1 - 1/e)$-approximation (Theorem 3). For each of DE and DC, we generate a pool of samples for training and testing; the size of the pool can be found in Table 3.

Remark 4. For DE, the rationale behind the selection of the size of X is that X cannot be either too small or too large. When X is too small, selecting the neighbor of X would result in a perfect influence to X, which makes the problem trivial; when X is too large, solving the influence maximization problem over the entire graph would always produce a near-optimal solution, which again makes the problem trivial.

E.1.2 Social-Inverse

To set up Social-Inverse, with K and β being the hyperparameters, we are left to specify the empirical distribution $\mathcal{M}_{em}$ for generating realizations. We consider the realizations generated by the following four diffusion models.

- $\mathcal{M}_q$. $\mathcal{M}_q$ is obtained from $\mathcal{M}_{true}$ by using a different set of parameters (i.e., $p_{(u,v)}$, $\alpha_{(u,v)}$, and $\beta_{(u,v)}$ for each $(u,v) \in E$). In particular, for each $q \in \{0.1, 0.5, 1.0\}$, $p_{(u,v)}$, $\alpha_{(u,v)}$, and $\beta_{(u,v)}$ are sampled from uniform distributions, respectively, over
$$\left[p^{true}_{(u,v)} \cdot (1-q), q \cdot p^{true}_{(u,v)}(1+q)\right],$$
$$\left[\alpha^{true}_{(u,v)} \cdot (1-q), q \cdot \alpha^{true}_{(u,v)}(1+q)\right],$$
and
$$\left[\beta^{true}_{(u,v)} \cdot (1-q), q \cdot \beta^{true}_{(u,v)}(1+q)\right].$$
Clearly, smaller q implies that $\mathcal{M}_q$ is closer to $\mathcal{M}_{true}$.
- $\mathcal{M}_\infty$. Similarly, $\mathcal{M}_\infty$ is obtained from $\mathcal{M}_{true}$ by using a different set of parameters. In particular, all the parameters are selected uniformly at random from $[0, 1]$.

For each empirical distribution, we generate a pool of realizations (Table 3); given K, the realizations used in Social-Inverse are randomly selected from the pool we generate. The training process is done by the n-slack cutting plane algorithm [45] via the implementation of Pystruct [46]. The hyperparameter C is set as 0.01 by default.

E.1.3 Other methods

Conceptually, when $T_{\textcircled{s}} = T_{\textcircled{t}}$, Problem 4 can be taken as a supervised learning task that leverages $S^{\overline{\alpha}_{\textcircled{s}}}_{T_{\textcircled{s}}}$ to build mapping from $\mathcal{X}_{\textcircled{s}}$ to $\mathcal{Y}_{\textcircled{s}}$. When the source task is different from the target task, the success of doing so in mitigating the task migrations hinges on the hypothesis that, for each $X \subseteq V$, the cascade that can maximally influence the nodes in X may also be the cascade that can effectively block the negative cascade from X, which is however not true in some cases. Therefore, as seen from the experimental results, standard learning methods can perform poorly, especially when $T_{\textcircled{s}} \neq T_{\textcircled{t}}$.

Naive Bayes (NB). NB solves this problem by learning $\Pr[X|Y]\Pr[Y]$ for each $Y \in \mathcal{Y}_{\textcircled{t}}$ and $X \in \mathcal{X}_{\textcircled{t}}$. In addition to the Naive Bayes assumption, we assume the nodes in Y are also independent, and therefore, we are to learn $\prod_{u \in X}\prod_{v \in Y}\Pr[u|v]\Pr[v]$ by learning a collection of Bernoulli distributions.

Graph neural network (GNN). GNN is a reasonable choice because the graph structure of G is known to us. For GNN, we adopt the standard implementation [54] including two graph convolution layers followed by a dropout layer with a ratio of 0.4 and a sequence of MLP of size (512, 512, 256) with ReLU as the activation function. The input and output are coded as one-hot vectors. The parameters are computed by minimizing the binary cross-entropy, using an Adam optimizer with a learning rate of 0.01 and an L2 regularizer. The seeds are selected from $\{7, 11, 17, 23, 43\}$.

Deep set prediction network (DSPN). DSPN [55] is a deep architecture designed specifically for dealing with set structures, and it is therefore suitable for our purpose as Problem 4 seeks a mapping between subsets. In implementing DSPN, the best configuration observes an outer learning rate of 0.01, an inner learning rate of 10^7, and an inner iteration of 10, with the set encoder being an MLP encoder of dimension 1024 and the input encoder being an MLP encoder of dimension 256.

High-Degree (HD). Given the size of the decision, the HD method selects the nodes with the highest degree.

Random. This is a baseline method that outputs a decision by randomly selecting nodes.

E.1.4 Training, testing, and evaluation

Given the training size and the testing size, the samples are randomly selected from the pool we generate, and each method is executed once. The entire process is repeated five times, and we report the average performance ratio together with the standard deviation.

Performance ratio. When the target task is DE, for a testing sample (X, Y) and a prediction $\hat{Y}$, the performance is measured by the ratio

$$f^{\mathrm{DE}}_{\mathcal{M}_{true}}(X, \hat{Y}) / f^{\mathrm{DE}}_{\mathcal{M}_{true}}(X, Y);$$

when the target task is DC, the performance is measured by the ratio

$$\frac{f^{\mathrm{DC}}_{\mathcal{M}_{true}}(X, \hat{Y}) - f^{\mathrm{DC}}_{\mathcal{M}_{true}}(X, \emptyset)}{f^{\mathrm{DC}}_{\mathcal{M}_{true}}(X, Y) - f^{\mathrm{DC}}_{\mathcal{M}_{true}}(X, \emptyset)},$$

which means that the performance is measured by the improvement based on the empty decision. In any case, a larger ratio implies a better performance. It is possible that the ratio is occasionally larger than 1 because the sample decisions are approximate solutions.

E.2 Time complexity of Social-Inverse

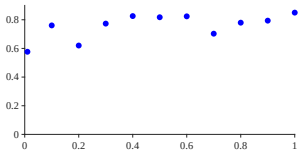
The training complexity of Social-Inverse follows immediately from the analysis of the n-slack cutting plane algorithm [69, 41]. Experimentally, the training cost of Social-Inverse is presented in Table 4, and the cost of making one prediction is less than 1s in all cases. Our experiments are executed on Amazon EC2 C5 Instance with 96 virtual CPUs and a memory of 192G. As we can see from the tables, the training cost is almost linear in K as well as linear in the training size. The training is reasonably fast in all cases. Under the same setting, DC-DE takes more time on $M_{0,1}$ than DE-DC does, but DC-DE is slightly faster than DE-DC on $\mathcal{M}_{\infty}$.

Table 4: **Training time.** The training time denotes the cost (measured by sec) used for computing $\overline{\mathbf{w}}$ under one setting.

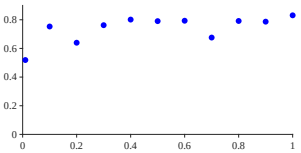
			DC-DE				**DE-DC**			
			5	15	30	60	5	15	30	60
Kro	$\mathcal{M}_{0.1}$	90	13.839	33.331	69.675	133.554	12.207	23.723	49.753	115.068
		270	37.151	75.557	152.075	333.557	20.674	40.084	88.198	217.873
		1350	138.313	369.822	766.713	1519.132	109.309	292.122	382.564	906.334
	$\mathcal{M}_{\infty}$	90	8.545	14.579	23.599	47.149	7.990	12.488	19.890	39.480
		270	18.278	31.937	55.734	112.496	16.300	26.043	44.581	90.024
		1350	86.981	160.363	259.810	497.938	93.215	135.816	252.990	474.851
ER	$\mathcal{M}_{0.1}$	90	5.707	9.197	20.420	29.963	5.324	7.235	13.739	21.746
		270	11.977	21.634	38.295	71.406	9.618	16.201	26.176	58.331
		1350	88.852	103.064	167.285	320.701	84.946	72.091	132.034	218.223
	$\mathcal{M}_{\infty}$	90	4.834	7.695	12.402	21.230	4.460	6.544	10.689	18.063
		270	9.581	17.026	28.196	50.213	8.836	14.301	22.206	40.461
		1350	39.967	82.883	133.520	233.780	50.439	119.346	112.180	190.187
Higgs	$\mathcal{M}_{0.1}$	90	89.026	187.430	364.440	709.585	84.301	174.703	328.194	676.772
		270	201.856	433.293	811.732	1626.409	186.414	394.361	732.861	1462.641
		1350	869.795	1897.335	3685.378	7206.658	929.341	1698.253	3301.660	6289.700
	$\mathcal{M}_{\infty}$	90	85.827	182.209	337.944	652.565	81.637	168.980	312.979	600.219
		270	193.269	413.589	785.807	1528.932	184.091	381.032	704.073	1363.695
		1350	913.599	1852.769	3474.103	6807.414	800.248	1721.938	3058.026	5941.948
Hep	$\mathcal{M}_{\infty}$	90	145.346	309.781	573.465	1113.464	140.602	293.864	534.508	1039.320
		270	332.663	700.047	1307.256	2567.882	318.912	651.294	1212.271	2346.511
		1350	911.897	1898.069	3538.037	7042.835	843.958	1782.247	3180.850	6251.398

E.3 The impact of β.

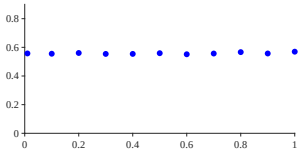
To explore the impact of β, we integrate the coefficient $\widetilde{\alpha}_{\textcircled{t}}(\widetilde{\alpha}_{\textcircled{t}} - \beta)$ as a constant C^* in running the n-slack cutting plane algorithm, and examine the performance of Social-Inverse with different C^* selected from $\{0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$. We are particularly interested in the case where the efficacy of samples is evident in mitigating the task migrations. The results of this part are given in Figure 3. According to the figures, the performance on the Kronecker graph is slightly sensitive to C^*, and larger C^* in general gives better results. On Erdős-Rényi and Higgs, the performance does not vary much with C^* as long as C^* is larger than zero.



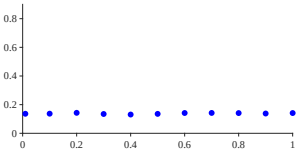
(a) [Kro, DC-DE, $\mathcal{M}_{0.1}$, 30, 270]



(b) [Kro, DC-DE, $\mathcal{M}_{0.5}$, 30, 270]



(c) [ER, DC-DE, $\mathcal{M}_{\infty}$, 30, 1350]



(d) [Higgs, DE-DC, $\mathcal{M}_{\infty}$, 5, 270]

Figure 3: Each subgraph is labeled by [graph, source-target, empirical distribution, K, training size]. The x-axle denotes C^* and the y-axle denotes the performance ratio.

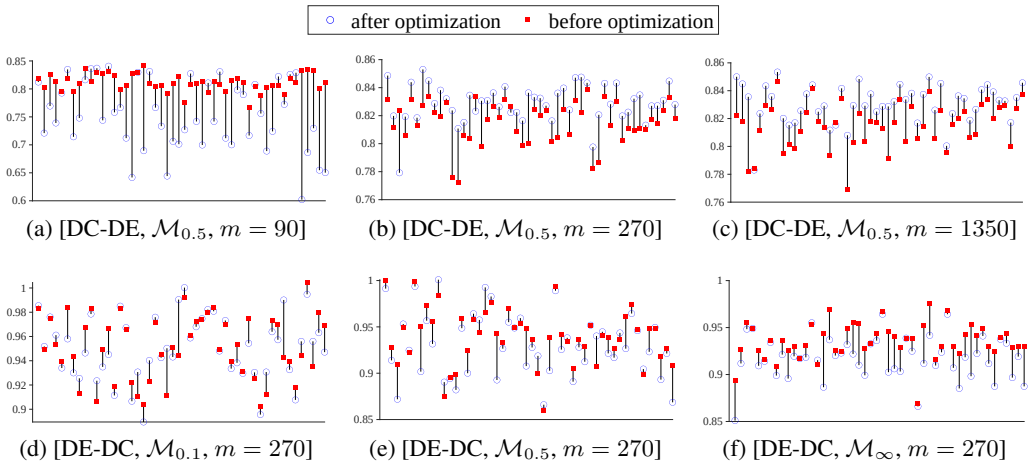
E.4 More experimental results

E.4.1 Full results on Kronecker

The full results on the Kronecker graph are given in Table 5 and Figure 4. The output of HD and Random are independent of the training samples, and they therefore have the results for DE-DE and DC-DE; similar arguments apply to DC-DC and DE-DC. The last column shows the results of GNN under different seeds. In addition to the observations mentioned in the main paper, we see that the ratios of DE-DE (resp., DC-DC) are clearly better than those in DC-DE (resp., DE-DC), which coincides with the expectation that samples from the same task are naturally more effective than those from another task. It is also worth noting that NB is effective in the case of DC-DC. According to the figures, the efficacy of DC samples in solving DE is quite evident. However, under the same setting, the DE samples are not very useful in providing a high-quality weight $\overline{\mathbf{w}}$; one plausible reason is that the performance ratio is already too high (when $K = 30$) so that there is little room for improvement.

Table 5: **Full results on Kronecker.** Each cell presents the mean of performance ratio with std.

		Social-Inverse					**Other Methods**		
	K	5	15	30	60	180			GNN
DE | DE	$\mathcal{M}_\infty$	0.482(0.015)	0.528(0.018)	0.556(0.011)	0.578(0.010)	0.612(0.002)	NB	DSPN	0.431 (23)
	$\mathcal{M}_1$	0.592(0.022)	0.748(0.015)	0.776(0.008)	0.810(0.004)	0.802(0.012)	0.644(0.006)	0.456(0.140)	0.545 (17)
	$\mathcal{M}_{0.5}$	0.644(0.038)	0.784(0.012)	0.824(0.016)	0.846(0.011)	0.856(0.003)	HD	Random	0.486 (11)
	$\mathcal{M}_{0.1}$	0.665(0.049)	0.781(0.040)	0.817(0.025)	0.872(0.008)	0.880(0.005)	0.589(0.003)	0.150(0.014)	0.521 (43)
DC | DE	$\mathcal{M}_\infty$	0.489(0.010)	0.540(0.014)	0.567(0.016)	0.584(0.006)	0.606(0.008)	NB	DSPN	0.459 (23)
	$\mathcal{M}_1$	0.654(0.031)	0.748(0.022)	0.794(0.016)	0.811(0.011)	0.842(0.026)	0.584(0.005)	0.434(0.056)	0.516 (17)
	$\mathcal{M}_{0.5}$	0.653(0.035)	0.767(0.028)	0.837(0.004)	0.851(0.004)	0.850(0.021)	HD	Random	0.515 (11)
	$\mathcal{M}_{0.1}$	0.591(0.036)	0.770(0.025)	0.850(0.018)	0.853(0.040)	0.888(0.007)	0.589(0.003)	0.150(0.014)	0.509 (43)
DE | DC	$\mathcal{M}_\infty$	0.759(0.013)	0.846(0.005)	0.913(0.005)	0.985(0.014)	1.069(0.014)	NB	DSPN	0.660 (23)
	$\mathcal{M}_1$	0.720(0.030)	0.796(0.056)	0.845(0.040)	0.899(0.030)	0.935(0.028)	0.877(0.006)	0.567(0.237)	0.835 (17)
	$\mathcal{M}_{0.5}$	0.708(0.039)	0.850(0.036)	0.937(0.058)	1.021(0.040)	1.104(0.025)	HD	Random	0.731 (11)
	$\mathcal{M}_{0.1}$	0.682(0.037)	0.862(0.014)	0.953(0.018)	1.041(0.026)	1.137(0.026)	0.886(0.006)	0.258(0.003)	0.794 (43)
DC | DC	$\mathcal{M}_\infty$	0.760(0.020)	0.860(0.021)	0.933(0.021)	1.002(0.018)	1.063(0.005)	NB	DSPN	0.751 (23)
	$\mathcal{M}_1$	0.706(0.008)	0.853(0.026)	0.959(0.014)	1.052(0.012)	1.099(0.010)	0.922(0.009)	0.585(0.131)	0.815 (17)
	$\mathcal{M}_{0.5}$	0.732(0.025)	0.892(0.029)	0.995(0.014)	1.097(0.018)	1.169(0.010)	HD	Random	0.779 (11)
	$\mathcal{M}_{0.1}$	0.720(0.022)	0.872(0.024)	1.002(0.013)	1.093(0.012)	1.173(0.012)	0.886(0.006)	0.258(0.003)	0.806 (43)



(a) [DC-DE, $\mathcal{M}_{0.5}$, $m = 90$] (b) [DC-DE, $\mathcal{M}_{0.5}$, $m = 270$] (c) [DC-DE, $\mathcal{M}_{0.5}$, $m = 1350$]

(d) [DE-DC, $\mathcal{M}_{0.1}$, $m = 270$] (e) [DE-DC, $\mathcal{M}_{0.5}$, $m = 270$] (f) [DE-DC, $\mathcal{M}_\infty$, $m = 270$]

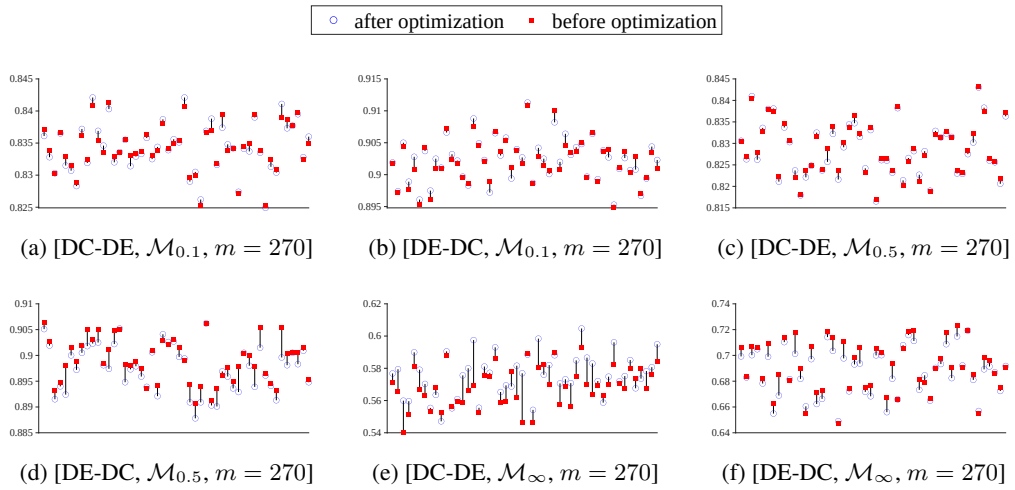
Figure 4: Each subgraph shows the results of 50 testing queries on the Kronecker graph with $K = 30$. For each query, we report the quality (i.e. performance ratio) of the predictions made based on the initial weight (before optimization) and final weight (after optimization).

E.4.2 Full results on Erdős-Rényi

The results on the Erdős-Rényi graph are given in Table 6 and Figure 5. Similar to the observations on the Kronecker graph, it is clear that Social-Inverse performs better when K becomes large or $\mathcal{M}_{em}$ is closer to $\mathcal{M}_{true}$. One interesting observation is that the methods based on neural networks (i.e., GNN and DSPN) can hardly have any optimization effect except for DE-DE, and NB only works when the source task is the same as the target task, which suggests that they cannot mitigate the task migrations on the Erdős-Rényi graph. In addition, according to Figure 5, the efficacy of the samples is evident for solving DC-DE with $\mathcal{M}_{\infty}$, but it is less significant in other cases.

Table 6: **Results on Erdős-Rényi.** Each cell presents the mean of performance ratio with std.

		Social-Inverse				**Other Methods**		
	K	5	15	30	60			GNN
DE-DE	$\mathcal{M}_{\infty}$	0.421(0.009)	0.551(0.28)	0.579(0.023)	0.587(0.010)	DSPN	NB	0.339 (11)
	$\mathcal{M}_{1}$	0.715(0.011)	0.746(0.011)	0.750(0.005)	0.747(0.003)	0.378(0.177)	0.496(0.005)	0.381 (17)
	$\mathcal{M}_{0.5}$	0.784(0.013)	0.816(0.006)	0.828(0.004)	0.830(0.006)	Random	HD	0.317 (13)
	$\mathcal{M}_{0.1}$	0.806(0.006)	0.825(0.006)	0.828(0.007)	0.833(0.006)	0.063(0.002)	0.317(0.006)	0.397 (43)
DC-DE	$\mathcal{M}_{\infty}$	0.424(0.017)	0.541(0.010)	0.585(0.018)	0.577(0.007)	DSPN	NB	0.028 (11)
	$\mathcal{M}_{1}$	0.724(0.014)	0.744(0.005)	0.752(0.003)	0.754(0.003)	0.060(0.028)	0.077(0.005)	0.068 (13)
	$\mathcal{M}_{0.5}$	0.784(0.005)	0.825(0.005)	0.830(0.005)	0.830(0.005)	Random	HD	0.057 (43)
	$\mathcal{M}_{0.1}$	0.798(0.004)	0.829(0.002)	0.833(0.004)	0.833(0.004)	0.063(0.002)	0.317(0.006)	0.035 (17)
DE-DC	$\mathcal{M}_{\infty}$	0.263(0.012)	0.526(0.020)	0.677(0.016)	0.750(0.010)	DSPN	NB	0.049 (11)
	$\mathcal{M}_{1}$	0.725(0.018)	0.773(0.016)	0.796(0.010)	0.800(0.009)	0.042(0.007)	0.041(0.002)	0.053 (13)
	$\mathcal{M}_{0.5}$	0.823(0.006)	0.879(0.002)	0.900(0.002)	0.892(0.002)	Random	HD	0.054 (43)
	$\mathcal{M}_{0.1}$	0.837(0.002)	0.886(0.005)	0.902(0.003)	0.895(0.003)	0.043(0.002)	0.094(0.006)	0.051 (17)
DC-DC	$\mathcal{M}_{\infty}$	0.273(0.019)	0.528(0.022)	0.682(0.019)	0.748(0.019)	DSPN	NB	0.065 (11)
	$\mathcal{M}_{1}$	0.741(0.005)	0.796(0.013)	0.807(0.009)	0.807(0.011)	0.083(0.013)	0.576(0.028)	0.071 (13)
	$\mathcal{M}_{0.5}$	0.820(0.009)	0.879(0.007)	0.901(0.006)	0.895(0.005)	Random	HD	0.073 (43)
	$\mathcal{M}_{0.1}$	0.830(0.008)	0.887(0.009)	0.902(0.001)	0.895(0.002)	0.043(0.002)	0.094(0.006)	0.059 (17)



(a) [DC-DE, $\mathcal{M}_{0.1}$, $m = 270$] (b) [DE-DC, $\mathcal{M}_{0.1}$, $m = 270$] (c) [DC-DE, $\mathcal{M}_{0.5}$, $m = 270$]

(d) [DE-DC, $\mathcal{M}_{0.5}$, $m = 270$] (e) [DC-DE, $\mathcal{M}_{\infty}$, $m = 270$] (f) [DE-DC, $\mathcal{M}_{\infty}$, $m = 270$]

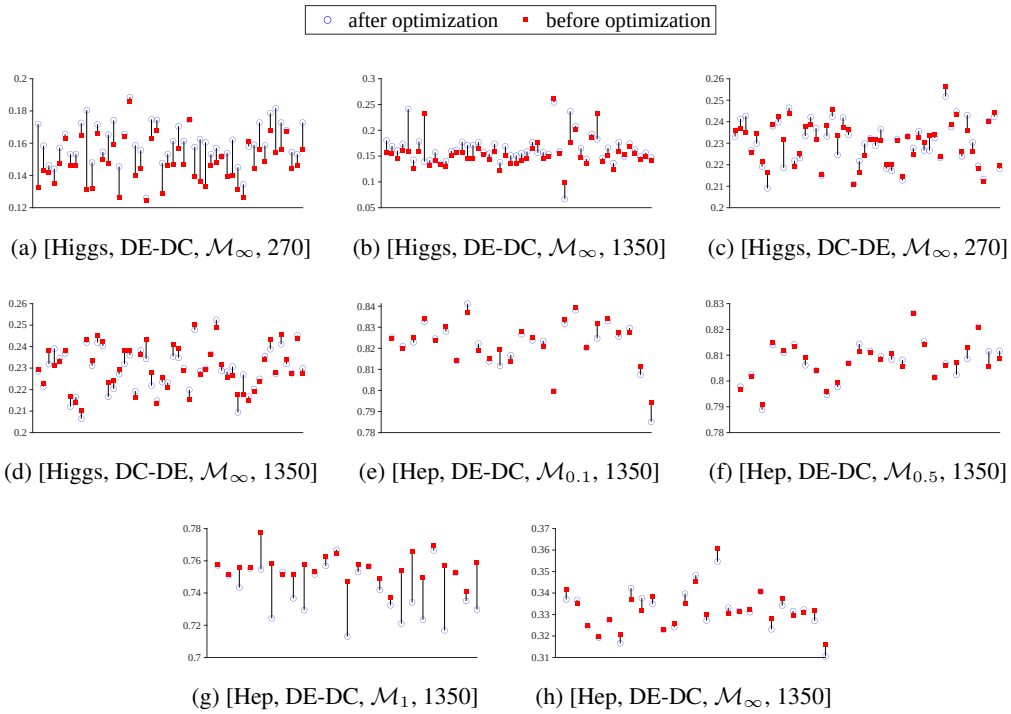
Figure 5: Each subgraph shows the results of 50 testing queries on the Erdős-Rényi graph with $K = 30$. For each query, we report the quality (i.e. performance ratio) of the predictions made based on the initial weight (before optimization) and final weight (after optimization).

E.5 Results on Higgs and Hep

The results on Higgs and Hep can be found in Table 7 and Figure 6. GNN is not feasible for such large graphs due to the memory limit. In addition to the aforementioned observations, we observe that none of the learning-based methods or HD can offer decent performance on large graphs like Higgs and Hep. Second, there is a clear gap between $\mathcal{M}_\infty$ and other empirical distributions, which suggests that prior knowledge is critical when the graph is large. According to Figure 6, the samples are clearly useful on Higgs for the case of DE-DC but less effective for DC-DE; on Hep, the task migration seems to be essential, and therefore, the efficacy of Social-Inverse depends entirely on the realizations but not on the samples at all. We believe that further investigation is needed to explore the conditions in which samples could be leveraged for solving DE-DC or DC-DE on large graphs.

Table 7: **Results on Higgs and Hep** Each cell presents the mean of performance ratio with std.

			Social-Inverse				**Other Methods**	
		K	5	15	30	60		
Higgs	**DE-DE**	$\mathcal{M}_\infty$	0.221(0.008)	0.350(0.008)	0.417(0.059)	0.477(0.010)	DSPN	NB
		$\mathcal{M}_1$	0.802(0.007)	0.844(0.007)	0.862(0.011)	0.876(0.006)	0.015(0.013)	0.122(0.011)
		$\mathcal{M}_{0.5}$	0.882(0.010)	0.940(0.003)	0.950(0.002)	0.956(0.003)	HD	Random
		$\mathcal{M}_{0.1}$	0.919(0.008)	0.950(0.001)	0.960(0.003)	0.960(0.001)	0.0257(0.002)	0.002(0.001)
	DC-DE	$\mathcal{M}_\infty$	0.233(0.013)	0.358(0.007)	0.419(0.010)	0.481(0.008)	DSPN	NB
		$\mathcal{M}_1$	0.805(0.011)	0.860(0.007)	0.872(0.003)	0.878(0.005)	0.004(0.002)	0.034(0.001)
		$\mathcal{M}_{0.5}$	0.897(0.010)	0.939(0.006)	0.951(0.002)	0.960(0.001)	HD	Random
		$\mathcal{M}_{0.1}$	0.919(0.006)	0.952(0.001)	0.960(0.001)	0.960(0.001)	0.0257(0.002)	0.002(0.001)
	DE-DC	$\mathcal{M}_\infty$	0.139(0.012)	0.173(0.017)	0.214(0.010)	0.228(0.028)	DSPN	NB
		$\mathcal{M}_1$	0.809(0.007)	0.891(0.010)	0.934(0.009)	0.951(0.008)	0.030(0.015)	0.019(0.006)
		$\mathcal{M}_{0.5}$	0.874(0.009)	0.960(0.007)	0.991(0.006)	1.010(0.006)	HD	Random
		$\mathcal{M}_{0.1}$	0.877(0.017)	0.975(0.008)	1.010(0.002)	1.036(0.006)	0.075(0.058)	0.004(0.004)
	DC-DC	$\mathcal{M}_\infty$	0.141(0.011)	0.184(0.015)	0.222(0.018)	0.279(0.018)	DSPN	NB
		$\mathcal{M}_1$	0.804(0.011)	0.906(0.009)	0.931(0.007)	0.968(0.004)	0.010(0.007)	0.110(0.010)
		$\mathcal{M}_{0.5}$	0.879(0.007)	0.965(0.007)	1.004(0.003)	1.026(0.007)	HD	Random
		$\mathcal{M}_{0.1}$	0.897(0.007)	0.977(0.007)	1.006(0.005)	1.030(0.004)	0.075(0.058)	0.004(0.003)
Hep	**DE-DE**	$\mathcal{M}_\infty$	0.340(0.008)	0.396(0.007)	0.400(0.007)	0.411(0.006)	DSPN	NB
		$\mathcal{M}_1$	0.728(0.005)	0.763(0.003)	0.773(0.002)	0.774(0.002)	0.038(0.020)	0.344(0.017)
		$\mathcal{M}_{0.5}$	0.797(0.005)	0.846(0.004)	0.856(0.001)	0.862(0.002)	HD	Random
		$\mathcal{M}_{0.1}$	0.844(0.003)	0.889(0.002)	0.899(0.002)	0.901(0.002)	0.067(0.001)	0.003(0.001)
	DC-DE	$\mathcal{M}_\infty$	0.341(0.007)	0.400(0.003)	0.402(0.004)	0.404(0.004)	DSPN	NB
		$\mathcal{M}_1$	0.727(0.004)	0.758(0.002)	0.765(0.008)	0.774(0.005)	0.013(0.005)	0.036(0.001)
		$\mathcal{M}_{0.5}$	0.795(0.004)	0.836(0.007)	0.841(0.006)	0.845(0.009)	HD	Random
		$\mathcal{M}_{0.1}$	0.837(0.011)	0.886(0.005)	0.890(0.003)	0.897(0.004)	0.067(0.001)	0.003(0.001)
	DE-DC	$\mathcal{M}_\infty$	0.329(0.009)	0.517(0.010)	0.608(0.010)	0.684(0.026)	DSPN	NB
		$\mathcal{M}_1$	0.725(0.013)	0.809(0.008)	0.849(0.008)	0.895(0.014)	0.020(0.009)	0.005(0.001)
		$\mathcal{M}_{0.5}$	0.782(0.011)	0.882(0.008)	0.908(0.011)	0.984(0.005)	HD	Random
		$\mathcal{M}_{0.1}$	0.785(0.008)	0.889(0.004)	0.923(0.011)	0.990(0.007)	0.046(0.002)	0.004(0.002)
	DC-DC	$\mathcal{M}_\infty$	0.321(0.004)	0.534(0.016)	0.612(0.009)	0.713(0.011)	DSPN	NB
		$\mathcal{M}_1$	0.711(0.011)	0.800(0.025)	0.843(0.012)	0.911(0.011)	0.023(0.011)	0.046(0.002)
		$\mathcal{M}_{0.5}$	0.776(0.013)	0.858(0.009)	0.898(0.003)	0.926(0.012)	HD	Random
		$\mathcal{M}_{0.1}$	0.789(0.014)	0.865(0.015)	0.935(0.013)	0.932(0.014)	0.046(0.002)	0.004(0.002)



(a) [Higgs, DE-DC, $\mathcal{M}_\infty$, 270] (b) [Higgs, DE-DC, $\mathcal{M}_\infty$, 1350] (c) [Higgs, DC-DE, $\mathcal{M}_\infty$, 270]

(d) [Higgs, DC-DE, $\mathcal{M}_\infty$, 1350] (e) [Hep, DE-DC, $\mathcal{M}_{0.1}$, 1350] (f) [Hep, DE-DC, $\mathcal{M}_{0.5}$, 1350]

(g) [Hep, DE-DC, $\mathcal{M}_1$, 1350] (h) [Hep, DE-DC, $\mathcal{M}_\infty$, 1350]

Figure 6: Each subgraph is labeled by [graph, source-target, empirical distribution, training size] showing the results under one setting with $K = 5$. For each testing query, we report the quality (i.e. performance ratio) of the predictions made based on the initial weight (before optimization) and final weight (after optimization).

F Related works

Social contagion management. The study of lunching information cascades for maximizing the cascade influence is popularized by the seminal work of Kempe *et al.* [25], following which various social contagion management tasks have been proposed and studied, such as scalable influence maximization [26, 17, 28], competitive influence maximization [20, 36], and epidemics source detection [70, 71, 72, 73]. The early studies focus on algorithmic developments based on the setting that the diffusion models are presumably known, where submodular optimization [25, 20] and stochastic optimization [74] are critical in solving these problems. Different from these works, our paper considers the case where the contagion management task is defined over unknown models. To better support real-world applications, another branch of research seeks to develop methodologies for learning diffusion model from data, with a particular focus on solving decision-making tasks based on the learned model (e.g., [31, 75, 76, 77, 32, 30, 29]).

Learn-and-optimize methods. When presented with unknown decision-making models, one natural choice is the learn-and-optimize process. In the initial studies, the learning stage and the optimization stage are treated separately [78, 79, 80, 81]. Such methods are justified when the hidden distribution $\mathcal{M}_{true}$ can be perfectly recovered and the optimization problem can be solved effectively; however, learning errors are not avoidable, either in theory or in practice, thereby leading to unsatisfactory optimization performance (as observed in a few works [82, 83]). A growing body of research has been focused on the idea of learning-for-optimization aiming to design end-to-end training methods that can push the learning phase towards the final objectives [84, 85, 86]; such an idea can be dated back to Bengio's work twenty years ago [87]. While these approaches focus on the optimization effects, they still attempt to learn the decision model by using input-value pairs, which are $((X, Y), f^T_{\mathcal{M}_{true}}(X, Y))$ under our context. In contrast, we seek to solve the problem using data of query-decision pairs $(X, Y^{\overline{\alpha}}_{\mathcal{M}_{true},T,X})$. This means that our data has encoded an optimization process, which we believe is a novel way of thinking.

Inverse engineering. Our work is also related to the study of inverse engineering. One branch of such study focuses on finding a minimal adjustment of the objective functions such that a given set of feasible solutions becomes efficient [88, 89, 90]; another relevant topic is to recover decision-making models or parameterized representations from optimal or suboptimal decisions [91, 92, 93, 94, 95]. A key differentiating factor is that our work seeks to directly compute future decisions using query-decision pairs, rather than accurately inferring the latent diffusion model. Under the setting of reverse decision making, one recent work [36] explores the case of DC-DC, and later, such settings are generalized to abstract stochastic combinatorial optimizations [40]; comparing to these works, we generalize the existing settings and introduce the concept of task migration, which allows the source task to be different from the target task, thereby addressing the case where the data availability of the target task is limited; in addition, this paper introduces a parameterized margin together with new analysis and training schemes to resolve the task migrations, and therefore, our framework generalizes the existing ones (e.g., StratLearner [36] and USCO-Solver [40]) and can be further fine-tuned.