Learnability Gaps of Strategic Classification

Lee Cohen LEECOHENCS@GMAIL.COM
Stanford

Yishay Mansour MANSOUR.YISHAY@GMAIL.COM
Tel Aviv University and Google Research

Shay Moran SMORAN@TECHNION.AC.IL
Departments of Mathematics, Computer Science, and Data and Decision Sciences, Technion and Google Research

Han Shao HAN@TTIC.EDU
Toyota Technological Institute of Chicago

Editors: Shipra Agrawal and Aaron Roth

Abstract

In contrast with standard classification tasks, strategic classification involves agents strategically modifying their features in an effort to receive favorable predictions. For instance, given a classifier determining loan approval based on credit scores, applicants may open or close their credit cards and bank accounts to fool the classifier. The learning goal is to find a classifier robust against strategic manipulations. Various settings, based on what and when information is known, have been explored in strategic classification. In this work, we focus on addressing a fundamental question: the learnability gaps between strategic classification and standard learning.

We essentially show that any learnable class is also strategically learnable: we first consider a fully informative setting, where the manipulation structure (which is modeled by a manipulation graph $G^\star$) is known and during training time the learner has access to both the pre-manipulation data and post-manipulation data. We provide nearly tight sample complexity and regret bounds, offering significant improvements over prior results. Then, we relax the fully informative setting by introducing two natural types of uncertainty.

First, following Ahmadi et al. (2023), we consider the setting in which the learner only has access to the post-manipulation data. We improve the results of Ahmadi et al. (2023) and close the gap between mistake upper bound and lower bound raised by them.

Our second relaxation of the fully informative setting introduces uncertainty to the manipulation structure. That is, we assume that the manipulation graph is unknown but belongs to a known class of graphs. We provide nearly tight bounds on the learning complexity in various unknown manipulation graph settings. Notably, our algorithm in this setting is of independent interest and can be applied to other problems such as multi-label learning.

Keywords: strategic classification, PAC learning, mistake bound in online learning, VC dimension, Littlestone dimension

1. Introduction

Strategic classification lies at the intersection of machine learning and game theory, addressing the intricate challenges arising from the strategic behavior of self-interested agents seeking to manipulate classifiers for personal advantage (Brückner and Scheffer, 2011; Hardt et al., 2016). This emerging field has attracted significant attention due to its relevance in real-world applications, such as loan approvals and college admissions. For example, in loan applications, individuals might strategically

try to increase their credit scores by acquiring additional credit cards, or by changing jobs to increase their salaries. Within the context of college admissions, students may opt for less challenging courses to improve their GPA, retake the SAT, or even transfer schools, all to improve their chances of acceptance. This introduces a more complex environment, where agents strategically adapt their features to influence the outcome of the current classifier while preserving their true underlying qualifications (e.g., being successful in college or repaying a loan). The foundational challenge in strategic classification resides in selecting a classifier that is robust to manipulations, ensuring the learning process remains resilient to strategic behavior.

When modeling strategic classification, the manipulation power of the agents is considered limited. Our approach to representing feasible manipulations involves a manipulation graph—a concept initially introduced in Zhang and Conitzer (2021) and subsequently studied in, e.g., Lechner and Urner (2022); Lechner et al. (2023); Ahmadi et al. (2023). In this model, each agent has a feature vector x, represented as a node in a graph, and a binary label, y. An arc from x to x' indicates that an agent with the feature vector x can adjust their feature vector to x'—essentially, an arc denotes a potential manipulation. Similar to the assumptions made in Ahmadi et al. (2023), we posit that the manipulation graph has a bounded degree, k. This constraint adds a realistic dimension to the model, reflecting the limited manipulation capabilities of the agents within the strategic classification framework. In addition, as we discuss in Theorems 19 and 21, this bounded degree constraint is also necessary in the sense that relaxing it leads to impossibility results for extremely easy-to-learn hypothesis classes.

Consider a sequence of agents arriving and interacting with the learner. At each time step, the learner selects a hypothesis, and the current agent manipulates its feature vector x in accordance with this hypothesis. Each agent is aiming to secure a positive classification, and thus an agent x will manipulate to node x' only when such manipulation alters their predicted value from negative to positive. Importantly, y, the true label of the agent, does not depend on the current predictor. Also note that the learner does not have the ability to counterfactually inquire about the manipulations the previous agents would have adopted under a different hypothesis.

In this work, we aim to address a fundamental question:

Does learnability imply strategic learnability?

We study the above question within both PAC and online learning frameworks, addressing it across both realizable and agnostic contexts. In each case, we analyze various levels of information received by the learner as feedback.

We start with a *fully informative* setting. Here, the learner has full information of both the manipulation graph as well as the pre- and post-manipulation features: at each time step the learner (1) observes the pre-manipulation features of the current agent, (2) selects and implements a hypothesis (according to which the agent manipulates), and (3) observes both the manipulated features and the true label of the agent.

We are mainly interested in this model because it is the simplest and therefore a natural first step from a theoretical perspective. Nevertheless, it still could be interesting from a practical perspective because it aligns with real-world situations like customers inquiring about mortgage rates and providing their credit score or income as input, or in college admissions where colleges can access sophomore-year grades. In other scenarios, such as those introduced by by Shao et al. (2023), pre-manipulation features are known. For instance, when a teacher assigns a writing task or take-home exam to students. While the teacher may have knowledge of students' abilities (pre-existing features)

from class performance, grading must objectively evaluate the work submitted for that specific assignment. Students might attempt manipulation using resources like ChatGPT. The teacher aims to design assignments resilient to such manipulation tools. This basic setting already deviates from the standard learning paradigm, as the learner needs to consider the potential manipulation of the current agents.

We consider the fully informative setting as a baseline. The other two settings we consider deviate from this baseline by introducing two pragmatic elements of uncertainty regarding the learner's knowledge: one concerning the pre-manipulation features (previously studied for finite hypotheses class in Ahmadi et al. (2023)), and the other regarding the manipulation graph (this type of uncertainty is introduced in this work).

In the second setting, *post-manipulation feedback*, the learner selects the hypothesis before observing any information about the agent. Then, after the agent's manipulation and its classification, the learner observes either (i) the original feature vector, or (ii) the manipulated feature vector (we consider both variants). Thus, significantly less information is provided to the learner compared to the first setting, wherein the learner could compute the set of potential manipulations for each feature vector x for every hypothesis. This setting was previously studied in Ahmadi et al. (2023) which focused on finite hypotheses classes. There are also scenarios in which the original features are observed afterward—for example, when a high-school student takes the SAT test multiple times, most colleges commit to only consider the highest one (or even to "superscore" the test by considering the highest score separately in each section), and some colleges require students to submit all SAT scores (reference).

We highlight that while we assume knowledge of the manipulation graph for the first two settings, our algorithms exclusively depend on local information concerning the manipulation set of the current feature vector. Thus, our algorithms remain applicable in scenarios where global information may not be readily available.

In the last setting, the *unknown manipulation graph*, we relax the fully informative baseline by introducing uncertainty about the manipulation graph. So, the learner observes both the pre- and post-manipulation feature vectors, but no longer has access to the entire manipulation graph. Instead, the learner has prior knowledge in the form of a finite class of graphs $\mathcal{G}$. The class $\mathcal{G}$ is assumed to contain the true manipulation graph in the realizable setting, and in the agnostic setting, $\mathcal{G}$ serves as a comparator class for competitive/regret analysis. This setting is a natural variant of the fully informative feedback model. For instance, a lender can observe the credit score of a customer but does not know the (at most k) easiest and most effective types of manipulations for the agent to pursue —whether they involve obtaining more credit cards, increasing their salary, or even changing employment status from unemployed to employed.

From a technical perspective, the last setting gives rise to a natural graph learning task: consider an unknown target graph. At training time, we observe random examples of the form (vertex v, and a random arc from v). During testing, we receive a random vertex u (drawn from the same distribution as the training nodes), and our objective is to predict the entire neighborhood of u. This problem is motivated by applications such as recommendation systems like Netflix, where a learner observes a random user (vertex) and a random movie they watched (random arc). During testing, the learner observes a user and aims to predict a small set of movies to recommend to this user. Our algorithm in the unknown manipulation graph setting can be applied to solve this neighborhood learning problem.

1.1. Results Overview

In classical binary classification, it is well-known that Probably Approximately Correct (PAC) learnability is characterized by the Vapnik–Chervonenkis (VC) dimension (Vapnik and Chervonenkis, 1974) and that online learnability is characterized by the Littlestone dimension (Littlestone, 1988) (see Appendix B for their definitions). As detailed below, we essentially show that *every learnable class is also strategically learnable*. For certain scenarios, we efficiently transform standard learning algorithms into their strategic counterparts, while for others, we develop entirely new algorithms. Please see Table 1.1 for a summary according to different settings.

Fully Informative Setting We establish upper and lower bounds for the sample complexity and mistake bounds, which are tight up to logarithmic factors. In particular, our bounds imply that in this basic setting, learnability implies strategic learnability. Quantitatively, we prove that the learning complexity of strategic learning is larger by at most a multiplicative logarithmic factor of the maximal degree of the manipulation graph. We further prove that this logarithmic factor is necessary, which implies that manipulation indeed makes learning harder.

In this setting, both PAC and online learnability are captured by the VC and Littlestone dimension of the corresponding classes of strategic losses (i.e. the 0-1 loss under manipulations). In the PAC setting we show that the strategic VC dimension is $\widetilde{\Theta}(d_1 \log k)$, where d_1 is the standard VC dimension of $\mathcal{H}$. This bound is tight up to a logarithmic factor in d_1 (see Theorems 18 and 19). In the online setting we provide a tight bound of $d_2 \log k$ on the strategic Littlestone dimension of $\mathcal{H}$, where d_2 is the standard Littlestone dimension of $\mathcal{H}$ (see Theorems 20 and 21).

Algorithmically, in the PAC setting any empirical risk minimizer w.r.t. the strategic losses achieves near optimal sample complexity in both the realizable and agnostic setting. In the online setting we construct an efficient transformation that converts an online algorithm in the standard setting to a strategic online algorithm. Our transformation involves using the algorithm from the standard setting to generate multiple experts, and running a weighted-majority vote scheme over these experts.

Post-Manipulation Feedback Setting In the PAC setting, we derive sample complexity bounds identical to those that appear in the fully informative setting. This occurs because, when the graph is known, the timing of observing the original feature—whether beforehand or afterward—does not make any difference. Even if the learner can only observe manipulated features, by implementing the all-positive or all-negative hypothesis at each round, one can obtain the same information feedback as observing original features.

However, in online learning, this setting is more subtle because we cannot separate exploration (training) from exploitation (test). To handle this we again devise an efficient transformation that that converts online learners from the classical setting to strategic online learners. Our algorithm here is similar to the one from the fully informative setting, yet the optimal mistake bound here is $d_2 k$, which has an exponentially worse dependence on k (see Theorems 9 and 10). This optimal mistake bound also resolves an open question posed by Ahmadi et al. (2023).

Unknown Manipulation Graph Setting We answer our overarching question affirmatively for this setting too. The main challenge of PAC strategic learning with an unknown graph setting is that we cannot obtain an unbiased estimate of the loss of each graph, and simply running ERM does not work. We tackle this challenge by using the expected degree as a regularizer over the graphs. Then, in the realizable setting, by finding the consistent minimal empirical degree graph-hypothesis pair, we can obtain sample complexity $\widetilde{O}((d_1 \log k + k \log |\mathcal{G}|)/\varepsilon)$ (Theorem 6).

Setting	Sample complexity		Regret
	Realizable	Agnostic	Realizable$^{(*)}$
Fully informative	$\widetilde{O}(\frac{d_1 \log k}{\varepsilon})$ (Thm 18) $\Omega(\frac{d_1 \log k}{\varepsilon})$ (Thm 19)	$\widetilde{O}(\frac{d_1 \log k}{\varepsilon^2})$ (Thm 18) $\Omega(\frac{d_1 \log k}{\varepsilon^2})$ (Thm 19)	$O(d_2 \log k)$ (Thm 20) $\Omega(d_2 \log k)$ (Thm 21)
Post-manipulation feedback			$\widetilde{O}(d_2 k)$ (Thm 9) $\Omega(d_2 k)$ (Det, Thm 10)
Unknown manipulation graph	$\widetilde{O}(\frac{d_1 \log k + k \log(\lvert\mathcal{G}\rvert)}{\varepsilon})$ (Thm 6) $\Omega(\log \lvert\mathcal{G}\rvert)$ (Ada, Thm 7)	$O(\frac{k^2 \log(\lvert\mathcal{G}\rvert) + d_1 \log k}{\varepsilon^2})$ (Mul, Thm 28)	$O(d_2 k \log k + \log \lvert\mathcal{G}\rvert)$ (Thm 12) $\widetilde{\Omega}(\log \lvert\mathcal{G}\rvert)$ (Det, Thm 13)

Table 1: Summary of results, where k is the upper bound of the maximum out-degree of the underlying manipulation graph $G^\star$ as well as all graphs in $\mathcal{G}$, $d_1 = \mathrm{VCdim}(\mathcal{H})$ is the VC dimension of $\mathcal{H}$, and $d_2 = \mathrm{Ldim}(\mathcal{H})$ is the Littlestone dimension of $\mathcal{H}$. *Det* means a lower bound for all deterministic algorithms. *Ada* means in this lower bound, the true graph is generated adaptively. *Mul* means that there is a multiplicative factor of k in the loss. $^{(*)}$ Some results can be extended to the agnostic case by standard techniques as discussed in Appendix F.

In the agnostic PAC setting, although we cannot obtain an unbiased estimate of the graph loss, we propose a proxy loss, which can approximate the graph loss up to a multiplicative factor k. By minimizing the empirical proxy loss to find an approximate graph and running ERM over the strategic loss under this approximate graph, we are able to obtain a hypothesis that is at most a factor of k from the best hypothesis-graph pair (see Theorem 28).

In the realizable online setting, we design an algorithm that runs the online learning algorithm from the post-manipulation feedback setting over the majority vote of the graphs and achieves a mistake bound of $O(d_2 k \log(k) + \log |\mathcal{G}|)$ (see Theorem 12). In both realizable PAC and online settings, we provide a lower bound with logarithmic dependency on $|\mathcal{G}|$, demonstrating that $\log |\mathcal{G}|$ is inherent (see Theorems 7 and 13). We further discuss the application of our results in the unknown graph setting to the multi-label learning problem, which is of independent interest.

Related work Due to space constraints, we defer related work to Appendix A.

2. Model

2.1. Strategic Classification

Throughout this work, we focus on the binary classification task in both adversarial online setting and PAC setting. Let $\mathcal{X}$ denote the feature vector space, $\mathcal{Y} = \{0, 1\}$ denote the label space, and $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$ denote the hypothesis class. In the strategic setting, an example (x, y), referred to as an *agent* in this context, consists of a pair of feature vector and label. An agent will try to receive a positive prediction by modifying their feature vector to some reachable feature vectors. Following prior works (e.g., Zhang and Conitzer (2021); Lechner and Urner (2022); Ahmadi et al. (2023); Lechner et al. (2023)), we model the set of reachable feature vectors as a manipulation graph $G = (\mathcal{X}, \mathcal{E})$, where the nodes are all feature vectors in $\mathcal{X}$. For any two feature vectors $x, x' \in \mathcal{X}$, there is a directed arc $(x, x') \in \mathcal{E}$ from x to x' if and only if an agent with feature vector x can manipulate to x'. We let $N_G(x) = \{x' | (x, x') \in \mathcal{E}\}$ denote the out-neighborhood of x in graph G and $N_G[x] = \{x\} \cup N_G(x)$

denote the closed out-neighborhood. Let $k(G)$ denote the maximum out-degree of graph G. When a hypothesis $h \in \mathcal{Y}^{\mathcal{X}}$ is implemented, if an agent (x, y) is classified by h as negative at their original feature vector x but has a neighbor which is classified as positive by h, she will manipulate to such neighbor. Otherwise, she will stay at her original feature vector, x. Either way, the label of the agent remains y. Formally,

Definition 1 (Manipulated feature) *Given a manipulation graph $G = (\mathcal{X}, \mathcal{E})$ and a hypothesis h, the manipulated feature vector of x induced by (G, h) is*

$$\pi_{G,h}(x) = \begin{cases} v \in \mathcal{X}_{h,+} \cap N_G(x) & \text{if } h(x) = 0 \text{ and } \mathcal{X}_{h,+} \cap N_G(x) \neq \emptyset, \\ x & \text{otherwise,} \end{cases}$$

where $\mathcal{X}_{h,+} = \{x' \in \mathcal{X} | h(x') = 1\}$ is the positive region by h, and the tie-breaking here is assumed to be arbitrary unless specified otherwise.

The post-manipulation labeling of the agent with original feature vector x by a hypothesis h is determined by their prediction at the resulting manipulated feature vector, $\pi_{G,h}(x)$.

Definition 2 (Manipulation-induced labeling) *Given a manipulation graph $G = (\mathcal{X}, \mathcal{E})$ and a hypothesis h, the labeling of x induced by (G, h) is defined as $\overline{h}_G(x) = h(\pi_{G,h}(x))$. We define $\overline{\mathcal{H}}_G = \{\overline{h}_G | h \in \mathcal{H}\}$ to be the $(G, \mathcal{H})$-induced labeling class of hypothesis class $\mathcal{H}$.*

When implementing a hypothesis h, the loss incurred at the agent (x, y) is the misclassification error when the agent manipulates w.r.t. h. We define the strategic loss as follows.

Definition 3 (Strategic loss) *Given the manipulation graph G, for any hypothesis $h \in \mathcal{Y}^{\mathcal{X}}$, the strategic loss of h at agent (x, y) is*

$$\ell_G^{str}(h, (x, y)) := \mathbb{1}(\overline{h}_G(x) \neq y) \,.$$

2.2. Learning with Strategic Agents

Let $G^\star$ denote the underlying true manipulation graph and let $k = k(G^\star)$ denote the maximum out-degree of $G^\star$. In the strategic setting, a sequence of agents arrives and interacts with the learner. The learner perceives agents through the lens of the currently implemented hypothesis, and lacks the ability to counterfactually inquire about the manipulations previous agents would have adopted under a different hypothesis. More formally, consider a sequence of T agents $(x_1, y_1), \ldots, (x_T, y_T)$ arriving sequentially. At each round t, an agent (x_t, y_t) arrives. Then, the learner implements a hypothesis $h_t \in \mathcal{Y}^{\mathcal{X}}$ and reveals h_t to the agent. After observing h_t, the agent responds by manipulating her feature vector from x_t to $v_t = \pi_{G^\star, h_t}(x_t)$. At the end of the round, the learner receives the result of her prediction on the manipulated feature, $\widehat{y}_t = h_t(v_t)$, and the true label, y_t. The interaction protocol (which repeats for $t = 1, \ldots, T$) is described in Protocol 1. In this work, we consider both the PAC and adversarial online setting.

PAC Learning In the PAC setting, the agents are sampled from an underlying distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$. For any data distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, the strategic population loss of h is defined as

$$\mathcal{L}^{\text{str}}_{G^\star, \mathcal{D}}(h) := \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\ell^{\text{str}}_{G^\star}(h, (x, y))\right] \,.$$

Protocol 1 Learner-Agent Interaction

1: **for** $t = 1, \ldots, T$ **do**
2: The environment picks an agent (x_t, y_t). *// In the online setting, the agent is chosen adversarially, while in the PAC setting, the agent is sampled i.i.d.*
3: The learner implements a hypothesis $h_t \in \mathcal{Y}^{\mathcal{X}}$ and discloses h_t to the agent
4: The agent manipulates to $v_t = \pi_{G^\star, h_t}(x_t)$
5: The learner receives prediction $\widehat{y}_t = h_t(v_t)$, the true label y_t and possibly additional information about x_t and/or v_t
6: **end for**

Similar to classical PAC learning, the learner's goal is to find a hypothesis h with low strategic population loss $\mathcal{L}^{\text{str}}_{G^\star, \mathcal{D}}(h)$. However, in contrast to classical PAC learning, where the learner obtains a batch of i.i.d. training data, the strategic setting involves an interactive training phase. The interactive phase is due to the fact that the learner's hypothesis influences the information the learner observes. For this reason, in the interactive phase, the learner sequentially modifies her hypothesis. As a result, we explore a scenario in which the learner interacts with T i.i.d. agents (as illustrated in Protocol 1), and produces a predictor after completing T rounds of interaction. In the *realizable* case, there exists a perfect classifier $h^\star \in \mathcal{H}$ with strategic population loss $\mathcal{L}^{\text{str}}_{G^\star, \mathcal{D}}(h^\star) = 0$.

Online Learning In the online setting, the agents are generated by an adversary. The learner's goal is to minimize the total number of mistakes, $\sum_{t=1}^{T} \mathbb{1}(\widehat{y}_t \neq y_t)$. More specifically, the learner's goal is to minimize the Stackelberg regret[1] with respect to the best hypothesis $h^\star \in \mathcal{H}$ in hindsight, had the agents manipulated given $h^\star$:

$$\text{Regret}(T) := \sum_{t=1}^{T} \ell^{\text{str}}_{G^\star}(h_t, (x_t, y_t)) - \min_{h^\star \in \mathcal{H}} \sum_{t=1}^{T} \ell^{\text{str}}_{G^\star}(h^\star, (x_t, y_t))$$

In the *realizable* case, for the sequence of agents $(x_1, y_1), \ldots, (x_T, y_T)$, there exists a perfect hypothesis $h^\star \in \mathcal{H}$ that makes no mistakes, i.e., $\ell^{\text{str}}_{G^\star}(h^\star, (x_t, y_t)) = 0$ for all $t = 1, \ldots, T$. In this case, the Stackelberg regret is reduced to the number of mistakes.

2.3. Uncertainties about Manipulation Structure and Features

As discussed in the related work (Appendix A), various settings based on information about the manipulation structure and features have been studied. In this work, we begin with the simplest fully informative setting, where the underlying manipulation graph $G^\star$ is known, and the original feature vector x_t is revealed to the learner before the implementation of the hypothesis. We use this fully informative setting as our baseline model and investigate two forms of uncertainty: either the original feature vector x_t is not disclosed beforehand or the manipulation graph $G^\star$ is unknown.

1. **Fully Informative Setting:** Known manipulation graph and known original feature beforehand. More specifically, the true manipulation graph $G^\star$ is known, and x_t is observed before the learner implements h_t. This setting is the simplest and serves as our baseline, providing the learner with the most information. Some may question the realism of observing x_t beforehand, but there are scenarios in which this is indeed the case. For instance, when customers inquire about mortgage rates and provide their credit score or income as input, or in college admissions where colleges

1. This is Stackelberg regret since the learner selects a hypothesis h_t and assumes that the agent will best respond to h_t.

can access sophomore-year grades. In cases where each agent represents a subpopulation and known statistics about that population are available, pre-observation of x_t can be feasible.

2. **Post-Manipulation Feedback Setting:** Here the underlying graph $G^\star$ is still known, but the original feature vector x_t is not observable before the learner implements the hypothesis. Instead, either the original feature vector x_t or the manipulated feature vector v_t are revealed afterward (we consider both variants). It is worth noting that, since the learner can compute $v_t = \pi_{G^\star,h_t}(x_t)$ on her own given x_t, knowing x_t is more informative than knowing v_t. Ahmadi et al. (2023) introduced this setting in the context of online learning problems and offered preliminary guarantees on Stackelberg regret, which depend on the cardinality of the hypothesis class $\mathcal{H}$ and the maximum out-degree of graph $G^\star$.
3. **Unknown Manipulation Graph Setting:** In real applications, the manipulation graph may not always be known. We therefore introduce the unknown manipulation graph setting in which we relax the assumption of a known graph. Instead, we assume that the learner has prior knowledge of a certain graph class $\mathcal{G}$; we consider both the realizable setting in which $G^\star \in \mathcal{G}$ and the agnostic setting in which $\mathcal{G}$ serves as a comparator class. When the true graph $G^\star$ is undisclosed, the learner cannot compute v_t from x_t. Thus, there are several options depending on when the learner gets the feedback: observing x_t before implementing h_t followed by v_t afterward, observing (x_t, v_t) after the implementation, and observing x_t only (or v_t only) after the implementation, arranged in order of increasing difficulty. In this work, we mainly focus on the most informative option, which involves observing x_t beforehand followed by v_t afterward. However, we show that in the second easiest setting of observing (x_t, v_t) afterward, no deterministic algorithm can achieve sublinear (in $|\mathcal{H}|$ and $|\mathcal{G}|$) mistake bound in the online setting.

Note that the first two settings encompass all possible scenarios regarding the observation timing of x_t and v_t when $\mathcal{G}^\star$ is known, given that v_t depends on the implemented hypothesis. It is important to emphasize that our algorithms do not necessitate knowledge of the entire graph. Instead, we only require specific local manipulation structure information, specifically, knowledge of the out-neighborhood of the currently observed feature.

3. Highlights of Some Main Results

Due to the page limit, we only highlight a few of our main results and technical ideas in this section and defer other results and complete proofs to the appendices.

3.1. PAC Learning in Fully Informative and Post-Manipulation Feedback Settings

When the manipulation graph $G^\star$ is known, and we have an i.i.d. sample $S = ((x_1, y_1), \ldots, (x_T, y_T))$, we can obtain a hypothesis $\widehat{h}$ by minimizing the empirical strategic loss,

$$\widehat{h} = \operatorname*{arg\,min}_{h \in \mathcal{H}} \sum_{t=1}^{T} \ell^{\mathrm{str}}_{G^\star}(h, (x_t, y_t)) \,. \tag{1}$$

Although in these two settings we assume the learner knows the entire graph $G^\star$, the learner only needs access to the out-neighborhood $N_{G^\star}(x_t)$ to implement ERM. If only the pair $(v_t, N_{G^\star}(v_t))$ is observed, we can obtain the knowledge of $(x_t, N(x_t))$ by implementing $h_t = \mathbb{1}(\mathcal{X})$ so that $v_t = x_t$. We can then guarantee that $\widehat{h}$ achieves low strategic population loss by VC theory. Hence, all we need to do for generalization is to bound the VC dimension of $\overline{\mathcal{H}}_{G^\star}$. We establish that $\mathrm{VCdim}(\overline{\mathcal{H}}_{G^\star})$

can be both upper bounded (up to a logarithmic factor in $\mathrm{VCdim}(\mathcal{H})$) and lower bounded by $\mathrm{VCdim}(\mathcal{H})\log k$. This implies that for any hypothesis class $\mathcal{H}$ that is PAC learnable, it can also be strategically PAC learnable, provided that the maximum out-degree of $G^\star$ is finite. Achieving this only necessitates a sample size that is $\log k$ times larger and this logarithmic factor is necessary in some cases.

Theorem 4 *For any hypothesis class $\mathcal{H}$ and graph $G^\star$, we have $\mathrm{VCdim}(\overline{\mathcal{H}}_{G^\star}) \le d\log(kd)$ where $d = \mathrm{VCdim}(\mathcal{H})$. Moreover, for any $d, k > 0$, there exists a class $\mathcal{H}$ with $\mathrm{VCdim}(\mathcal{H}) = d$ and a graph $G^\star$ with maximum out-degree k, such that $\mathrm{VCdim}(\overline{\mathcal{H}}_{G^\star}) \ge d\log k$.*

Proof sketch of lower bound Let us start with the simple case of $d = 1$. Consider $(k+1)\log k$ nodes $\mathcal{X} = \{x_{i,j} | i = 1, \dots, \log k, j = 0, \dots, k\}$, where there is an arc from $x_{i,0}$ to each of $\{x_{i,j} | j = 1, \dots, k\}$ for all $i \in [\log k]$ in $G^\star$. For $j = 1, \dots, k$, let $\mathrm{bin}(j)$ denote the $\log k$-bit-binary representation of j. Let hypothesis h_j be the hypothesis that labels $x_{i,j}$ by $\mathrm{bin}(j)_i$ and all the other nodes by 0. Let $\mathcal{H} = \{h_j | j \in [k]\}$. Note that $\mathrm{VCdim}(\mathcal{H}) = 1$. This is because for any two points $x_{i,j}$ and $x_{i',j'}$, if $j \neq j'$, no hypothesis will label them as positive simultaneously; if $j = j'$, they can only be labeled as $(\mathrm{bin}(j)_i, \mathrm{bin}(j)_{i'})$ by h_j and $(0,0)$ by all the other hypotheses. However, $\overline{\mathcal{H}}_{G^\star}$ can shatter $\{x_{1,0}, x_{2,0}, \dots, x_{\log k,0}\}$ since h_j's labeling over $\{x_{1,0}, x_{2,0}, \dots, x_{\log k,0}\}$ is $\mathrm{bin}(j)$. Hence, we have $\mathrm{VCdim}(\overline{\mathcal{H}}_{G^\star}) = \log k$. We extend the example to the case of $\mathrm{VCdim}(\mathcal{H}) = d$ by making d independent copies of $(\mathcal{X}, \mathcal{H}, G^\star)$. The detailed proof of Theorem 4 appears in Appendix C. ■

Remark 5 *We have similar results for Littlestone dimension as well. More specifically, that $\mathrm{Ldim}(\overline{\mathcal{H}}_{G^\star}) \le \mathrm{Ldim}(\mathcal{H}) \cdot \log k$ for any $\mathcal{H}, G^\star$ (Theorem 20). We also show that $\mathrm{Ldim}(\overline{\mathcal{H}}_{G^\star}) \ge \mathrm{Ldim}(\mathcal{H}) \cdot \log k$ using the same example we construct for VC dimension (Theorem 21). In contrast with the PAC setting where we derive the sample complexity bounds by a combinatorial analysis of the VC dimension, in the online setting we obtain the upper bound using an explicit algorithm.*

3.2. PAC Learning in the Unknown Manipulation Graph Setting

Without the knowledge of $G^\star$ (either the entire graph or the local structure), we can no longer implement ERM. For any implemented predictor h, if $h(x) = 0$ and $N_{G^\star}(x) \cap \mathcal{X}_{h,+}$ is non-empty, we assume that the agent will break ties randomly by manipulating to[2] $\pi_{G^\star,h}(x) \sim \mathrm{Unif}(N_{G^\star}(x) \cap \mathcal{X}_{h,+})$. Since each agent can only manipulate within her neighborhood, we define a graph loss $\mathcal{L}_{\text{neighborhood}}$ for graph G as the 0-1 loss of neighborhood prediction,

$$\mathcal{L}_{\text{neighborhood}}(G) := \mathbb{P}_{(x,y)\sim\mathcal{D}}(N_{G^\star}(x) \neq N_G(x)) \,.$$

Then we upper bound the strategic loss by the sum of the graph loss of G and the strategic loss under the graph G, i.e.,

$$\mathcal{L}^{\text{str}}_{G^\star,\mathcal{D}}(h) \le \mathcal{L}_{\text{neighborhood}}(G) + \mathcal{L}^{\text{str}}_{G,\mathcal{D}}(h), \text{ for all } G \,. \tag{2}$$

This upper bound implies that if we can find a good approximate graph, we can then solve the problem by running ERM over the strategic loss under the approximate graph. The main challenge of finding a good approximation of $G^\star$ is that **we cannot obtain an unbiased estimate of $\mathcal{L}_{\text{neighborhood}}(G)$**. This is because we cannot observe $N_{G^\star}(x_t)$. The only information we could obtain regarding $G^\star$ is a random neighbor of x_t, i.e., $v_t \sim \mathrm{Unif}(N_{G^\star}(x_t) \cap \mathcal{X}_{h_t,+})$. While one may think of finding a consistent graph whose arc set contains all arcs of (x_t, v_t) in the historical data, such consistency

2. This is not necessary. All we need is a fixed tie-breaking way that is consistent in both training and test time.

is not sufficient to guarantee a good graph. For example, considering a graph G whose arc set is a superset of $G^\star$'s arc set, then G is consistent but its graph loss could be high. We tackle this challenge through a regulation term based on the empirical degree of the graph.

In the realizable case, where there exists a perfect graph and a perfect hypothesis, i.e., there exists $G^\star$ such that $\mathcal{L}_{\text{neighborhood}}(G^\star) = 0$ and $\mathcal{L}^{\text{str}}_{G^\star,\mathcal{D}}(h^\star) = 0$, our algorithm is described as follows.

- Prediction: At each round t, after observing x_t, we implement the hypothesis $h(x) = \mathbb{1}(x \neq x_t)$, which labels all every point as positive except x_t. This results in a manipulated feature vector v_t sampled uniformly from $N_{G^\star}(x_t)$.
- Output: Let $(\widehat{G}, \widehat{h})$ be a pair that holds

$$(\widehat{G}, \widehat{h}) \in \operatorname*{arg\,min}_{(G,h)\in(\mathcal{G},\mathcal{H})} \sum_{t=1}^{T} |N_G(x_t)|$$

$$\text{s.t.} \quad \sum_{t=1}^{T} \mathbb{1}(v_t \notin N_G(x_t)) = 0\,. \tag{3}$$

$$\sum_{t=1}^{T} \mathbb{1}(\overline{h}_G(x_t) \neq y_t) = 0\,. \tag{4}$$

  Notice that Eq (3) guarantees G is consistent with all arcs of (x_t, v_t) in the historical data and Eq (4) guarantees that h has zero empirical strategic loss when the manipulation graph is G. $(\widehat{G}, \widehat{h})$ is the graph-hypothesis pair that satisfies Eqs (3) and (4) with **minimal empirical degree**. Finally, we output $\widehat{h}$.

Next, we derive sample complexity bound to guarantee that $\widehat{h}$ will have a small error.

Theorem 6 *For any hypothesis class $\mathcal{H}$ with $\mathrm{VCdim}(\mathcal{H}) = d$, the underlying true graph $G^\star$ with maximum out-degree at most k, finite graph class $\mathcal{G}$ in which all graphs have maximum out-degree at most k, any $(\mathcal{G}, \mathcal{H})$-realizable data distribution, and any $\varepsilon, \delta \in (0, 1)$, with probability at least $1 - \delta$ over $S \sim \mathcal{D}^T$ and $v_{1:T}$ where $T = O(\frac{d\log(kd)\log(1/\varepsilon)+\log|\mathcal{G}|\cdot(\log(1/\varepsilon)+k)+k\log(1/\delta)}{\varepsilon})$, the output $\widehat{h}$ satisfies $\mathcal{L}^{str}_{G^\star,\mathcal{D}}(\widehat{h}) \leq \varepsilon$.*

Proof sketch Finding a graph-hypothesis pair satisfying Eq (4) would guarantee small $\mathcal{L}^{\text{str}}_{G,\mathcal{D}}(h)$ by uniform convergence. According to the decomposition of strategic loss in Eq (2), it is sufficient to show the graph loss $\mathcal{L}_{\text{neighborhood}}(\widehat{G})$ of $\widehat{G}$ is low. Note that satisfying Eq (3) does not guarantee low graph loss. As aforementioned, a graph G whose arc set is a superset of $G^\star$'s arc set satisfies Eq (3), but its graph loss could be high. Hence, returning a graph-hypothesis pair that satisfies Eqs (3) and (4) is not enough and it is crucial to have empirical degree as a regularize.

Now we show the graph loss $\mathcal{L}_{\text{neighborhood}}(\widehat{G})$ is small. First, the 0-1 loss of the neighborhood prediction can be decomposed as

$$\mathbb{1}(N_{G^\star}(x) \neq N_G(x)) \leq |N_G(x) \setminus N_{G^\star}(x)| + |N_{G^\star}(x) \setminus N_G(x)|\,.$$

For any G satisfying Eq (3), G will have a small $\mathbb{E}_x\left[|N_{G^\star}(x) \setminus N_G(x)|\right]$. This is because

$$|N_{G^\star}(x) \setminus N_G(x)| = |N_{G^\star}(x)|\,\mathbb{P}_{v\sim\mathrm{Unif}(N_{G^\star}(x))}(v \notin N_G(x_t)) \leq k\mathbb{P}_{v\sim\mathrm{Unif}(N_{G^\star}(x))}(v \notin N_G(x))\,.$$

Note that $\mathbb{1}(v_t \notin N_G(x_t))$ is an unbiased estimate of $\mathbb{P}_{x,v\sim\mathrm{Unif}(N_{G^\star}(x))}(v \notin N_G(x))$. Hence, for any graph G satisfying Eq (3), $\mathbb{P}_{x,v\sim\mathrm{Unif}(N_{G^\star}(x))}(v \notin N_G(x))$ is small by uniform convergence and thus, $\mathbb{E}_x\left[|N_{G^\star}(x) \setminus N_G(x)|\right]$ is small.

Then we utilize the "minimal degree" to connect $\mathbb{E}_x\left[|N_G(x) \setminus N_{G^\star}(x)|\right]$ and $\mathbb{E}_x\left[|N_{G^\star}(x) \setminus N_G(x)|\right]$. For any graph with expected degree $\mathbb{E}_x\left[|N_G(x)|\right] \leq \mathbb{E}_x\left[|N_{G^\star}(x)|\right]$, subtracting $\mathbb{E}_x\left[|N_{G^\star}(x) \cap N_G(x)|\right]$ from both sides implies

$$\mathbb{E}_x\left[|N_G(x) \setminus N_{G^\star}(x)|\right] \leq \mathbb{E}_x\left[|N_{G^\star}(x) \setminus N_G(x)|\right], \tag{5}$$

By minimizing the empirical degree, we get $\mathbb{E}_x\left[|N_G(x) \setminus N_{G^\star}(x)|\right] \lesssim \mathbb{E}_x\left[|N_{G^\star}(x) \setminus N_G(x)|\right]$. The formal proof can be found in Appendix E. ■

So, our algorithm can find a good hypothesis and needs at most $k \log|\mathcal{G}|\varepsilon^{-1}$ more training data. However, as we show in the next theorem, the $\log|\mathcal{G}|$ factor is necessary.

Theorem 7 *There exists a hypothesis class $\mathcal{H}$ with $\mathrm{VCdim}(\mathcal{H}) = 1$ and a graph class $\mathcal{G}$ in which all graphs have a maximum out-degree of 1. For any algorithm $\mathcal{A}$, there exists a data distribution $\mathcal{D}$ such that for any i.i.d. sample of size T, there exists a graph $G \in \mathcal{G}$ consistent with the data history such that when $T \leq \frac{\log|\mathcal{G}|}{\varepsilon}$, we have $\mathcal{L}^{str}_{G,\mathcal{D}}(\widehat{h}) \geq \varepsilon$.*

Agnostic case Similar to the realizable case, the challenge lies in the fact that $\mathcal{L}_{\text{neighborhood}}(G)$ is not estimable. Moreover, in the agnostic case there might not exist a consistent graph satisfying Eqs (3) and (4) now. Instead, we construct the following alternative loss function as a proxy:

$$\mathcal{L}_{\text{proxy}}(G) := 2\mathbb{E}_x[P_{v\sim\mathrm{Unif}(N_{G^\star})(x)}(v \notin N_G(x))] + \frac{1}{k}\mathbb{E}[|N_G(x)|] - \frac{1}{k}\mathbb{E}[|N_{G^\star}(x)|]\,.$$

Note that in this proxy loss, $2 \cdot \mathbb{1}(v_t \notin N_G(x_t)) + \frac{1}{k}N_G(x_t)$ is an unbiased estimate of the first two terms, and the third term is a constant. Hence, we can find a graph with low proxy loss by minimizing its empirical value. We then show that this proxy loss is a good approximation of the graph loss when k is small.

Lemma 8 *Suppose that $G^\star$ and all graphs $G \in \mathcal{G}$ have maximum out-degree at most k. Then, we have*

$$\frac{1}{k}\mathcal{L}_{\textit{neighborhood}}(G) \leq \mathcal{L}_{\textit{proxy}}(G) \leq 3\mathcal{L}_{\textit{neighborhood}}(G)\,.$$

By minimizing the proxy loss $\mathcal{L}_{\text{proxy}}(\cdot)$, we can obtain an approximately optimal graph up to a multiplicative factor of k due to the gap between the proxy loss and the graph loss. See details and proofs in Appendix E. [3]

Application of the Graph Learning Algorithms in Multi-Label Learning Our graph learning algorithms have the potential to be of independent interest in various other learning problems, such as multi-label learning. To illustrate, let us consider scenarios like the recommender system, where we aim to recommend movies to users. In such cases, each user typically has multiple favorite movies, and our objective is to learn this set of favorite movies.

This multi-label learning problem can be effectively modeled as a bipartite graph denoted as $G^\star = (\mathcal{X}, \mathcal{Y}, \mathcal{E}^\star)$, where $\mathcal{X}$ represents the input space (in the context of the recommender system, it represents users), $\mathcal{Y}$ is the label space (in this case, it represents movies), and $\mathcal{E}^\star$ is a set of arcs. In this graph, the presence of an arc $(x, y) \in \mathcal{E}^\star$ implies that label y is associated with input x (e.g., user x liking movie y). Our primary goal here is to learn this graph, i.e., the arcs $\mathcal{E}^\star$. More formally, given a marginal data distribution $\mathcal{D}_\mathcal{X}$, our goal is to find a graph $\widehat{G}$ with minimum neighborhood prediction loss $\mathcal{L}_{\text{neighborhood}}(G) = \mathbb{P}_{x\sim\mathcal{D}_\mathcal{X}}(N_G(x) \neq N_{G^\star}(x))$. Suppose we are given a graph class $\mathcal{G}$. Then, our goal is to find the graph $\arg\min_{G\in\mathcal{G}} \mathcal{L}_{\text{neighborhood}}(G)$.

3. Different tie-breaking rules can impact the estimation factor of $\mathcal{L}_{\text{proxy}}(\cdot)$ and consequently the sample complexity.

However, in real-world scenarios, the recommender system cannot sample a user along with her favorite movie set. Instead, at each time t, the recommender recommends a set of movies h_t to the user and the user randomly clicks one of the movies in h_t that she likes (i.e., $v_t \in N_{G^\star}(x_t) \cap h_t$). Here we abuse the notation a bit by letting h_t represent the set of positive points labeled by this hypothesis. This setting perfectly fits into our unknown graph setting and our algorithms can find a good graph.

3.3. Online Learning

In this section, we mainly present our algorithmic result in the post-manipulation feedback setting. The post-manipulation setting is introduced by Ahmadi et al. (2023), where they exclusively consider the finite hypothesis class $\mathcal{H}$ and derive Stackelberg regret bounds based on the cardinality $|\mathcal{H}|$. In contrast, our work extends to dealing with infinite hypothesis classes and offers algorithms with guarantees dependent on the Littlestone dimension.

In the post-manipulation feedback setting, the learner observes either x_t or v_t after implementing h_t. It might seem plausible that observing x_t provides more information than observing v_t since we can compute v_t given x_t. However, we show that there is no gap between these two cases. We provide an algorithm based on the "harder" case of observing v_t and a lower bound based on the "easier" case of observing x_t and show that the bounds are tight.

Our algorithm is based on a reduction from strategic online learning to standard online learning. More specifically, given any standard online learning algorithm $\mathcal{A}$, we construct a weighted expert set E which is initialized to be $\{\mathcal{A}\}$ with weight $w_{\mathcal{A}} = 1$. At each round, we predict x as 1 iff the weight of experts predicting x as 1 is higher than $1/2(k+1)$ of the total weight. Once a mistake is made at the observed node v_t, we differentiate between the two types of mistakes.

- False positive: Namely, the labeling of x_t induced by h_t is 1 but the true label is $y_t = 0$. Since the true label is 0, the entire neighborhood of x_t should be labeled as 0 by the target hypothesis, including the observed feature v_t. Hence, we proceed by updating all experts incorrectly predicting v_t as 1 with the example $(v_t, 0)$ and halve their weights.
- False negative: Namely, the labeling of x_t induced by h_t is 0, but the true label is $y_t = 1$. Thus, $h^\star$ must label some neighbor of x_t, say $x^\star$, by 1. For any expert A labeling the entire neighborhood by 0, they make a mistake. But we do not know which neighbor should be labeled as 1. So we split A into $|N_{G^\star}[x_t]|$ number of experts, each of which is fed by one neighbor $x \in N_{G^\star}[x_t]$ labeled by 1. Then at least one of the new experts is fed with the correct neighbor $(x^\star, 1)$. We also split the weight equally and halve it.

The pseudo-code is included in Algorithm 1. Next, we derive a mistake bound for Algorithm 1 that depends on the mistake bound of the standard online learning algorithm, and by plugging in the Standard Optimal Algorithm (SOA), we derive a mistake bound of $O(k \log k \cdot M)$.

Theorem 9 *For any hypothesis class $\mathcal{H}$, graph $G^\star$ with maximum out-degree k, and a standard online learning algorithm with mistake bound M for $\mathcal{H}$, for any realizable sequence, we can achieve mistake bound of $O(k \log k \cdot M)$ by Algorithm 1. By letting the SOA algorithm by Littlestone (1988) as the input standard online algorithm $\mathcal{A}$, Red2Online-PMF(SOA) makes at most $O(k \log k \cdot \mathrm{Ldim}(\mathcal{H}))$ mistakes.*

Compared to the fully informative setting (in which the dependency on k of the optimal mistake bound is $\Theta(\log k)$), the dependency of the number of mistakes made by our Algorithm 1 on k is

Algorithm 1 Red2Online-PMF: Reduction to online learning for post-manipulation feedback

Input: a standard online learning algorithm $\mathcal{A}$, maximum out-degree upper bound k
Initialization: expert set $E = \{\mathcal{A}\}$ and weight $w_{\mathcal{A}} = 1$
for $t = 1, 2, \ldots$ **do**
  Prediction: at each point x, $h_t(x) = 1$ if $\sum_{A \in E: A(x)=1} w_A \geq \frac{\sum_{A \in E} w_{\mathcal{A}_H}}{2(k+1)}$
  Update: *//when we make a mistake at the observed node* v_t
  if $y_t = 0$ **then**
    for all $A \in E$ satisfying $A(v_t) = 1$, feed A with $(v_t, 0)$ and update $w_A \leftarrow \frac{1}{2} w_A$
  else if $y_t = 1$ **then**
    for all $A \in E$ satisfying $\forall x \in N_{G^\star}[v_t], A(x) = 0$
    for all $x \in N_{G^\star}[v_t]$, by feeding $(x, 1)$ to A, we can obtain a new expert $A(x, 1)$
    remove A from E and add $\{A(x,1) | x \in N_{G^\star}[x_t]\}$ to E with weights $w_{A(x,1)} = \frac{w_A}{2|N_{G^\star}[x_t]|}$
  end if
end for

$O(k \log k)$. However, as we show in the next theorem, no deterministic algorithm can make fewer than $\Omega(\mathrm{Ldim}(\mathcal{H}) \cdot k)$ mistakes. Hence, our algorithm is nearly optimal (among all deterministic algorithms) up to a logarithmic factor in k.

Theorem 10 *For any $k, d > 0$, there exists a graph $G^\star$ with maximum out-degree k and a hypothesis class $\mathcal{H}$ with $\mathrm{Ldim}(\mathcal{H}) = d$ such that for any deterministic algorithm, there exists a realizable sequence for which the algorithm will make at least $d(k-1)$ mistakes.*

Remark 11 *Ahmadi et al. (2023) leave the gap between the upper bound of $O(k \log(|\mathcal{H}|))$ and the lower bound of $\Omega(k)$ as an open question. Theorems 9 and 10 closes this gap up to a logarithmic factor in k.*

Extension to Unknown Graph Setting In the unknown graph setting, we do not observe $N_{G^\star}(x_t)$ or $N_{G^\star}(v_t)$ anymore. We then design an algorithm by running an instance of Algorithm 1 over the majority vote of the graphs. More specifically, after observing x_t, let $\widetilde{N}(x_t)$ denote the set of nodes x satisfying that (x_t, x) is an arc in at least half of the consistent graphs.

- Reduce inconsistent graphs: For any $x \notin \widetilde{N}(x_t)$, which means (x_t, x) is an arc in at most half of the current consistent graphs, we predict $h_t(x) = 1$. If we make a mistake at such a $v_t = x$, it implies that (x_t, x) is an arc in the true graph, so we can eliminate at least half of the graphs.
- Approximate $N_{G^\star}(x_t)$ by $\widetilde{N}(x_t)$: For any $x \in \widetilde{N}(x_t)$, we predict $h_t(x)$ by following Algorithm 1. We update Algorithm 1 in the same way when we make a false positive mistake at v_t as it does not require the neighborhood information. However, when we make a false negative mistake at v_t, it implies that $v_t = x_t$ and the entire neighborhood $N_{G^\star}(x_t)$ is labeled as 0. We then utilize $\widetilde{N}(x_t)$ as the neighborhood feedback to update Algorithm 1. Note that this majority vote neighborhood $\widetilde{N}(x_t)$ must be a superset of $N_{G^\star}(v_t) = N_{G^\star}(x_t)$ since the entire $N_{G^\star}(v_t)$ is labeled as 0 by h_t and every $x \notin \widetilde{N}(x_t)$ is labeled as 1. Moreover, the size of $\widetilde{N}(x_t)$ will not be greater than $2k$ since all graphs have maximum degree at most k.

We defer the formal description of this algorithm to Algorithm 3 that appears in Appendix E. In the following, we derive an upper bound over the number of mistakes for this algorithm.

Theorem 12 *For any hypothesis class $\mathcal{H}$, the underlying true graph $G^\star$ with maximum out-degree at most k, finite graph class $\mathcal{G}$ in which all graphs have maximum out-degree at most k, for any realizable sequence, Algorithm 3 will make at most $O(k\log(k)\cdot \mathrm{Ldim}(\mathcal{H}) + \log|\mathcal{G}|)$ mistakes.*

So when we have no access to the neighborhood information, we will suffer $\log|\mathcal{G}|$ more mistakes. However, we show that this $\log|\mathcal{G}|$ term is necessary for all deterministic algorithms.

Theorem 13 *For any $n \in \mathbb{N}$, there exists a hypothesis class $\mathcal{H}$ with $\mathrm{Ldim}(\mathcal{H}) = 1$, a graph class $\mathcal{G}$ satisfying that all graphs in $\mathcal{G}$ have maximum out-degree at most 1 and $|\mathcal{G}| = n$ such that for any deterministic algorithm, there exists a graph $G^\star \in \mathcal{G}$ and a realizable sequence for which the algorithm will make at least $\frac{\log n}{\log\log n}$ mistakes.*

Finally, as mentioned in Section 2.3, there are several options for the feedback, including observing x_t beforehand followed by v_t afterward, observing (x_t, v_t) afterward, and observing either x_t or v_t afterward, arranged in order of increasing difficulty. We mainly focus on the simplest one, that is, observing x_t beforehand followed by v_t afterward. We show that even in the second simplest case of observing (x_t, v_t) afterward, any deterministic algorithm suffers mistake bound linear in $|\mathcal{G}|$ and $|\mathcal{H}|$.

Proposition 14 *When (x_t, v_t) is observed afterward, for any $n \in \mathbb{N}$, there exists a class $\mathcal{G}$ of graphs of degree 2 and a hypothesis class $\mathcal{H}$ with $|\mathcal{G}| = |\mathcal{H}| = n$ such that for any deterministic algorithm, there exists a graph $G^\star \in \mathcal{G}$ and a realizable sequence for which the algorithm will make at least $n-1$ mistakes.*

4. Discussion

In this work, we have investigated the learnability gaps of strategic classification in both PAC and online settings. We demonstrate that learnability implies strategic learnability when the manipulation graph has a finite maximum out-degree, $k < \infty$. Additionally, strategic manipulation does indeed render learning more challenging. In scenarios where we consider both the true graph information and pre-manipulation feedback, manipulation only results in an increase in both sample complexity and regret by a $\log k$ multiplicative factor. However, in cases where we only have post-manipulation feedback, the dependence on k in the sample complexity remains $\log k$, but increases to k in the regret. When the true graph is unknown and only a graph class $\mathcal{G}$ is available, there is an additional increase of a $\log|\mathcal{G}|$ additive factor in both sample complexity and regret. Our algorithms for learning an unknown graph are of independent interest to the multi-label learning problem.

Several questions remain open. The first pertains to agnostic online learning. We have explored extending some realizable online learning results to the agnostic case in Appendix F through a standard reduction technique. However, it is unclear how to address this issue, especially in the case where there is no perfect graph in the class. Additionally, several other technical questions remain open, such as improving the lower bounds presented in the work and eliminating the multiplicative factor in Theorem 28.

Another interesting open question is gaps in computational complexity. In the fully informative PAC setting, our algorithm minimizes the empirical strategic loss. Computing $\overline{h}_{G^\star}(x)$ for each x requires $\mathcal{O}(k)$ time, by checking if there is any neighbor of x that is labeled as positive by h. Hence, minimizing the empirical strategic loss would take $O(k)$ times more computational complexity than minimizing the standard $0-1$ loss. In the unknown graph PAC learning setting, we optimize over all graph-hypothesis pairs, which makes the computational complexity scale linearly with the cardinality of the graph class.

Acknowledgements

Lee Cohen is supported by the Simons Foundation Collaboration on the Theory of Algorithmic Fairness, the Sloan Foundation Grant 2020-13941, and the Simons Foundation investigators award 689988.
Yishay Mansour was supported by funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 882396), by the Israel Science Foundation, the Yandex Initiative for Machine Learning at Tel Aviv University and a grant from the Tel Aviv University Center for AI and Data Science (TAD).
Shay Moran is a Robert J. Shillman Fellow; he acknowledges support by ISF grant 1225/20, by BSF grant 2018385, by an Azrieli Faculty Fellowship, by Israel PBC-VATAT, by the Technion Center for Machine Learning and Intelligent Systems (MLIS), and by the the European Union (ERC, GENERALIZATION, 101039692). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.
Han Shao was supported in part by the National Science Foundation under grants CCF-2212968 and ECCS-2216899, by the Simons Foundation under the Simons Collaboration on the Theory of Algorithmic Fairness, and by the Defense Advanced Research Projects Agency under cooperative agreement HR00112020003. The views expressed in this work do not necessarily reflect the position or the policy of the Government and no official endorsement should be inferred.

References

Saba Ahmadi, Hedyeh Beyhaghi, Avrim Blum, and Keziah Naggita. The strategic perceptron. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, pages 6–25, 2021.

Saba Ahmadi, Avrim Blum, and Kunhe Yang. Fundamental bounds on online strategic classification. In Kevin Leyton-Brown, Jason D. Hartline, and Larry Samuelson, editors, *Proceedings of the 24th ACM Conference on Economics and Computation, EC 2023, London, United Kingdom, July 9-12, 2023*, pages 22–58. ACM, 2023.

Yahav Bechavod, Katrina Ligett, Steven Wu, and Juba Ziani. Gaming helps! learning from strategic interactions in natural dynamics. In *International Conference on Artificial Intelligence and Statistics*, pages 1234–1242. PMLR, 2021.

Yahav Bechavod, Chara Podimata, Steven Wu, and Juba Ziani. Information discrepancy in strategic learning. In *International Conference on Machine Learning*, pages 1691–1715. PMLR, 2022.

Shai Ben-David, Dávid Pál, and Shai Shalev-Shwartz. Agnostic online learning. In *COLT*, volume 3, page 1, 2009.

Mark Braverman and Sumegha Garg. The role of randomness and noise in strategic classification. In *Foundations of Responsible Computing (FORC)*, volume 156 of *LIPIcs*, pages 9:1–9:20, 2020.

Michael Brückner and Tobias Scheffer. Stackelberg games for adversarial prediction problems. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 547–555, 2011.

Yiling Chen, Yang Liu, and Chara Podimata. Learning strategy-aware linear classifiers. *Advances in Neural Information Processing Systems*, 33:15265–15276, 2020.

Lee Cohen, Saeed Sharifi Malvajerdi, Kevin Stangl, Ali Vakilian, and Juba Ziani. Sequential strategic screening. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Nilesh Dalvi, Pedro Domingos, Sumit Sanghai, and Deepak Verma. Adversarial classification. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 99–108, 2004.

Jinshuo Dong, Aaron Roth, Zachary Schutzman, Bo Waggoner, and Zhiwei Steven Wu. Strategic classification from revealed preferences. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 55–70, 2018.

Yuval Filmus, Steve Hanneke, Idan Mehalel, and Shay Moran. Optimal prediction using expert advice and randomized littlestone dimension. In Gergely Neu and Lorenzo Rosasco, editors, *The Thirty Sixth Annual Conference on Learning Theory, COLT 2023, 12-15 July 2023, Bangalore, India*, volume 195 of *Proceedings of Machine Learning Research*, pages 773–836. PMLR, 2023.

Ganesh Ghalme, Vineet Nair, Itay Eilat, Inbal Talgam-Cohen, and Nir Rosenfeld. Strategic classification in the dark. In *International Conference on Machine Learning*, pages 3672–3681. PMLR, 2021.

Nika Haghtalab, Nicole Immorlica, Brendan Lucier, and Jack Z. Wang. Maximizing welfare with incentive-aware evaluation mechanisms. 2020.

Nika Haghtalab, Thodoris Lykouris, Sloan Nietert, and Alexander Wei. Learning in stackelberg games with non-myopic agents. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, pages 917–918, 2022.

Moritz Hardt, Nimrod Megiddo, Christos Papadimitriou, and Mary Wootters. Strategic classification. In *Proceedings of the 2016 ACM conference on innovations in theoretical computer science*, pages 111–122, 2016.

Keegan Harris, Hoda Heidari, and Steven Z Wu. Stateful strategic regression. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:28728–28741, 2021.

Lily Hu, Nicole Immorlica, and Jennifer Wortman Vaughan. The disparate effects of strategic manipulation. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 259–268, 2019.

Meena Jagadeesan, Celestine Mendler-Dünner, and Moritz Hardt. Alternative microfoundations for strategic classification. In *International Conference on Machine Learning*, pages 4687–4697. PMLR, 2021.

Moein Khajehnejad, Behzad Tabibian, Bernhard Schölkopf, Adish Singla, and Manuel Gomez-Rodriguez. Optimal decision making under strategic behavior. *arXiv preprint arXiv:1905.09239*, 2019.

Tosca Lechner and Ruth Urner. Learning losses for strategic classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 7337–7344, 2022.

Tosca Lechner, Ruth Urner, and Shai Ben-David. Strategic classification with unknown user manipulations. In *International Conference on Machine Learning*, pages 18714–18732. PMLR, 2023.

Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine learning*, 2(4):285–318, 1988.

Lydia T Liu, Ashia Wilson, Nika Haghtalab, Adam Tauman Kalai, Christian Borgs, and Jennifer Chayes. The disparate equilibria of algorithmic decision making when individuals invest rationally. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 381–391, 2020.

Andreas Maurer and Massimiliano Pontil. Empirical bernstein bounds and sample variance penalization. *stat*, 1050:21, 2009.

John Miller, Smitha Milli, and Moritz Hardt. Strategic classification is causal modeling in disguise. In *International Conference on Machine Learning*, pages 6917–6926. PMLR, 2020.

Smitha Milli, John Miller, Anca D Dragan, and Moritz Hardt. The social cost of strategic classification. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 230–239, 2019.

Omar Montasser, Steve Hanneke, and Nathan Srebro. Vc classes are adversarially robustly learnable, but only improperly. In *Conference on Learning Theory*, pages 2512–2530. PMLR, 2019.

Han Shao, Avrim Blum, and Omar Montasser. Strategic classification under unknown personalized manipulation. *arXiv preprint arXiv:2305.16501*, 2023.

Yonadav Shavit, Benjamin Edelman, and Brian Axelrod. Causal strategic linear regression. In *International Conference on Machine Learning (ICML)*, pages 8676–8686, 2020.

Ravi Sundaram, Anil Vullikanti, Haifeng Xu, and Fan Yao. Pac-learning for strategic classification. In *International Conference on Machine Learning*, pages 9978–9988. PMLR, 2021.

Wei Tang, Chien-Ju Ho, and Yang Liu. Linear models are robust optimal under strategic behavior. In *International Conference on Artificial Intelligence and Statistics*, pages 2584–2592. PMLR, 2021.

V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971.

V. Vapnik and A. Chervonenkis. *Theory of Pattern Recognition*. Nauka, Moscow, 1974.

Hanrui Zhang and Vincent Conitzer. Incentive-aware pac learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 5797–5804, 2021.

Appendix A. Related Work

Our work aligns with strategic classification, a recent line of research that explores the influence of strategic behavior on decision making and machine learning algorithms. This line of study is extensive and encompasses various settings, including distributional vs. online, different knowledge of manipulation structure, pre-manipulation vs. post-manipulation feedback, and the goals of gaming vs. improving. However, none of the prior work has addressed the learnability gaps between standard learning and strategic learning studied in this work.

Strategic classification was first studied in a distributional model by Hardt et al. (2016), assuming that agents manipulate by best responding with respect to a uniform cost function known to the learner. Building on the framework of Hardt et al. (2016), Lechner and Urner (2022); Hu et al. (2019); Milli et al. (2019); Sundaram et al. (2021); Zhang and Conitzer (2021) studied the distributional learning problem with the goal of returning a classifier with small strategic population loss, and all of them assumed that the manipulations are predefined and known to the learner, either by a cost function or a predefined manipulation graph. Then Lechner et al. (2023) and Shao et al. (2023) studied the unknown manipulation structure. However, Lechner et al. (2023) considered a different loss function that penalizes manipulation even when the prediction at the manipulated feature is correct, and Shao et al. (2023) provided guarantees only for finite hypothesis classes.

For the online setting, Chen et al. (2020); Dong et al. (2018); Ahmadi et al. (2021, 2023); Shao et al. (2023) studied Stackelberg regret in the online setting learning. However, Dong et al. (2018) studied Stackelberg regret in linear classification and focused on finding appropriate conditions of the cost function to achieve sub-linear Stackelberg regret. Chen et al. (2020) studied linear classification under uniform ball manipulations and Ahmadi et al. (2021, 2023) studied Stackelberg regret under a predefined and known manipulation. Shao et al. (2023) studied the mistake bound under unknown manipulation structures, but only provided results for finite hypothesis classes.

Models of manipulation structure Hardt et al. (2016) modeled manipulation structure by a cost function, where agents manipulate to maximize their utility function, which is the reward of obtaining a positive prediction minus the cost of manipulation. It then naturally leads to two modeling ways of manipulation structure: The first is a geometric model (e.g., Dong et al. (2018); Sundaram et al. (2021); Chen et al. (2020); Ghalme et al. (2021); Haghtalab et al. (2020); Ahmadi et al. (2021); Shao et al. (2023)), where each agent is able to manipulate within a bounded-radius ball in some known metric space. Other works (e.g., Zhang and Conitzer (2021); Lechner and Urner (2022); Lechner et al. (2023); Ahmadi et al. (2023)) modeled feasible manipulations using a manipulation graph, where each agent is represented by a node x, and an arc from x to x' indicates that an agent with the feature vector x may adjust their features to x' if it increases their likelihood of receiving a positive classification. However, these two ways of modeling are actually equivalent as we can define a manipulation graph given the manipulation balls and we can define ball manipulations w.r.t. the graph distance on a predefined known graph.

Different information settings All PAC learning scenarios involving either known manipulation structures or known local neighborhoods (Hardt et al., 2016; Lechner and Urner, 2022; Sundaram et al., 2021; Zhang and Conitzer, 2021; Hu et al., 2019; Milli et al., 2019) can be encompassed within our fully informative setting. We remark that the work of Lechner and Urner (2022) considered the task of learning the manipulation graph, but assumed that each sample is of the form $(x,$ neighborhood of $x)$, which is closer to our fully informative setting as our results also only require

the neighborhood information. Ahmadi et al. (2021) examined online learning within our fully informative setting, while Ahmadi et al. (2023) explored online learning within our post-manipulation feedback setting. Regarding unknown graph settings, Lechner et al. (2023) and Shao et al. (2023) have made contributions. However, Lechner et al. (2023) considered a different loss function that penalizes manipulation even when the prediction at the manipulated feature is correct and Shao et al. (2023) provided guarantees solely based on the cardinality of the hypothesis class. We primarily study strategic classification in online and distributional settings in various restrictive feedback settings.

Other works. Other works explored strategic classification in various contexts, e.g.: noisy classifiers (Braverman and Garg, 2020), causality (Miller et al., 2020; Shavit et al., 2020), noisy agents (Jagadeesan et al., 2021), decision making (Khajehnejad et al., 2019; Tang et al., 2021), sequential setting (Cohen et al., 2023), social objectives (e.g., social burden) (Hu et al., 2019; Liu et al., 2020; Milli et al., 2019; Lechner et al., 2023), Strategic interactions in the regression (Bechavod et al., 2021), non-myopic agents (Haghtalab et al., 2022; Harris et al., 2021), and lack of transparency in decision rules (Bechavod et al., 2022). Beyond strategic classification, there is a more general research area of learning using data from strategic sources, such as a single data generation player who manipulates the data distribution (Brückner and Scheffer, 2011; Dalvi et al., 2004). Adversarial perturbations can be viewed as another type of strategic source (Montasser et al., 2019).

Appendix B. Learning Complexity- Background

Here we recall the definitions of the Vapnik–Chervonenkis dimension (VCdim) (Vapnik and Chervonenkis, 1974), and Littlestone dimension (Ldim) (Littlestone, 1988).

Definition 15 (VC-dimension) *Let $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ be a hypothesis class. A subset $S = \{x_1, ..., x_{|S|}\} \subseteq \mathcal{X}$ is shattered by $\mathcal{H}$ if: $\left|\left\{\left(h(x_1), ..., h(x_{|S|})\right) : h \in \mathcal{H}\right\}\right| = 2^{|S|}$. The VC-dimension of $\mathcal{H}$, denoted $VCdim(\mathcal{H})$, is the maximal cardinality of a subset $S \subseteq \mathcal{X}$ shattered by $\mathcal{H}$.*

Definition 16 ($\mathcal{H}$-shattered tree) *Let $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ be a hypothesis class and let $d \in \mathbb{N}$. A sequence $(x_1, \ldots, x_{2^d-1}) \in \mathcal{X}^{2^d-1}$ is an $\mathcal{H}$-*shattered tree *of depth d if, for every labeling $(y_1, \ldots, y_d) \in \mathcal{Y}^d$, there exists $h \in \mathcal{H}$ such that for all $i \in [d]$ we have that $h(x_{j_i}) = y_i$, where $j_i = 2^{i-1} + \sum_{k=1}^{i-1} y_k 2^{i-1-k}$.*

Definition 17 (Littlestone dimension) *The* Littlestone dimension *of a hypothesis class $\mathcal{H}$ is the maximal depth of a tree shattered by $\mathcal{H}$.*

Appendix C. Fully Informative Setting

In this setting, the manipulation graph $G^\star$ is known and the learner observes x_t before implementing h_t. We remark that though the learner has knowledge of the entire graph $G^\star$ in this model, the algorithms in this section only require access to the out-neighborhood $N_{G^\star}(x_t)$ of x_t.

C.1. PAC Learning

In the PAC setting, no matter what learner h_t implemented in each round, we obtain T i.i.d. examples $S = ((x_1, y_1), \ldots, (x_T, y_T))$ after T rounds of interaction. Then, by running ERM over $\mathcal{H}$ w.r.t. the

strategic loss, we can obtain a hypothesis $\widehat{h}$ by minimizing the empirical strategic loss,

$$\widehat{h} = \arg\min_{h \in \mathcal{H}} \sum_{t=1}^{T} \ell^{\text{str}}_{G^\star}(h, (x_t, y_t)) \,. \tag{1}$$

Since $\ell^{\text{str}}_{G^\star}(h, (x, y))$ (see Def 3) only depends on the graph $G^\star$ through the neighborhood $N_{G^\star}(x)$, we can optimize Eq (1) given only $\{N_{G^\star}(x_t) | t \in [T]\}$ instead of the entire graph $G^\star$. Readers familiar with VC theory might notice that the sample complexity can be bounded by the VC dimension of the $(G^\star, \mathcal{H})$-induced labeling class $\overline{\mathcal{H}}_{G^\star}$, $\text{VCdim}(\overline{\mathcal{H}}_{G^\star})$. This is correct and has been shown in Sundaram et al. (2021). However, since our goal is to connect the PAC learnability in the strategic setting and standard setting, we provide sample complexity guarantees dependent on the VC dimension of $\mathcal{H}$. In fact, our sample complexity results are proved by providing upper and lower bounds of $\text{VCdim}(\overline{\mathcal{H}}_{G^\star})$ using $\text{VCdim}(\mathcal{H})$.

Theorem 4 *For any hypothesis class $\mathcal{H}$ and graph $G^\star$, we have $\text{VCdim}(\overline{\mathcal{H}}_{G^\star}) \leq d \log(kd)$ where $d = \text{VCdim}(\mathcal{H})$. Moreover, for any $d, k > 0$, there exists a class $\mathcal{H}$ with $\text{VCdim}(\mathcal{H}) = d$ and a graph $G^\star$ with maximum out-degree k, such that $\text{VCdim}(\overline{\mathcal{H}}_{G^\star}) \geq d \log k$.*

Proof For any $x \in \mathcal{X}$, if two hypotheses h, h' label its closed neighborhood $N_{G^\star}[x] = \{x\} \cup N_{G^\star}(x)$ in the same way, i.e., $h(z) = h'(z), \forall z \in N_{G^\star}[x]$, then the labeling of x induced by $(G^\star, h)$ and $(G^\star, h')$ are the same, i.e., $\overline{h}_{G^\star}(x) = \overline{h'}_{G^\star}(x)$. Hence, for any n points $X = \{x_1, x_2, \ldots, x_n\} \subset \mathcal{X}$, the labeling of X induced by $(G^\star, h)$ is determined by the prediction of h at their closed neighborhoods, $h(\cup_{x \in X} N_{G^\star}[x])$. Since $|\cup_{x \in X} N_{G^\star}[x]| \leq n(k+1)$, there are at most $O((nk)^d)$ different ways of implementation since $\text{VCdim}(\mathcal{H}) = d$ by Sauer-Shelah-Perels Lemma. If n points can be shattered by $\overline{\mathcal{H}}_{G^\star}$, then the number of their realized induced labeling ways is 2^n, which should be bounded by the number of possible implementations of their neighborhoods, which is upper bounded by $O((nk)^d)$. Hence, $2^n \leq O((nk)^d)$ and thus, we have $n \leq O(d \log(kd))$. Hence, we have $\text{VCdim}(\overline{\mathcal{H}}_{G^\star}) \leq O(d \log(kd))$.
For the lower bound, let us start with the simplest case of $d = 1$. Consider $(k+1) \log k$ nodes $\{x_{i,j} | i = 1, \ldots, \log k, j = 0, \ldots, k\}$, where $x_{i,0}$ is connected to $\{x_{i,j} | j = 1, \ldots, k\}$ for all $i \in [\log k]$. For $j = 1, \ldots, k$, let $\text{bin}(j)$ denote the $\log k$-bit-binary representation of j. Let hypothesis h_j be the hypothesis that labels $x_{i,j}$ by $\text{bin}(j)_i$ and all the other nodes by 0. Let $\mathcal{H} = \{h_j | j \in [k]\}$. Note that $\text{VCdim}(\mathcal{H}) = 1$. This is because for any two points $x_{i,j}$ and $x_{i',j'}$, if $j \neq j'$, no hypothesis will label them as positive simultaneously; if $j = j'$, they can only be labeled as $(\text{bin}(j)_i, \text{bin}(j)_{i'})$ by h_j and $(0, 0)$ by all the other hypotheses. However, $\overline{\mathcal{H}}_{G^\star}$ can shatter $\{x_{1,0}, x_{2,0}, \ldots, x_{\log k, 0}\}$ since h_j's labeling over $\{x_{1,0}, x_{2,0}, \ldots, x_{\log k, 0}\}$ is $\text{bin}(j)$. We can extend the example to the case of $\text{VCdim}\mathcal{H} = d$ by making d independent copies of $(\mathcal{X}, \mathcal{H}, G^\star)$. ■

Following directly Theorem 4 through the tools from VC theory (Vapnik and Chervonenkis, 1971, 1974), we prove the following sample complexity upper and lower bounds.

Theorem 18 *For any hypothesis class $\mathcal{H}$ with $\text{VCdim}(\mathcal{H}) = d$, graph $G^\star$ with maximum out-degree k, any data distribution $\mathcal{D}$ and any $\varepsilon, \delta \in (0, 1)$, with probability at least $1 - \delta$ over $S \sim \mathcal{D}^T$ where $T = O(\frac{d \log(kd) + \log(1/\delta)}{\varepsilon^2})$, the output $\widehat{h}$ satisfies $\mathcal{L}^{str}_{G^\star, \mathcal{D}}(\widehat{h}) \leq \min_{h^\star \in \mathcal{H}} \mathcal{L}^{str}_{G^\star, \mathcal{D}}(h^\star) + \varepsilon$.*
In the realizable case, the output $\widehat{h}$ satisfies $\mathcal{L}^{str}_{G^\star, \mathcal{D}}(\widehat{h}) \leq \varepsilon$ when $T = O(\frac{d \log(kd) \log(1/\varepsilon) + \log(1/\delta)}{\varepsilon})$.

Algorithm 2 Red2Online-FI: Reduction to online learning in the fully informative setting

Input: a standard online learning algorithm $\mathcal{A}$
Initialization: let the expert set $E = \{\mathcal{A}\}$ and weight $w_{\mathcal{A}}$ =1
for $t = 1, 2, \ldots$ **do**
  Prediction: after observing x_t
  if $\sum_{A \in E: \exists x \in N_{G^\star}[x_t], A(x)=1} w_A \geq \sum_{A \in E} w_A/2$ **then**
    let $h_t(x_t) = 1$ and $h_t(x) = 0$ for all $x \neq x_t$
  else
    let $h_t = \mathbb{1}_\emptyset$ be the all-negative function
  end if
  Update: when we make a mistake
  if $y_t = 0$ **then**
    for all $A \in E$ satisfying $\exists x \in N_{G^\star}[x_t], A(x) = 1$, feed A with $(x, 0)$ and update $w_A \leftarrow \frac{1}{2} w_A$
  else if $y_t = 1$ **then**
    for all $A \in E$ satisfying $\forall x \in N_{G^\star}[x_t], A(x) = 0$
    for all $x \in N_{G^\star}[x_t]$, by feeding $(x, 1)$ to A, we can obtain a new expert $A(x, 1)$
    remove A from E and add $\{A(x, 1) | x \in N_{G^\star}[x_t]\}$ to E with weights $w_{A(x,1)} = \frac{w_A}{2|N_{G^\star}[x_t]|}$
  end if
end for

Theorem 19 *For any $k, d > 0$, there exists a graph $G^\star$ with maximum out-degree k and a hypothesis class $\mathcal{H}$ with VC dimension d such that for any algorithm, there exists a distribution $\mathcal{D}$ for which achieving $\mathcal{L}^{str}_{G^\star,\mathcal{D}}(\widehat{h}) \leq \min_{h^\star \in \mathcal{H}} \mathcal{L}^{str}_{G^\star,\mathcal{D}}(h^\star) + \varepsilon$ with probability $1 - \delta$ requires sample complexity $\Omega(\frac{d \log k + \log(1/\delta)}{\varepsilon^2})$. In the realizable case, it requires sample complexity $\Omega(\frac{d \log k + \log(1/\delta)}{\varepsilon})$.*

C.2. Online Learning

Recall that in the online setting, the agents are generated by an adversary, and the learner's goal is to minimize the Stackelberg regret (which is the mistake bound in the realizable case). Similar to Algorithm 1, we will show that any learning algorithm in the standard online setting can be converted to an algorithm in the strategic online setting, with the mistake bound increased by a logarithmic multiplicative factor of k. The main difference from Algorithm 1 is that now we know x_t before implementing the hypothesis, we do not need to reduce the weight of predicting positive by a factor $1/k$ as we did in line 4 of Algorithm 1.

Theorem 20 *For any hypothesis class $\mathcal{H}$, graph $G^\star$ with maximum out-degree k, and a standard online learning algorithm with mistake bound M for $\mathcal{H}$, for any realizable sequence, we can achieve mistake bound of $O(M \log k)$ by Algorithm 2. By letting the SOA algorithm by Littlestone (1988) as the input standard online algorithm $\mathcal{A}$, Red2Online-FI(SOA) makes at most $O(\mathrm{Ldim}(\mathcal{H}) \log k)$ mistakes.*

Proof Let $\mathcal{A}$ be an online learning algorithm with mistake bound M. Let W_t denote the total weight of experts at the beginning of round t. When we make a mistake at round t, there are two cases.

- False positive: It means that the induced labeling of x_t is 1 but the true label is $y_t = 0$. By our prediction rule, we have $\sum_{\mathcal{A}_H \in E: \exists x \in N_{G^\star}[x_t], \mathcal{A}_H(x)=1} w_{\mathcal{A}_H} \geq \frac{W_t}{2}$, and thus, we will reduce the total weight by at least $\frac{W_t}{4}$. Hence, we have $W_{t+1} \leq (1 - \frac{1}{4})W_t$.

- False negative: It means that the induced labeling of x_t is 0, but the true label is $y_t = 1$. This implies that our learner h_t labeled the entire neighborhood of x_t with 0, but $h^\star$ labels some point in the neighborhood by 1. By our prediction rule, we have $\sum_{\mathcal{A}_H \in E: \forall x \in N_{G^\star}[x_t] \mathcal{A}_H(x)=0} w_{\mathcal{A}_H} \geq \frac{W_t}{2}$. We split $\mathcal{A}_H$ into $\{\mathcal{A}_{H_{x,+1}} | x \in N_{G^\star}[x_t]\}$ and split the weight $w_{\mathcal{A}_H}$ equally and multiply by $\frac{1}{2}$. Thus, we have $W_{t+1} \leq (1 - \frac{1}{4})W_t$.

Hence, whenever we make a mistake the total weight is reduced by a multiplicative factor of $\frac{1}{4}$. Let N denote the total number of mistakes by Red2Online-FI($\mathcal{A}$). Then we have total weight $\leq (1 - 1/4)^N$. On the other hand, note that there is always an expert, whose hypothesis class contains $h^\star$, being fed with examples $(\pi_{G^\star,h^\star}(x_t), y_t)$. At each mistake round t, if the expert (containing $h^\star$) makes a false positive mistake, its weight is reduced by half. If the expert made a false negative mistake, it is split into a few experts, one of which is fed by $(\pi_{G^\star,h^\star}(x_t), y_t)$. This specific new expert will contain $h^\star$ and the weight is reduced by at most $\frac{1}{2(k+1)}$. Thus, since this expert will make at most M mistakes, the weight of such an expert will be at least $(\frac{1}{2(k+1)})^M$.
Thus we have $(1 - 1/4)^N \geq 1/(2(k+1))^M$, which yields the bound $N \leq 4M \ln(2(k+1))$. ■

Next, we derive a lower bound for the number of mistakes.

Theorem 21 *For any $k, d > 0$, there exists a graph $G^\star$ with maximum out-degree k and a hypothesis class $\mathcal{H}$ with Littlestone dimension $\mathrm{Ldim}(\mathcal{H}) = d$ such that for any algorithm, there exists a realizable sequence for which the algorithm will make at least $d \log k$ mistakes.*

Proof Consider the same hypothesis class $\mathcal{H}$ and the graph $G^\star$ as Theorem 19. Again, let us start with the simplest case of $d = 1$. Consider $(k+1)\log k$ nodes $\{x_{i,j} | i = 1, \ldots, \log k, j = 0, \ldots, k\}$, where $x_{i,0}$ is connected to $\{x_{i,j} | j = 1, \ldots, k\}$ for all $i \in [\log k]$. For $j = 1, \ldots, k$, let $\mathrm{bin}(j)$ denote the $\log k$-bit-binary representation of j. Let hypothesis h_j labels $x_{i,j}$ by $\mathrm{bin}(j)_i$ and all the other nodes by 0. Let $\mathcal{H} = \{h_j | j \in [k]\}$. The Littlestone dimension $\mathrm{Ldim}(\mathcal{H}) = 1$ because there is an easy online learning in the standard setting by predicting the all-zero function until we observe a positive point $x_{i,j}$. Then we know h_j is the target hypothesis.
At round t, the adversary picks $x_{t,0}$. For any deterministic learning algorithm:

- If the learner predicts any point in $\{x_{t,j} | j = 0, 1, \ldots, k\}$ by positive, then the prediction is positive. We know half of the hypotheses predict negative. Hence the adversary forces a mistake by letting $y_t = 0$ and reduces half of the hypotheses.

- If the learner predicts all-zero over $\{x_{t,j} | j = 0, 1, \ldots, k\}$, the adversary forces a mistake by letting $y_t = 1$ and reduces half of the hypotheses.

We can extend the example to the case of $\mathrm{Ldim}\mathcal{H} = d$ by making d independent copies of $(X, \mathcal{H}, G^\star)$. Then we prove the lower bound for all deterministic algorithms. We can then extend the lower bound to randomized algorithms by the technique of Filmus et al. (2023). ■

Appendix D. Post-Manipulation Feedback Setting

In this setting, the underlying graph $G^\star$ is known, but the original feature x_t is not observable before the learner selects the learner. Instead, either the original feature x_t or the manipulated feature v_t is revealed afterward. For PAC learning, since we can obtain an i.i.d. sample $(x_t, N_{G^\star}(x_t), y_t)$ by implementing $h_t = \mathbb{1}(\mathcal{X})$, we can still run ERM and obtain $\widehat{h}$ by optimizing Eq (1). Hence, the positive PAC learning result (Theorem 18) in the fully informative setting should still apply here. Since the post-manipulation feedback setting is harder than the fully informative setting, the negative result (Theorem 19) also holds. However, this is not true for online learning.

D.1. Online Learning

We have stated the algorithms and results in Section 3.3. Here we only provide the proofs for the theorems.

Theorem 9 *For any hypothesis class $\mathcal{H}$, graph $G^\star$ with maximum out-degree k, and a standard online learning algorithm with mistake bound M for $\mathcal{H}$, for any realizable sequence, we can achieve mistake bound of $O(k \log k \cdot M)$ by Algorithm 1. By letting the SOA algorithm by Littlestone (1988) as the input standard online algorithm $\mathcal{A}$, Red2Online-PMF(SOA) makes at most $O(k \log k \cdot \mathrm{Ldim}(\mathcal{H}))$ mistakes.*

Proof Given any online learning algorithm $\mathcal{A}$ with mistake bound M. Let W_t denote the total weight of experts at the beginning of round t. When we make a mistake at round t, there are two cases.

- False positive: It means that the induced labeling of x_t is 1 but the true label is $y_t = 0$. Since the true label is 0, the neighbor v_t we observe is labeled 0 by the target hypothesis $h^\star$. In this case, we proceed by updating all experts that predict v_t with 1 with the example $(v_t, 0)$ and we also half their weights. Since $h_t(v_t) = 1$, we have $\sum_{\mathcal{A}_H \in E: \mathcal{A}_H(x)=1} w_{\mathcal{A}_H} \geq \frac{W_t}{2(k+1)}$. Therefore, we have $W_{t+1} \leq W_t(1 - \frac{1}{4(k+1)})$.
- False negative: It means that the induced labeling of x_t is 0, but the true label is $y_t = 1$. This implies that our learner h_t labeled the entire neighborhood of $x_t = v_t$ with 0, but $h^\star$ labels some point in the neighborhood by 1. By our prediction rule, we have

$$\begin{aligned}\sum_{\mathcal{A}_H \in E: \forall x \in N_{G^\star}[x_t] \mathcal{A}_H(x)=0} w_{\mathcal{A}_H} =& W_t - \sum_{x \in N_{G^\star}[x_t]} \sum_{\mathcal{A}_H \in E: \mathcal{A}_H(x)=1} w_{\mathcal{A}_H} \\ \geq& (1 - (k+1)\frac{1}{2(k+1)})W_t \\ =& \frac{1}{2} W_t\end{aligned}$$

  We split $\mathcal{A}_H$ into $\{\mathcal{A}_{H_{x,+1}} | x \in N_{G^\star}[x_t]\}$ and split the weight $w_{\mathcal{A}_H}$ equally and multiply by $\frac{1}{2}$. Thus, we have $W_{t+1} \leq (1 - \frac{1}{4})W_t$.

Hence, whenever we make a mistake the total weight is reduced by a multiplicative factor of $\frac{1}{4(k+1)}$. Let N denote the total number of mistakes by Red2Online($\mathcal{A}$)-FI. Then we have total weight $\leq (1 - \frac{1}{4(k+1)})^N$.

On the other hand, note that there is always an expert, whose hypothesis class contains $h^\star$, being fed with examples $(\pi_{G^\star,h^\star}(x_t), y_t)$. At each mistake round t, if the expert (containing $h^\star$) makes a false positive mistake, its weight is reduced by half. If the expert made a false negative mistake, it is split into a few experts, one of which is fed by $(\pi_{G^\star,h^\star}(x_t), y_t)$. This specific new expert will contain $h^\star$ and the weight is reduced by at most $\frac{1}{2(k+1)}$. Thus, since this expert will make at most M mistakes, the weight of such an expert will be at least $(\frac{1}{2(k+1)})^M$.
Thus we have $(1 - \frac{1}{4(k+1)})^N \geq 1/(2(k+1))^M$, which yields the bound $N \leq 4kM \ln(2(k+1))$. ■

Theorem 10 *For any $k, d > 0$, there exists a graph $G^\star$ with maximum out-degree k and a hypothesis class $\mathcal{H}$ with $\mathrm{Ldim}(\mathcal{H}) = d$ such that for any deterministic algorithm, there exists a realizable sequence for which the algorithm will make at least $d(k-1)$ mistakes.*

Proof Consider a star graph $G = (X, E)$ where x_0 is the center and $x_1, \ldots, x_k$ are the leaves. The hypothesis class is singletons over leaves, i.e., $H = \{h_i | i \in [k]\}$ with $h_i = \mathbb{1}(x_i)$. Similar to the proof of Theorem 4.6 in Ahmadi et al. (2023), any deterministic algorithm will make at least $k-1$ mistakes. At round t:

- If the learner predicts all points in X as 0, the adversary picks the agent x_0. The agent will not move, the induced labeling is 0, and the true label is 1 (no matter which singleton is the target hypothesis). We do not learn anything about the target hypothesis.
- If the learner predicts x_0 as positive, then the adversary picks the agent x_0. The agent does not move, the induced labeling is 1, and the true label is 0 (no matter which singleton is the target hypothesis). Again, we learn nothing about the target hypothesis.
- If the learner predicts any node x_i with i as positive, the adversary picks the agent x_i and label it by 0. The learner's induced labeling of x_i is 1, and the true label is 0. Only one hypothesis $\mathbb{1}(X_i)$ is inconsistent and eliminated.

Hence, for any deterministic algorithm, there exists a realizable sequence such that the algorithm will make at least $k-1$ mistakes.
Now consider d independent copies of the star graphs and singletons. More specifically, consider $d(k+1)$ nodes $\{x_{i,j} | i = 1, \ldots, d, j = 0, \ldots, k\}$, where $x_{i,0}$ is connected to $\{x_{i,j} | j = 1, \ldots, k\}$ for all $i \in [\log k]$. So $\{x_{i,j} | j = 0, \ldots, k\}$ compose a star graph. For each hypothesis in the hypothesis class $\mathcal{H}$, it labels exactly one node in $\{x_{i,j} | j = 1, \ldots, k\}$ as positive for all $i \in [d]$. Hence, $\mathcal{H}$ has Littlestone dimension $\mathrm{Ldim}(\mathcal{H}) = d$. Since every copy of the star graph-singletons is independent, we can show that any deterministic algorithm will make at least $d(k-1)$ mistakes. ■

Appendix E. Unknown Manipulation Graph Setting

In this setting, the underlying graph $G^\star$ is unknown. Instead, the learner is given the prior knowledge of a finite graph class $\mathcal{G}$. When $G^\star$ is undisclosed, we cannot compute v_t given x_t. Consequently, the scenarios involving the observation timing of the features encompass the following: observing x_t beforehand followed by v_t afterward, observing (x_t, v_t) afterward, and observing either x_t or v_t

afterward, arranged in order of increasing difficulty. In this section, we provide results for the easiest case of observing x_t beforehand followed by v_t afterward. At the end, we will provide a negative result in the second easiest setting of observing (x_t, v_t) afterward.
Since we not only have a hypothesis class $\mathcal{H}$ but also a graph class $\mathcal{G}$, we formally define realizability based on both the hypothesis class and the graph class as follows.

Definition 22 ($(\mathcal{G}, \mathcal{H})$-Realizability) *A sequence of agents $(x_1, y_1), \ldots, (x_T, y_T)$ is $(\mathcal{G}, \mathcal{H})-$realizable if there exists a graph $G \in \mathcal{G}$ such that the neighborhood of x_t in G is identical to that in $G^\star$, i.e., $N_G(x_t) = N_{G^\star}(x_t)$ and there exists a perfect hypothesis $h^\star \in \mathcal{H}$ satisfying $\ell^{str}_{G^\star}(h, (x_t, y_t)) = 0$ for all $t = 1, \ldots, T$.*
For any data distribution $\mathcal{D}$, we say that $\mathcal{D}$ is $(\mathcal{G}, \mathcal{H})$-realizable if there exists a graph $G \in \mathcal{G}$ such that $\mathbb{P}_{(x,y)\sim\mathcal{D}}\left(N_G(x) \neq N_{G^\star}(x)\right) = 0$ and there exists a perfect hypothesis $h^\star \in \mathcal{H}$ s.t. $\mathcal{L}^{str}_{G^\star,\mathcal{D}}(h^\star) = 0$.

E.1. PAC Learning

Realizable PAC Learning We have described our algorithm in Section 3.2. Here, we restate the algorithm and the theorems with the proofs.
Prediction: At each round t, after observing x_t, we implement the hypothesis $h(x) = \mathbb{1}(x \neq x_t)$, which labels all every point as positive except x_t. Then we can obtain a manipulated feature vector $v_t \sim \mathrm{Unif}(N_{G^\star}(x))$.
Output: Let W denote the all graph-hypothesis pairs of (G, h) satisfying that

$$\sum_{t=1}^{T} \mathbb{1}(v_t \notin N_G(x_t)) = 0\,, \tag{6}$$

$$\sum_{t=1}^{T} \mathbb{1}(\overline{h}_G(x_t) \neq y_t) = 0\,, \tag{7}$$

where Eq (6) guarantees that every observed feature v_t is a neighbor of x_t in G and Eq (7) guarantees that h has zero empirical strategic loss when the graph is G.
Let

$$(\widehat{G}, \widehat{h}) = \operatorname*{arg\,min}_{(G,h)\in W} \sum_{t=1}^{T} |N_G(x_t)|$$

be the graph-hypothesis pair such that the graph has a **minimal empirical degree**. Finally, we output $\widehat{h}$.

Theorem 6 *For any hypothesis class $\mathcal{H}$ with $\mathrm{VCdim}(\mathcal{H}) = d$, the underlying true graph $G^\star$ with maximum out-degree at most k, finite graph class $\mathcal{G}$ in which all graphs have maximum out-degree at most k, any $(\mathcal{G}, \mathcal{H})$-realizable data distribution, and any $\varepsilon, \delta \in (0, 1)$, with probability at least $1 - \delta$ over $S \sim \mathcal{D}^T$ and $v_{1:T}$ where $T = O(\frac{d\log(kd)\log(1/\varepsilon) + \log|\mathcal{G}|\cdot(\log(1/\varepsilon)+k) + k\log(1/\delta)}{\varepsilon})$, the output $\widehat{h}$ satisfies $\mathcal{L}^{str}_{G^\star,\mathcal{D}}(\widehat{h}) \leq \varepsilon$.*

Proof Since the manipulation is local (i.e., agents can only manipulate to their neighbors), if the neighborhood of x is the same in two different graphs G_1, G_2, i.e., $N_{G_1}(x) = N_{G_2}(x)$, then for any

implementation h, the induced labeling of x is the same $\overline{h}_{G_1}(x) = \overline{h}_{G_2}(x)$. Therefore, the strategic loss of $\widehat{h}$ can be written as

$$\begin{aligned}\mathcal{L}^{\text{str}}_{G^\star,\mathcal{D}}(\widehat{h}) =& \mathbb{P}_{(x,y)\sim\mathcal{D}}\left(\overline{\widehat{h}}_{G^\star}(x) \neq y\right) \\ \leq& \mathbb{P}_{(x,y)\sim\mathcal{D}}\left(N_{G^\star}(x) \neq N_{\widehat{G}}(x)\right) + \mathbb{P}_{(x,y)\sim\mathcal{D}}\left(\overline{\widehat{h}}_{\widehat{G}}(x) \neq y \wedge N_{G^\star}(x) = N_{\widehat{G}}(x)\right) \\ \leq& \mathbb{P}_{(x,y)\sim\mathcal{D}}\left(N_{G^\star}(x) \neq N_{\widehat{G}}(x)\right) + \mathbb{P}_{(x,y)\sim\mathcal{D}}\left(\overline{\widehat{h}}_{\widehat{G}}(x) \neq y\right) . \end{aligned} \tag{8}$$

We bound the second term by uniform convergence since its empirical estimate is zero (see Eq (7)) in Lemma 23 and the first term in Lemma 24. ■

Lemma 23 *With probability at least $1-\delta$ over $(x_1, y_1), \ldots, (x_T, y_T) \sim \mathcal{D}^T$, for all (h, G) satisfying $\frac{1}{T}\sum_{t=1}^{T} \mathbb{1}(\overline{h}_G(x_t) \neq y_t) = 0$, we have*

$$\mathbb{P}_{(x,y)\sim\mathcal{D}}(\overline{h}_G(x) \neq y) \leq \varepsilon\,,$$

when $T = O(\frac{(d\log(kd)+\log|\mathcal{G}|)\log(1/\varepsilon)+\log(1/\delta)}{\varepsilon})$.
Hence, we have

$$\mathbb{P}_{(x,y)\sim\mathcal{D}}(\overline{\widehat{h}}_{\widehat{G}}(x) \neq y) \leq \varepsilon\,.$$

Proof Let $\ell_{\text{pair}}(h, G) := \mathbb{P}_{(x,y)\sim\mathcal{D}}(\overline{h}_G(x) \neq y)$ denote the loss of the hypothesis-graph pair and $\widehat{\ell}_{\text{pair}}(h, G) = \frac{1}{T}\sum_{t=1}^{T} \mathbb{1}(\overline{h}_G(x_t) \neq y_t)$ denote the corresponding empirical loss. To prove the lemma, it suffices to bound the VC dimension of the composite function class $\mathcal{H} \circ \mathcal{G} := \{\overline{h}_G | h \in \mathcal{H}, G \in \mathcal{G}\}$ and then apply the tool of VC theory.
Since for any n points, the number of labeling ways by $\mathcal{H} \circ \mathcal{G}$ is upper bounded by the sum of the number of labeling ways by $\overline{\mathcal{H}}_G$ over all graphs $G \in \mathcal{G}$. As discussed in the proof of Theorem 4, for any fixed G with maximum out-degree k, the number of labeling ways by $\overline{\mathcal{H}}_G$ is at most $O((nk)^d)$. Therefore, the total number of labeling ways over n points by $\mathcal{H} \circ \mathcal{G}$ is at most $O(|\mathcal{G}|(nk)^d)$. Hence, we have $\text{VCdim}(\mathcal{H} \circ \mathcal{G}) = O(d\log(kd) + \log(|\mathcal{G}|))$. Therefore, by VC theory, we have

$$\mathbb{P}(\ell_{\text{pair}}(\widehat{h}, \widehat{G}) > \varepsilon) \leq \mathbb{P}(\exists (h, G) \in \mathcal{H} \times \mathcal{G} \text{ s.t. } \ell_{\text{pair}}(h, G) > \varepsilon, \widehat{\ell}_{\text{pair}}(h, G) = 0) \leq \delta$$

when $T = O(\frac{(d\log(kd)+\log|\mathcal{G}|)\log(1/\varepsilon)+\log(1/\delta)}{\varepsilon})$.

■

Lemma 24 *With probability at least $1-\delta$ over $x_{1:T}, v_{1:T}$, for all G satisfying $\frac{1}{T}\sum_{t=1}^{T} \mathbb{1}(N_{G^\star}(x_t) \neq N_G(x_t)) = 0$, we have*

$$\mathbb{P}_{(x,y)\sim\mathcal{D}}(N_{G^\star}(x) \neq N_G(x)) \leq \varepsilon\,,$$

when $T \geq \frac{8k\log(|\mathcal{G}|/\delta)}{\varepsilon}$.
Hence, we have

$$\mathbb{P}_{(x,y)\sim\mathcal{D}}(N_{G^\star}(x) \neq N_{\widehat{G}}(x)) \leq \varepsilon\,.$$

Proof of Lemma 24 Let $\mathcal{L}_{\text{neighborhood}}(G) = \mathbb{P}_{(x,y)\sim\mathcal{D}}(N_{G^\star}(x) \neq N_G(x))$ denote the loss of a graph G, which is the 0/1 loss of neighborhood prediction. Let $\widehat{\mathcal{L}}_{\text{neighborhood}}(G) = \frac{1}{T}\sum_{t=1}^{T} \mathbb{1}(N_{G^\star}(x_t) \neq N_G(x_t))$ denote the corresponding empirical loss. It is hard to get an unbiased estimate of the loss $\mathcal{L}_{\text{neighborhood}}(G)$ since we cannot observe the neighborhood of any sampled x. However, it is easy for us to observe a $v \in N_{G^\star}(x)$ and remove all inconsistent G. Then the challenge is: how can we figure out the case of a strictly larger graph? This corresponds to the case that $N_{G^\star}(x)$ is a strict subset of $N_G(x)$. We deal with this case by letting $\widehat{G}$ be the "smallest" consistent graph.

Claim 1 *Suppose that $G^\star$ and all graphs $G \in \mathcal{G}$ have maximum out-degree at most k. For any $G \in \mathcal{G}$ with the minimal empirical degrees, i.e., $\sum_{t=1}^{T} |N_G(x_t)| - |N_{G^\star}(x_t)| \leq 0$, we have*

$$\widehat{\mathcal{L}}_{\textit{neighborhood}}(G) \leq \frac{2k}{T}\sum_{t=1}^{T} \mathbb{P}_{v\sim\text{Unif}(N_{G^\star}(x_t))}(v \notin N_G(x_t))\,.$$

Proof We decompose

$$\mathbb{1}(N_{G^\star}(x) \neq N_G(x)) \leq |N_G(x) \setminus N_{G^\star}(x)| + |N_{G^\star}(x) \setminus N_G(x)|\,. \tag{9}$$

Since $\sum_{t=1}^{T} |N_G(x_t)| \leq \sum_{t=1}^{T} |N_{G^\star}(x_t)|$, we have

$$\sum_{t=1}^{T} |N_G(x_t) \setminus N_{G^\star}(x_t)| \leq \sum_{t=1}^{T} |N_{G^\star}(x_t) \setminus N_G(x_t)|\,, \tag{10}$$

by subtracting $\sum_{t=1}^{T} |N_G(x_t) \cap N_{G^\star}(x_t)|$ on both sides.
By combining Eqs (9) and (10), we have

$$\begin{aligned}
\widehat{\mathcal{L}}_{\text{neighborhood}}(G) =& \frac{1}{T}\sum_{t=1}^{T} \mathbb{1}(N_{G^\star}(x_t) \neq N_G(x_t)) \\
\leq& \frac{2}{T}\sum_{t=1}^{T} |N_{G^\star}(x_t) \setminus N_G(x_t)| \\
=& \frac{2}{T}\sum_{t=1}^{T} \sum_{v\in N_{G^\star}(x_t)} \mathbb{1}(v \notin N_G(x_t)) \\
=& \frac{2}{T}\sum_{t=1}^{T} |N_{G^\star}(x_t)|\, \mathbb{P}_{v\sim\text{Unif}(N_{G^\star}(x_t))}(v \notin N_G(x_t)) \\
\leq& \frac{2}{T}\sum_{t=1}^{T} k\mathbb{P}_{v\sim\text{Unif}(N_{G^\star}(x_t))}(v \notin N_G(x_t))\,.
\end{aligned}$$

■

Now we have connected $\widehat{\mathcal{L}}_{\text{neighborhood}}(G)$ with $\frac{1}{T}\sum_{t=1}^{T} \mathbb{P}_{v\sim\text{Unif}(N_{G^\star}(x_t))}(v \notin N_G(x_t))$, where the latter one is estimable. Since $\widehat{G}$ is a consistent graph, we have

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{1}(v_t \notin N_{\widehat{G}}(x_t))) = 0\,,$$

which is an empirical estimate of $\frac{1}{T}\sum_{t=1}^{T}\mathbb{P}_{v\sim\mathrm{Unif}(N_{G^\star}(x_t))}(v \notin N_{\widehat{G}}(x_t))$. Then by showing that this loss is small, we can show that $\widehat{\mathcal{L}}_{\text{neighborhood}}(\widehat{G})$ is small.

Claim 2 *Suppose that $G^\star$ and all graphs $G \in \mathcal{G}$ have maximum out-degree at most k and $\mathcal{D}$ is $(\mathcal{G},\mathcal{H})$-realizable. For any fixed sampled sequence of $x_{1:T}$, with probability at least $1-\delta$ over $v_{1:T}$ (where v_t is sampled from $\mathrm{Unif}(N_{G^\star}(x_t))$), we have*

$$\widehat{\mathcal{L}}_{\textit{neighborhood}}(\widehat{G}) \le \varepsilon\,,$$

when $T \ge \frac{14k\log(|\mathcal{G}|/\delta)}{3\varepsilon}$.

Proof According to Claim 1, we only need to upper bound $\frac{1}{T}\sum_{t=1}^{T}\mathbb{P}_{v\sim\mathrm{Unif}(N_{G^\star}(x_t))}(v \notin N_{\widehat{G}}(x_t))$ by $\frac{\varepsilon}{2k}$. For any graph G, by empirical Bernstein bounds (Theorem 11 of Maurer and Pontil (2009)), with probability at least $1-\delta$ over $v_{1:T}$ we have

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T}\mathbb{P}_{v\sim\mathrm{Unif}(N_{G^\star}(x_t))}(v \notin N_G(x_t)) \le& \frac{1}{T}\sum_{t=1}^{T}\mathbb{1}(v_t \notin N_G(x_t))\\ &+\sqrt{\frac{2V_{G,T}(v_{1:T})\log(2/\delta)}{T}}+\frac{7\log(1/\delta)}{3T}\,,\end{aligned}$$

where $V_{G,T}(v_{1:T}) := \frac{1}{T(T-1)}\sum_{t,\tau=1}^{T}\frac{(\mathbb{1}(v_t\notin N_G(x_t))-\mathbb{1}(v_\tau\notin N_G(x_\tau)))^2}{2}$ is the sample variance. Since $\widehat{G}$ is consistent, we have $\frac{1}{T}\sum_{t=1}^{T}\mathbb{1}(v_t \notin N_{\widehat{G}}(x_t)) = 0$ and $V_{\widehat{G},T}(v_{1:T}) = 0$. Then by union bound over all $G \in \mathcal{G}$, we have that with probability at least $1-\delta$,

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{P}_{v\sim\mathrm{Unif}(N_{G^\star}(x_t))}(v \notin N_{\widehat{G}}(x_t)) \le \frac{7\log(|\mathcal{G}|/\delta)}{3T}\,.$$

■

Now we have that for any fixed sampled sequence of $x_{1:T}$, w.p. at least $1-\delta$ over $v_{1:T}$,

$$\widehat{\mathcal{L}}_{\text{neighborhood}}(\widehat{G}) \le \varepsilon.$$

We will apply concentration inequality over $x_{1:T}$ to show that $\mathcal{L}_{\text{neighborhood}}(\widehat{G})$ is small and then finish the proof of Lemma 24.

$$\begin{aligned}&\mathbb{P}_{x_{1:T},v_{1:T}}(\mathcal{L}_{\text{neighborhood}}(\widehat{G}) > 2\varepsilon)\\ \le&\mathbb{P}_{x_{1:T},v_{1:T}}(\mathcal{L}_{\text{neighborhood}}(\widehat{G}) > 2\varepsilon, \widehat{\mathcal{L}}_{\text{neighborhood}}(\widehat{G}) \le \varepsilon) + \mathbb{P}_{x_{1:T},v_{1:T}}(\widehat{\mathcal{L}}_{\text{neighborhood}}(\widehat{G}) > \varepsilon)\\ \le&\mathbb{P}_{x_{1:T}}(\exists G\in\mathcal{G}, \mathcal{L}_{\text{neighborhood}}(G) > 2\varepsilon, \widehat{\mathcal{L}}_{\text{neighborhood}}(G) \le \varepsilon) + \delta \qquad \text{(Claim 2)}\\ \le&2\delta\,,\end{aligned}$$

where the last inquality holds by Chernoff bounds when $T \ge \frac{8}{\varepsilon}(\log(|\mathcal{G}|/\delta))$. Hence, combined with Claim 2, we need $T \ge O(\frac{k\log(|\mathcal{G}|/\delta)}{\varepsilon})$ overall.

■

Theorem 7 *There exists a hypothesis class $\mathcal{H}$ with $\mathrm{VCdim}(\mathcal{H}) = 1$ and a graph class $\mathcal{G}$ in which all graphs have a maximum out-degree of 1. For any algorithm $\mathcal{A}$, there exists a data distribution $\mathcal{D}$ such that for any i.i.d. sample of size T, there exists a graph $G \in \mathcal{G}$ consistent with the data history such that when $T \leq \frac{\log|\mathcal{G}|}{\varepsilon}$, we have $\mathcal{L}^{str}_{G,\mathcal{D}}(\widehat{h}) \geq \varepsilon$.*

Proof Consider the input space of one node o and $n(n+1)$ nodes $\{x_{ij}|i = 0, \ldots, n, j = 1, \ldots, n\}$, which are partitioned into n subsets $X_0 = \{x_{01}, x_{02}, ...x_{0n}\}, X_1 = \{x_{11}, x_{12}, ...x_{1n}\}, ..., X_n$. The hypothesis will label one set in $\{X_1, \ldots, X_n\}$ by positive and the hypothesis class is $H = \{\mathbb{1}(X_i)|i \in [n]\}$. The target function is $\mathbb{1}(X_{i^*})$. This class is analogous to singletons if we view each group as a composite node. However, since the degree of the manipulation graph is limited to 1, we split one node into n copies. The marginal data distribution put probability mass $1 - 2\varepsilon$ on the irrelevant node o and the remaining 2ε uniformly over X_0.
Let the graph class $\mathcal{G}$ be the set of all graphs which connect x_{0i} to at most one node in $\{x_{1i}, \ldots, x_{ni}\}$ for all i. So the cardinality of the graph class $\mathcal{G}$ is $|\mathcal{G}| = (n+1)^n$.
When the sample size T is smaller than $\frac{n}{8\varepsilon}$, we can sample at most $n/2$ examples from X_0 with constant probability by Chernoff bounds. Then there are at least $n/2$ examples in X_0 that have not been sampled.
W.l.o.g., let $x_{01}, \ldots, x_{0\frac{n}{2}}$ denote the sampled examples. The graph G does not put any edge on these sampled examples. So all these examples are labeled as negative no matter what i^* is. Then looking at the output $\widehat{h}$ at any unseen example x_{0j} in X_0.

- If $\widehat{h}$ predicts all points in $\{x_{0j}, x_{1j}, \ldots, x_{nj}\}$ as 0, we add an edge between x_{0j} and x_{i^*j} to G. So $\widehat{h}$ misclassfy x_{0j} under manipulation graph G.

- If $\widehat{h}$ predicts x_{0j} as positive, then we do not add any edge on x_{0t} in G. So $\widehat{h}$ will classify x_{0j} as 1 but the target hypothesis will label x_{0j} as 0.

- If $\widehat{h}$ predicts any node x_{ij} with $i \neq i^* \in [n]$ as positive, we add an edge between x_{0j} and x_{ij}. So $\widehat{h}$ will classify x_{0j} as 1 but the target hypothesis will label x_{0j} as 0.

- If $\widehat{h}$ predicts exactly x_{i^*j} in $\{x_{0j}, x_{1j}, \ldots, x_{nj}\}$ as positive. Then since we can arbitrarily pick i^* at the beginning, there must exist at least one i^* such that at most this case will not happen.

Therefore, $\widehat{h}$ misclassfy every unseen point in X_0 under graph G and $\mathcal{L}^{\mathrm{str}}_{G,\mathcal{D}}(\widehat{h}) \geq \varepsilon$. ■

Agnostic PAC Learning Now, we explore the agnostic setting where there may be no perfect graph in $\mathcal{G}$, no perfect hypothesis $h^\star \in \mathcal{H}$, or possibly neither. In the following, we first define the loss of the optimal loss of $\mathcal{H}$ and the optimal loss of $\mathcal{G}$ and aim to find a predictor with a comparable loss.

Definition 25 (Optimal loss of $\mathcal{H}$) *Let $\Delta_{\mathcal{H}}$ denote the optimal strategic loss achievable by hypotheses in $\mathcal{H}$. That is to say,*

$$\Delta_{\mathcal{H}} := \inf_{h \in \mathcal{H}} \mathcal{L}^{str}_{G^\star,\mathcal{D}}(h)\,.$$

Definition 26 (Optimal loss of $\mathcal{G}$) *Let $\Delta_{\mathcal{G}}$ denote the graph loss (0/1 loss of neighborhood) of the optimal graph $G^\dagger \in \mathcal{G}$. That is to say,*

$$\Delta_{\mathcal{G}} := \min_{G^\dagger \in \mathcal{G}} \mathcal{L}_{\textit{neighborhood}}(G) = \min_{G^\dagger \in \mathcal{G}} \mathbb{P}_{(x,y)\sim\mathcal{D}}(N_{G^\star}(x) \neq N_G(x))\,.$$

We start introducing our algorithm by providing the following lemma, which states that if we are given an approximately good graph, then by minimizing the strategic loss under this approximate graph, we can find an approximately good hypothesis.

Lemma 27 *Given a graph G with the loss of neighborhood being $\mathcal{L}_{\textit{neighborhood}}(G) = \alpha$, for any h being ε-approximate optimal under manipulation graph G is the true graph, i.e., h satisfies $\mathcal{L}^{\textit{str}}_{G,\mathcal{D}}(h) \le \inf_{h^\star\in\mathcal{H}} \mathcal{L}^{\textit{str}}_{G,\mathcal{D}}(h^\star) + \varepsilon$, it must satisfy*

$$\mathcal{L}^{\textit{str}}_{G^\star,\mathcal{D}}(h) \le 2\alpha + \Delta_{\mathcal{H}} + \varepsilon\,.$$

Proof By definition, we have

$$\begin{aligned}
\mathcal{L}^{\text{str}}_{G^\star,\mathcal{D}}(h) =& \mathbb{P}_{(x,y)\sim\mathcal{D}}\left(\overline{h}_{G^\star}(x) \neq y\right)\\
=& \mathbb{P}_{(x,y)\sim\mathcal{D}}\left(\overline{h}_G(x) \neq y \wedge N_{G^\star}(x) = N_G(x)\right) + \mathbb{P}_{(x,y)\sim\mathcal{D}}\left(\overline{h}_{G^\star}(x) \neq y \wedge N_{G^\star}(x) \neq N_G(x)\right)\\
\le& \mathbb{P}_{(x,y)\sim\mathcal{D}}\left(\overline{h}_G(x) \neq y\right) + \mathbb{P}_{(x,y)\sim\mathcal{D}}\left(N_{G^\star}(x) \neq N_G(x)\right)\\
\le& \mathbb{P}_{(x,y)\sim\mathcal{D}}\left(\overline{h^\star}_G(x) \neq y\right) + \varepsilon + \alpha \qquad (\varepsilon\text{-approximate optimality of } h)\\
=& \mathbb{P}_{(x,y)\sim\mathcal{D}}\left(\overline{h^\star}_{G^\star}(x) \neq y \wedge N_{G^\star}(x) = N_G(x)\right) + \mathbb{P}_{(x,y)\sim\mathcal{D}}\left(N_{G^\star}(x) \neq N_G(x)\right) + \varepsilon + \alpha\\
\le& \mathbb{P}_{(x,y)\sim\mathcal{D}}\left(\overline{h^\star}_{G^\star}(x) \neq y\right) + \varepsilon + 2\alpha = \Delta_{\mathcal{H}} + \varepsilon + 2\alpha\,.
\end{aligned}$$

■

Then the remaining question is: **How to find a good approximate graph?** As discussed in the previous section, the graph loss $\mathcal{L}_{\text{neighborhood}}(\cdot)$ is not estimable. Instead, we construct a proxy loss that is not only close to the graph loss but also estimable. We consider the following alternative loss function as a proxy:

$$\mathcal{L}_{\text{proxy}}(G) = 2\mathbb{E}_x[P_{v\sim N_{G^\star}(x)}(v \notin N_G(x))] + \frac{1}{k}\mathbb{E}[|N_G(x)|] - \frac{1}{k}\mathbb{E}[|N_{G^\star}(x)|]\,,$$

where k is the degree of graph $G^\star$. Note that the first two terms are estimable and the third term is a constant. Hence, we can minimize this proxy loss.

Lemma 8 *Suppose that $G^\star$ and all graphs $G \in \mathcal{G}$ have maximum out-degree at most k. Then, we have*

$$\frac{1}{k}\mathcal{L}_{\textit{neighborhood}}(G) \le \mathcal{L}_{\textit{proxy}}(G) \le 3\mathcal{L}_{\textit{neighborhood}}(G)\,.$$

Proof Let $d_G = \mathbb{E}[|N_G(x)|] - \mathbb{E}[|N_{G^\star}(x)|]$ denote the difference between the expected degree of G and that of $G^\star$. We have

$$d_G = \mathbb{E}_{(x,y)\sim D}[|N_G(x)|] - \mathbb{E}_{(x,y)\sim D}[|N_{G^\star}(x)|] = \mathbb{E}[|N_G(x) \setminus N_{G^\star}(x)|] - \mathbb{E}[|N_{G^\star}(x) \setminus N_G(x)|]\,. \tag{11}$$

Then, we have

$$\begin{aligned}
\mathcal{L}_{\text{proxy}}(G) =& 2\mathbb{E}_x[\frac{|N_{G^\star}(x) \setminus N_G(x)|}{|N_{G^\star}(x)|}] + \frac{1}{k} d_G \\
\geq& 2 \cdot \frac{1}{k} \mathbb{E}_x[|N_{G^\star}(x) \setminus N_G(x)|] + \frac{1}{k} d_G \\
=& \frac{1}{k} (\mathbb{E}[|N_G(x) \setminus N_{G^\star}(x)| + |N_{G^\star}(x) \setminus N_G(x)|]) && \text{(Applying Eq (11))} \\
\geq& \frac{1}{k} \mathcal{L}_{\text{neighborhood}}(G) \,.
\end{aligned}$$

On the other hand, we have

$$\begin{aligned}
\mathcal{L}_{\text{proxy}}(G) =& 2\mathbb{E}_x[\frac{|N_{G^\star}(x) \setminus N_G(x)|}{|N_{G^\star}(x)|}] + \frac{1}{k} d_G \\
\leq& 2\mathbb{E}_x \left[\mathbb{1}(N_G(x) \neq N_{G^\star}(x))\right] + \mathbb{E}_x \left[\mathbb{1}(N_G(x) \neq N_{G^\star}(x))\right] \\
=& 3\mathcal{L}_{\text{neighborhood}}(G) \,.
\end{aligned}$$

■

Given a sequence of $S = ((x_1, v_1), \ldots, (x_T, v_T))$, we define the empirical proxy loss over the sequence S as

$$\widehat{\mathcal{L}}_{\text{proxy}}(G, S) = \frac{2}{T} \sum_{t=1}^{T} \mathbb{1}(v_t \notin N_G(x_t)) + \frac{1}{kT} \sum_{t=1}^{T} |N_G(x_t)| - \frac{1}{k} \mathbb{E}_{(x,y)\sim D} \left[|N_{G^\star}(x)|\right]$$

where the first two terms are the empirical estimates of the first two terms of $\mathcal{L}_{\text{proxy}}(G)$ and the last term is not dependent on G, which is a constant.
Similar to the realizable setting, by implementing the hypothesis $h_t = \mathbb{1}(x \neq x_t)$, which labels every point as positive except x_t, we observe the manipulated feature vector $v_t \sim \text{Unif}(N_{G^\star}(x))$.
Given two samples,

$$S_1 = ((x_1, v_1, y_1), \ldots, (x_{T_1}, v_{T_1}, y_{T_1}))$$

and

$$S_2 = ((x'_1, v'_1, y'_1), \ldots, (x'_{T_2}, v'_{T_2}, y'_{T_2})),$$

we use S_1 to learn an approximate good graph and S_2 to learn the hypothesis.
Let $\widehat{G}$ be the graph minimizing the empirical proxy loss, i.e.,

$$\widehat{G} = \arg\min_{G \in \mathcal{G}} \widehat{\mathcal{L}}_{\text{proxy}}(G, S_1) = \arg\min_{G \in \mathcal{G}} \frac{2}{T} \sum_{t=1}^{T_1} \mathbb{1}(v_t \notin N_G(x_t)) + \frac{1}{kT} \sum_{t=1}^{T_1} |N_G(x_t)| \,.$$

Then let $\widehat{h}$ be the ERM implementation assuming that $\widehat{G}$ is the true graph, i.e.,

$$\widehat{h} = \arg\min_{h \in \mathcal{H}} \frac{1}{T_2} \sum_{t=1}^{T_2} \mathbb{1}(\overline{h}_{\widehat{G}}(x'_t) \neq y'_t) \,.$$

Next, we bound the strategic loss of $\widehat{h}$ using $\varepsilon, \Delta_{\mathcal{G}}$ and $\Delta_{\mathcal{H}}$.

Theorem 28 *For any hypothesis class $\mathcal{H}$ with $\mathrm{VCdim}(\mathcal{H}) = d$, the underlying true graph $G^\star$ with maximum out-degree at most k, finite graph class $\mathcal{G}$ in which all graphs have maximum out-degree at most k, any data distribution, and any $\varepsilon, \delta \in (0,1)$, with probability at least $1-\delta$ over $S_1 \sim \mathcal{D}^{T_1}, S_2 \sim \mathcal{D}^{T_2}$ and $v_{1:T_1}$ where $T_1 = O(\frac{k^2 \log(|\mathcal{G}|/\delta)}{\varepsilon^2})$ and $T_2 = O(\frac{d\log(kd)+\log(1/\delta)}{\varepsilon^2})$, the output $\widehat{h}$ satisfies*

$$\mathcal{L}^{str}_{G^\star,\mathcal{D}}(\widehat{h}) \le 6k\Delta_{\mathcal{G}} + \Delta_{\mathcal{H}} + \varepsilon\,.$$

Proof We first prove that $\mathcal{L}_{\text{neighborhood}}(\widehat{G}) \le 3k\Delta_{\mathcal{G}} + 2\varepsilon$.
By Hoeffding bounds and union bound, with probability at least $1-\delta/2$ over S_1, for all $G \in \mathcal{G}$,

$$|\widehat{\mathcal{L}}_{\text{proxy}}(G, S_1) - \mathcal{L}_{\text{proxy}}(G)| \le \varepsilon_1\,, \tag{12}$$

when $T_1 = O(\frac{\log(|\mathcal{G}|/\delta)}{\varepsilon_1^2})$.
Hence, we have

$$\begin{aligned}
\mathcal{L}_{\text{neighborhood}}(\widehat{G}) \le& k\mathcal{L}_{\text{proxy}}(\widehat{G}) && \text{(Applying Lemma 8)}\\
\le& k\widehat{\mathcal{L}}_{\text{proxy}}(\widehat{G}, S_1) + k\varepsilon_1\\
\le& k\widehat{\mathcal{L}}_{\text{proxy}}(G^\dagger, S_1) + k\varepsilon_1\\
\le& k\mathcal{L}_{\text{proxy}}(G^\dagger) + 2k\varepsilon_1\\
\le& 3k\mathcal{L}_{\text{neighborhood}}(G^\dagger) + 2k\varepsilon_1 && \text{(Applying Lemma 8)}\\
=& 3k\Delta_{\mathcal{G}} + 2k\varepsilon_1\,. && \text{(Def 26)}
\end{aligned}$$

Then by VC theory and uniform convergence, with probability at least $1-\delta/2$ over S_2, for all $h \in \mathcal{H}$,

$$|\widehat{\mathcal{L}}^{\text{str}}_{\widehat{G}}(h) - \mathcal{L}^{\text{str}}_{\widehat{G},\mathcal{D}}(h)| \le \varepsilon_2\,,$$

when $T_2 = O(\frac{d\log(kd)+\log(1/\delta)}{\varepsilon_2^2})$.
Therefore $\widehat{h}$ is an approximately good implementation if $\widehat{G}$ is the true graph, i.e.,

$$\mathcal{L}^{\text{str}}_{\widehat{G},\mathcal{D}}(\widehat{h}) \le \min_{h\in\mathcal{H}} \mathcal{L}^{\text{str}}_{\widehat{G},\mathcal{D}}(h) + 2\varepsilon_2\,.$$

Then by applying Lemma 27, we get

$$\mathcal{L}^{\text{str}}_{G^\star,\mathcal{D}}(\widehat{h}) \le 6k\Delta_{\mathcal{G}} + \Delta_{\mathcal{H}} + 4k\varepsilon_1 + 2\varepsilon_2\,.$$

We complete the proof by plugging in $\varepsilon_1 = \frac{\varepsilon}{8k}$ and $\varepsilon_2 = \frac{\varepsilon}{4}$. ■

E.2. Application of the Graph Learning Algorithms in Multi-Label Learning

Our graph learning algorithms have the potential to be of independent interest in various other learning problems, including multi-label learning. To illustrate, let us consider scenarios like the recommender system, where we aim to recommend movies to users. In such cases, each user typically has multiple favorite movies, and our objective is to learn this set of favorite movies. Similarly, in

the context of object detection in computer vision, every image often contains multiple objects. Here, our learning goal is to accurately identify and output the set of all objects present in a given image.

This multi-label learning problem can be effectively modeled as a bipartite graph denoted as $G^\star = (\mathcal{X}, \mathcal{Y}, \mathcal{E}^\star)$, where $\mathcal{X}$ represents the input space (in the context of the recommender system, it represents users), $\mathcal{Y}$ is the label space (in this case, it signifies movies), and $\mathcal{E}^\star$ is a set of edges. In this graph, the presence of an edge $(x, y) \in \mathcal{E}^\star$ implies that label y is associated with input x (e.g., user x liking movie y). Our primary goal here is to learn this graph, i.e., the edges $\mathcal{E}^\star$. More formally, given a marginal data distribution $\mathcal{D}_\mathcal{X}$, our goal is to find a graph $\widehat{G}$ with minimum neighborhood prediction loss $\mathcal{L}_{\text{neighborhood}}(G) = \mathbb{P}_{x\sim\mathcal{D}_\mathcal{X}}(N_G(x) \neq N_{G^\star}(x))$. Suppose we are given a graph class $\mathcal{G}$, then our goal is to find a graph $G^\dagger$ such that

$$G^\dagger = \arg\min_{G\in\mathcal{G}} \mathcal{L}_{\text{neighborhood}}(G).$$

However, in real-world scenarios, the recommender system cannot sample a user along with her favorite movie set. Instead, at each time t, the recommender recommends a set of movies h_t to the user, and the user randomly clicks on one of the movies in h_t that she likes (i.e., $v_t \in N_{G^\star}(x_t) \cap h_t$). Here we abuse the notation a bit by letting h_t represent the set of positive points labeled by this hypothesis. This setting perfectly fits into our unknown graph setting.

Therefore, in the realizable setting where $\mathcal{L}_{\text{neighborhood}}(G^\dagger) = 0$, we can find $\widehat{G}$ such that $\widehat{G}$ is satisfying $\mathcal{L}_{\text{neighborhood}}(\widehat{G}) \leq \varepsilon$ given $O(\frac{8k\log(|\mathcal{G}|/\delta)}{\varepsilon})$ examples according to Lemma 24 . In the agnostic setting, we can find $\widehat{G}$ satisfying $\mathcal{L}_{\text{neighborhood}}(\widehat{G}) \leq 3k\mathcal{L}_{\text{neighborhood}}(G^\dagger) + \varepsilon$ given $O(\frac{k^2\log(|\mathcal{G}|/\delta)}{\varepsilon^2})$ examples according to Theorem 28.

E.3. Online Learning

In the unknown graph setting, we do not observe $N_{G^\star}(x_t)$ or $N_{G^\star}(v_t)$ anymore. We then design an algorithm by running an instance of Algorithm 1 over the majority vote of the graphs.

Algorithm 3 Online Algorithm in the Unknown Graph Setting

1: initialize an instance $\mathcal{A} =$ Red2Online-PMF (SOA, $2k$)
2: let $\mathcal{G}_1 = \mathcal{G}$
3: **for** $t = 1, 2, \ldots$ **do**
4: Prediction: after observing x_t, for every node x
5: **if** (x_t, x) are connected in at most half of the graphs in $\mathcal{G}_t$ **then**
6: $h_t(x) = 1$
7: **else**
8: let the prediction at x follow $\mathcal{A}$, i.e., $h_t(x) = \mathcal{A}(x)$
9: **end if**
10: Update: when we make a mistake at the observed node v_t:
11: **if** (x_t, v_t) are connected in at most half of the graphs in $\mathcal{G}_t$ **then**
12: update $\mathcal{G}_{t+1} = \{G \in \mathcal{G}_t | (x_t, v_t)$ are connected in $G\}$ to be the set of consistent graphs
13: **else**
14: feed $\mathcal{A}$ with $(v_t, \widetilde{N}(v_t), \widehat{y}_t, y_t)$ with $\widetilde{N}(v_t) = \{x| \, |\{G \in \mathcal{G}_t | (v_t, x)$ are connected in $G\}| \geq |\mathcal{G}_t|/2\}$, which is the set of vertices which are an out-neighbor of v_t in more than half of the graphs in $\mathcal{G}_t$
15: $\mathcal{G}_{t+1} = \mathcal{G}_t$
16: **end if**
17: **end for**

Theorem 12 *For any hypothesis class $\mathcal{H}$, the underlying true graph $G^\star$ with maximum out-degree at most k, finite graph class $\mathcal{G}$ in which all graphs have maximum out-degree at most k, for any realizable sequence, Algorithm 3 will make at most $O(k \log(k) \cdot \mathrm{Ldim}(\mathcal{H}) + \log |\mathcal{G}|)$ mistakes.*

Proof Let W_t denote the total weight of experts at the beginning of round t in the algorithm instance $\mathcal{A}$. When we make a mistake at round t, there are two cases.

- Type 1: (x_t, v_t) are connected in at most half of the graphs in $\mathcal{G}_t$. In this case, we can remove half of the graphs in $\mathcal{G}_t$, i.e., $|\mathcal{G}_{t+1}| \leq \frac{|\mathcal{G}_t|}{2}$. The total number of mistakes in this case will be $N_1 \leq \log(|\mathcal{G}|)$.

- Type 2: (x_t, v_t) are connected in more than half of the graphs in $\mathcal{G}_t$. In this case, our prediction rule applies $\mathcal{A}$ to predict at v_t. Then there are two cases (as we did in the proof of Theorem 9)

  - False positive: meaning that we predicted the neighbor v_t of x_t by 1 but the correct prediction is 0. This means that the neighbor v_t we saw should also be labeled as 0 by the target implementation $h^\star$. In this case, since Algorithm 1 does not use the neighborhood to update, and it does not matter what $\widetilde{N}_{G^\star}(v_t)$ is.
  - False negative: meaning we predicted x_t with 0 but the correct prediction is 1. In this case, $v_t = x_t$, the entire neighborhood $N_{G^\star}(x_t)$ is labeled as 0 by h_t, and $h^\star$ labeled some point in $N_{G^\star}(x_t)$ by 1. Since only nodes in $\widetilde{N}(v_t)$ are labeled as 0 by our algorithm (as stated in line 6, all nodes not in $\widetilde{N}(v_t)$ are labeled as 1 by h_t), the true neighborhood $N_{G^\star}(v_t)$ must be a subset of $\widetilde{N}(v_t)$. Since all graphs in $\mathcal{G}$ has maximum out-degree at most k, we have $|\widetilde{N}(v_t)| \leq \frac{k \cdot |\mathcal{G}_t|}{|\mathcal{G}_t|/2} = 2k$.

Then, by repeating the same analysis of Theorem 9 (since the analysis only relies on the fact that the observed $N_{G^\star}(v_t)$ satisfy $|N_{G^\star}(v_t)| \leq k$), we will make at most $N_2 \leq O(k \log k \cdot \mathrm{Ldim}(\mathcal{H}))$ type 2 mistakes.

Therefore, we will make at most $N_1 + N_2 \leq O(\log |\mathcal{G}| + k \log k \cdot \mathrm{Ldim}(\mathcal{H}))$ mistakes. ■

Theorem 13 *For any $n \in \mathbb{N}$, there exists a hypothesis class $\mathcal{H}$ with $\mathrm{Ldim}(\mathcal{H}) = 1$, a graph class $\mathcal{G}$ satisfying that all graphs in $\mathcal{G}$ have maximum out-degree at most 1 and $|\mathcal{G}| = n$ such that for any deterministic algorithm, there exists a graph $G^\star \in \mathcal{G}$ and a realizable sequence for which the algorithm will make at least $\frac{\log n}{\log\log n}$ mistakes.*

Proof The example is very similar to the one in the proof of Theorem 7. Consider the input space of $n(n+1)$ nodes, which are partitioned into $n+1$ subsets $X_0 = \{x_{01}, x_{02}, ...x_{0n}\}, X_1 = \{x_{11}, x_{12}, ...x_{1n}\}, ..., X_n$. The hypothesis will label one set in $\{X_1, \ldots, X_n\}$ by positive and the hypothesis class is $H = \{\mathbb{1}(X_i) | i \in [n]\}$. The target function is $\mathbb{1}(X_{i^*})$. This class is analogous to singletons if we view each group as a composite node. However, since the degree of the manipulation graph is limited to 1, we split one node into n copies.
The agent will always pick an agent in X_0 and the true label is positive only when it is connected to X_{i^*}. To ensure that the degree of every node is at most 1, only one node in each set will only be used in round t, i.e., $\{x_{0t}, x_{1t}, \ldots, x_{nt}\}$.
At round t:

- If the learner predicts all points in $\{x_{0t}, x_{1t}, \ldots, x_{nt}\}$ as 0, the adversary picks the agent x_{0t} and adds an edge between x_{0t} and x_{i^*t}. The agent does not move, the learner predicts 0, and the true label is 1 no matter what i^* is. We do not learn anything about the target function.

- If the learner predicts x_{0t} as positive, then the adversary picks the agent x_{0t} and does not add any edge on x_{0t}. The agent does not move, the learner predicts 1, and the true label is 0. We learn nothing.

- If the learner predicts any node x_{it} with $i \neq i^* \in [n]$ as positive, the adversary picks the agent x_{it} and adds no edge on x_{0t}. The agent will stay at x_{it}, the learner predicts 1, and the true label is 0. But we can only eliminate one hypothesis $\mathbb{1}(X_i)$.

Hence there must exist an i^* the algorithm will make at least $n-1$ mistakes. Since all possible graphs will add at most one edge between x_{0t} and $\{x_{1t}, \ldots, x_{nt}\}$, the graph class $\mathcal{G}$ has at most $(n+1)^n$ graphs. ■

Next, we restate Proposition 14.

Proposition 14 *When (x_t, v_t) is observed afterward, for any $n \in \mathbb{N}$, there exists a class $\mathcal{G}$ of graphs of degree 2 and a hypothesis class $\mathcal{H}$ with $|\mathcal{G}| = |\mathcal{H}| = n$ such that for any deterministic algorithm, there exists a graph $G^\star \in \mathcal{G}$ and a realizable sequence for which the algorithm will make at least $n-1$ mistakes.*

Proof Consider $n+2$ nodes, A, B and $C_1, C_2, \ldots, C_n$. The graph class $\mathcal{G}$ has n graphs in the form of $A - B - C_i$ for $i \in [n]$. The hypothesis class is singletons over $C_1, ..., C_n$. So we have n graphs with degree 2 and n hypotheses. Suppose that the true graph is $A - B - C_{i^*}$ and the target function is $\mathbb{1}(C_{i^*})$. Hence only the agents B and C_{i^*} are positive.
Then at each round:

- If the learner is all-negative, the adversary picks agent $(B, 1)$. The learner makes a mistake at $x_t = v_t = B$, and learn nothing.
- If the learner implements h_t satisfying $h_t(B) = 1$ (or $h_t(A) = 1$), then the adversary picks agent $(A, -1)$. The learner makes a mistake at $v_t = B$ (or $v_t = A$) and learns nothing.
- If the learner implements h_t predicting any C_i with $i \neq i^*$ by positive, then the adversary picks agent $(C_i, -1)$. The learner makes a mistake at $v_t = x_t = C_i$ and can only eliminate one hypothesis $\mathbb{1}(C_i)$ (and graph).

Hence, there must exist an i^* such that the learner makes at least $n - 1$ mistakes. ■

Appendix F. From Realizable Online Learning to Agnostic Online Learning

Since we cannot achieve sublinear regret even in the standard online learning by deterministic algorithms, we have to consider randomized algorithms in agnostic online learning as well. Following the model of "randomized algorithms" (instead of "fractional classifiers") by Ahmadi et al. (2023), for any randomized learner p (a distribution over hypotheses), the strategic behavior depends on the realization of the learner. Then the strategic loss of p is

$$\ell_G^{\text{str}}(p, (x, y)) = \mathbb{E}_{h \sim p}\left[\ell_G^{\text{str}}(h, (x, y))\right] .$$

When we know x_t and $N_{G^\star}(x_t)$, we actually have full information feedback (i.e., we can compute the prediction of every hypothesis ourselves). Hence, we can apply the realizable-to-agnostic technique by constructing a cover of $\mathcal{H}$ with T^M experts, where M is the mistake bound in the realizable setting, and then running multiplicative weights over these experts.
For the construction of the experts, we apply the same method by Ben-David et al. (2009). The only minor difference is that the learner in our setting needs to produce a hypothesis at each round to induce some labeling instead of deciding a labeling directly. Given the hypothesis $\widetilde{h}_t$ generated by a realizable algorithm $\mathcal{A}_{\text{REL}}$, to flip the prediction induced by $\widetilde{h}_t$, we can change the entire neighborhood of x_t to negative if $\widetilde{h}_t$'s prediction is positive, and predict x_t by positive if $\widetilde{h}_t$'s prediction is negative. If we do not know $(x_t, N_{G^\star}(x_t))$, then we do not even know how to flip the prediction.

Lemma 29 *Let $\mathcal{A}_{REL}$ be an algorithm that makes at most M mistakes in the realizable setting. Let $x_1, \ldots, x_T$ be any sequence of instances. For any $h \in \mathcal{H}$, there exists $L \leq M$ and indices $i_1, \ldots, i_L$, such that running Exp$(i_1, \ldots, i_L)$ (Algorithm 4) on the sequence $((x_1, N_{G^\star}(x_1)), \ldots, (x_T, N_{G^\star}(x_T)))$ generate the same prediction as implementing h on each round, i.e., $\overline{h}_{tG^\star}(x_t) = \overline{h}_{G^\star}(x_t)$ for all t.*

Algorithm 4 Construction of expert $\mathrm{Exp}(i_1, \ldots, i_L)$

Input: an algorithm in the realizable setting $\mathcal{A}_{\mathrm{REL}}$ and indices $i_1, \ldots, i_L$
for $t = 1, \ldots, T$ **do**
  receive x_t
  let $\widetilde{h}_t$ denote the hypothesis produced by $\mathcal{A}_{\mathrm{REL}}$, i.e.,
  $\widetilde{h}_t = \mathcal{A}_{\mathrm{REL}}((x_1, N_{G^\star}(x_1), \widetilde{y}_1), \ldots, (x_{t-1}, N_{G^\star}(x_{t-1}), \widetilde{y}_{t-1}))$
  if $t \in \{i_1, \ldots, i_L\}$ **then**
    if $\widetilde{h}_t$ labels the whole $N_{G^\star}(x_t)$ by negative **then**
      define $h_t = \widetilde{h}_t$ except $h_t(x_t) = 1$ and implement h_t
      let $\widetilde{y}_t = 1$
    else
      define h_t s.t. h_t labels the entire neighborhood $N_{G^\star}(x_t)$ by negative and implement h_t
      let $\widetilde{y}_t = 0$
    end if
  else
    implement $\widetilde{h}_t$ and let $\widetilde{y}_t = \overline{\widetilde{h}}_{tG^\star}(x_t)$
  end if
end for

Proof Fix $h \in \mathcal{H}$ and the sequence $((x_1, N_{G^\star}(x_1)), \ldots, (x_T, N_{G^\star}(x_T)))$. Then consider running $\mathcal{A}_{\mathrm{REL}}$ over $((x_1, N_{G^\star}(x_1), \overline{h}(x_1)), \ldots, (x_T, N_{G^\star}(x_T), \overline{h}(x_T)))$ and $\mathcal{A}_{\mathrm{REL}}$ makes $L \leq M$ mistakes. Define $\{i_1, \ldots, i_L\}$ to be the set of rounds $\mathcal{A}_{\mathrm{REL}}$ makes mistakes. Since the prediction of $\mathcal{A}_{\mathrm{REL}}$ is only different from $\overline{h}_{G^\star}$ at rounds in $\{i_1, \ldots, i_L\}$, the predictions when implementing $\mathrm{Exp}(i_1, \ldots, i_L)$ are the same as implementing h. ■

We finish with a result that holds directly by following Ben-David et al. (2009).

Proposition 30 *By running multiplicative weights over the experts, we can achieve regret*

$$\mathit{Regret}(T) \leq O(\sqrt{k \log k \cdot \mathrm{Ldim}(\mathcal{H}) T \log(T)}) .$$

It is unclear to us how to design algorithms in post-manipulation feedback and unknown graph settings.