Strategic Representation

Vineet Nair*
Technion – Israel Institute of Technology

Ganesh Ghalme†
Indian Institute of Technology, Hyderabad

Inbal Talgam-Cohen
Technion – Israel Institute of Technology

Nir Rosenfeld
Technion – Israel Institute of Technology

ABSTRACT

Humans have come to rely on machines for reducing excessive information to manageable representations. But this reliance can be abused—strategic machines might craft representations to manipulate their users. How can a user make good choices on strategic representations? We formalize this as a learning problem, and pursue algorithms for decision-making that are robust to manipulation. In our main setting of interest, the system represents attributes of an item to the user, who decides whether or not to consume. We model this interaction through the lens of strategic classification (Hardt et al. 2016), but *reversed*: the user, who learns, plays first; and the system, which responds, plays second. The system must respond with representations that reveal 'nothing but the truth', but need not reveal the entire truth; thus, the user faces the problem of learning set functions under strategic constraint. This presents distinct algorithmic and statistical challenges. Our main result is a learning algorithm that minimizes error despite strategic representations, and our analysis sheds light on the trade-off between learning effort and susceptibility to manipulation.

KEYWORDS

Strategic Learning; Strategic Classification; Gaming; Strategic Representation

1 INTRODUCTION

Strategic classification tells the story of how learning systems are susceptible to 'gaming' by self-interested users. But the reverse scenario—in which it is the system that strategically games its users—is quite routine. In settings where users make choices about items (e.g., click, buy, watch), systems often hold the power to determine how items are *represented*—a power they can utilize to promote their own goals. In this paper we formalize this scenario of *strategic representation*, and study how learning can aid users in choosing well despite strategic system behavior.

As a concrete example, consider a hotel booking platform, in which a hotel is presented to the user alongside a small set of representative images. As there are many images available for each hotel, the system must choose which subset of these to display. Clearly, the choice of representation can have a significant effect on users' decision-making (whether to click the hotel and even whether to book it), and therefore on the system's profit. The system may well attempt to capitalize on the control it has over what information is presented to the user at the expense of the user who may be swayed to make sub-optimal decisions. Given user's dependence on system crafted representations for making decisions, it is important that the users' *optimal* choice strategy takes into account the systems' strategizing.

Our goal in this paper is to support users in their choices. Indeed, when users face a strategic system, they face a hefty price if they don't learn to make decisions that guards against strategic behavior of system. We aim to learn user optimal choice strategy that maps representations to decisions while taking into account the fact that the representations are generated strategically. We consider binary decisions (buy, watch, click) and hence the choice strategy corresponds to binary classification rule defined on the representation space.

Our formalization. The user learns a *choice* strategy h, and the system responds with a representation of an item (the hotel in our example). We focus on discrete items, each composed of a subset of ground elements. In reality, a full description of item x is often too large for humans to process effectively, and so systems provide them with a compact representation, $\phi(x)$. We consider representation mappings ϕ that are lossy but *truthful*, meaning they reveal a cardinality-constrained subset of the item's full set of attributes: $\phi(x) \subseteq x$ and $k_1 \leq |\phi(x)| \leq k_2$. In other words, representations need not be 'the whole truth', but must include 'nothing but the truth.' Examples include retail items described by a handful of attributes; videos presented by several key frames; and movie recommendations described by a select set of reviews.

The user wishes to choose items that are 'worthwhile' to her, as determined by her valuation function—in our case, a set function, which can reflect complex tastes (e.g., account for complementarity among the attributes, such as 'balcony' and 'ocean view' in the hotel example). Meanwhile, the system aims to maximize user engagement by presenting to the user a subset of attributes (satisfying cardinality constraints) that will lead her to choose the item. E.g., in the hotel example, the system could present the attribute of having a balcony wihtout mentioning whether or not there is a view. Importantly, while values are based on the true attributes x, choices rely on the observed attributes $\phi(x)$, which the system controls. This misalignment in goals causes friction: a representation may be optimal for the system, but does not necessarily align with the interests of the user.

Overview of our results. Our main result is the user's learning algorithm in Sec. 4, which minimizes the empirical error over a hypothesis family of classifiers with complexity k (the degree of synergies—like complements—that can be taken into account when classifying the system's representation), assuming the user's true decisions based on their valuations is realizable (see Theorem 4.8).

*Vineet Nair is now at Google Research, India.
† This work was done when Ganesh was a post-doctoral fellow at Technion.

Appears at the 1st Workshop on Learning with Strategic Agents (LSA 2022). Held as part of the Workshops at the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2022), N. Bishop, M. Han, L. Tran-Thanh, H. Xu, H. Zhang (chairs), May 9–10, 2022, Online.

The algorithm builds upon several structural results we establish for binary classifiers that operate on sets, e.g., that it suffices to use simple-form set functions even when taking into account the system's strategic response (Theorem 4.7).

Thus our main result is "how to learn"; we also quantify how bad things can be if without learning, and show that even minimal learning can significantly help. We characterize how much is gained versus how much effort is invested in learning; mathematically this is reflected by the estimation and approximation errors. We analyze and bound both the estimation and approximation errors. For estimation, we give a generalization/PAC bound that is based on the VC dimension of the induced function class.

In terms of the interplay between k on the one hand (determines the power of the user) and k_1, k_2 (determines the power of the system), our results in Sec. 5 show the following balance of power: For fixed representation range $[k_1, k_2]$, the larger the complexity k, the better the user's payoff, because learned functions are more expressive and can better approximate the valuation v. Theorem 5.4 shows that when the complexity matches that of the user's valuation, the approximation error vanishes. On the other hand, a large k is costly in running time and in data—it means larger sample complexity and so larger estimation error (Theorem 5.9). We thus get a tradeoff controlled by the user's effort level k. Now keep the complexity k fixed and let the system control k_1, k_2. Perhaps surprisingly, the system can increase its payoff by 'tying its hands' to a lower k_2. This is because k_2 upper-bounds not only the system's range but also the 'effective' k of the user (who gets nothing from choosing $k > k_2$), and the lower the k, the better it is for the system (see Lem. 5.10). We discuss take-aways in Sec. 6.

1.1 Related Work

Relation to strategic classification. Our formalization of strategic representation shares a deep connection to strategic classification (SC) [12], a very active area of research—e.g. [7, 13, 16, 19, 20]. Both models have the same structure (a leading learning player who must take into account a strategic responder), but there are important differences. The first is conceptual – the roles of the system and the user are reversed.[1] E.g., in SC the system aims to learn an accurate classifier, whereas in our setting it is the user who learns. The implications of this change are mainly to the balance of power between the sides (see Sec. 5). The second difference is that the objects being classified are discrete in our setting rather than continuous, and their manipulation is only by hiding information (in the language of SC, we fix a particular cost function—distinct from the standard ones in the literature). From a learning perspective, whereas SC considers 'conventional' learning of vector functions, strategic representation focuses on learning *set functions*. We view the close connection to SC as a strength of our formalization – it shows that the SC setup is useful far beyond what was previously considered; and also that a fairly mild variation leads to a completely different learning problem, with distinct algorithmic and statistical challenges. SC works closest to ours are by Zrnic et al. [21], who change the order of play (while we switch the roles); and by Krishnaswamy et al. [15], who study masking attributes by the user (rather than dropping attributes by the system), and aim for classifiers that incentivize 'the whole truth' (while we aim to correctly classify despite manipulation).

Relation to Bayesian persuasion. In Bayesian persuasion [14], a more-informed player uses its information advantage coupled with commitment power to influence the choices of a decision-making player. The more-informed player moves first, committing to a scheme for signaling information (equiv., for recommending an action) to the decision-maker. This fits a scenario in which the system knows the user valuation and there is common knowledge of D, and this knowledge can be used to persuade (recommend an action to) the user. In our work, the order is reversed—the decision-maker (user) moves first and the more-informed player (system) follows—and the knowledge assumptions are more relaxed. Since who leads and who follows is determined by the relative frequencies at which the system and user adapt to each other's actions [21], both scenarios are well-worth studying. Bayesian persuasion works closest to ours are by Dughmi et al. [8], who upper bound the number of signaled attributes, and by Haghtalab et al. [11], who study strategic selection of anecdotes. In both these works as in ours, the human player is a learner.

2 A FORMALIZATION OF STRATEGIC REPRESENTATION

We begin by describing the setting from the perspective of the user, which is the learning entity in our setup. We then present the 'types' of users we study, and draw connections between our settings and others found in the literature.

2.1 Learning Setting

In our setup, a user is faced with a stream of items, and must choose which of these to consume. Items are discrete, with each item $x \in \mathcal{X} \subseteq 2^E$ described by a subset of ground attributes, E, where $|E| = q$. We assume all feasible items have at most n attributes, $|x| \leq n$. The value of items for the user are encoded by a *value function*, $v : \mathcal{X} \to \mathbb{R}$. We say an item x is *worthwhile* to the user if it has positive value, $v(x) > 0$, and use $y = \text{sign}(v(x)) = Y(x)$ to denote worthwhileness, i.e., $y = 1$ if x is worthwhile, and $y = -1$ if it is not. Items are presented to the user as samples drawn iid from some unknown distribution D over $\mathcal{X}$, and for each item, the user must choose whether to consume it (e.g., click, buy, watch) or not.

We assume that the user makes choices regarding items by initially committing to a *choice function* h that governs her choice behavior. In principle, the user is free to chose h from some predefined function class H; learning will consider finding a good $h \in H$, but the implications of the choice of H itself will play a central role in our analysis. Ideally, the user would like to choose items if and only if they are worthwhile to her; practically, her goal is to find an h for which this holds with large probability over D. For this, the user can make use of her knowledge regarding items she has already consumed, and therefore also knows their value; we model this as providing the user access to a labeled sample set $S = \{(x_i, y_i)\}_{i=1}^m$ where $x_i \sim D$ and $y_i = Y(x_i)$, which she can use for learning h.

[1] Our notions of system and user mirror practice – the system controls the platform where the interaction takes place, whereas the user is the (usually human) player using the platform.

Strategic representations. The difficulty in learning h is that user choices at test time must rely only on item *representations*, denoted $z \in \mathcal{Z}$, rather than on full item descriptions. Thus, learning considers choice functions that operate on representations, $h : \mathcal{Z} \to \{\pm 1\}$, and the challenge lies in that while choices must be made on the basis of representations z, item values are derived from their full descriptions x—of which representations reveal only partial information.

The crucial aspect of our setup is that representations are not arbitrary; rather, *representations are controlled by the system*, which can choose them strategically to promote its own goals. We model the system as acting through a *representation mapping*, $\phi : \mathcal{X} \to \mathcal{Z}$, which operates independently on any x, and can be determined in response to the user's choice of h. This mimics a setting in which a fast-acting system can infer and quickly respond to a user's (relatively fixed) choice patterns.

We assume the system's goal is to choose a ϕ_h that maximizes *expected user engagement*:

$$\mathbb{E}_{x\sim D}[\mathbb{1}\{h(\phi_h(x)) = 1\}] \tag{1}$$

Nonetheless, representations cannot be arbitrary, and we require ϕ_h to satisfy two properties. First, chosen representations must be *truthful*, meaning that $z \subseteq x$ for all x. Second, representations are subject to *cardinality constraints*, $k_1 \le |z| \le k_2$ for some predetermined $k_1, k_2 \in \mathbb{N}$. Both requirements stem from realistic considerations: An untruthful system which intentionally distorts item information is unlikely to be commercially successful in the long run; intuitively, truthfulness gives users some hope of resilience to manipulation. For k_1, k_2, we think of these as exogenous parameters of the environment, arising naturally due to physical restrictions (e.g., screen size) or cognitive considerations (e.g., information processing capacity); if $k_2 < n$, we say representations are *lossy*.

Under these constraints, the system can optimize Eq. (1) by choosing representations via the *best-response mapping*:

$$\phi_h(x) = \operatorname*{argmax}_{z\in\mathcal{Z}} h(z) \quad \text{s.t.} \quad z \subseteq x, |z| \in [k_1, k_2] \tag{2}$$

Eq. (2) is a best-response since it maximizes Eq. (1) for any given h: for every x, ϕ_h chooses a feasible $z \subseteq x$ that triggers a positive choice event, $h(z) = 1$—if such a z exists. In this way, k_1, k_2 control how much leeway the system has in revealing only partial truths; as we will show, both parameters play a key role in determining outcomes for both system and user. From now on we overload notation and by $\phi_h(x)$ refer to this best-response mapping.

Learning objective. Given function class H and a labeled sample set S, the user aims to find a choice function $h \in H$ that correctly identifies worthwhile items given their representation, and in a way that is robust to strategic system manipulation. The user's objective is therefore to maximize:

$$\mathbb{E}_{x\sim D}[\mathbb{1}\{h(\phi_h(x)) = y\}] \tag{3}$$

where ϕ_h is the best-response mapping in Eq. (2).

Note that since h is binary, the argmax of ϕ_h may not be unique; e.g., if some $z_1 \subseteq x, z_2 \subseteq x$ both have $h(z_1) = h(z_2) = 1$. Nonetheless, the particular choice of z does not matter—from the user's perspective, her choice of h is invariant to the system's choice of best-response z:

Observation 2.1. *Every best-response $z \in \phi_h(x)$ induces the same value in the user's objective function (Eq. (3)).*

2.2 User types

Our main focus throughout the paper will be on users that learn h by optimizing Eq. (3). But to understand the potential benefit of learning, we also analyze 'simpler' types of user behavior. Overall, we study three user types, varying in their sophistication and the amount of effort they invest in choosing h. These include:

- **The naïve user:** Acts under the (false) belief that representations are chosen in her own best interest. This user type truthfully reports her preferences to the system by setting $h = v$ as her choice function.[2]
- **The agnostic user:** Makes no assumptions about the system. This user employs a simple strategy that relies on basic data statistics which provides minimal but robust guarantees regarding her payoff.
- **The strategic user:** Knows that the system is strategic, and anticipates it to best-respond. This user is willing to invest effort (in terms of data and compute) in learning a choice function h that maximizes her payoff by accounting for the system's strategic behavior.

Our primary goal is to study the balance of power between users (that choose) and the system (which represents). In particular, we will be interested in exploring the tradeoff between a user's effort and her susceptibility to manipulation.

2.3 Strategic representation as a game

Before proceeding, we give an equivalent characterization of strategic representation as a game. Our setting can be compactly described as a single-step Stackelberg game: the first player is User, which observes samples $S = \{(x_i, y_i)\}_{i=1}^m$, and commits to a choice function $h : \mathcal{Z} \to \{\pm 1\}$; the second player is System, which given h, chooses a truthful $\phi_h : \mathcal{X} \to \mathcal{Z}$ (note how ϕ_h depends on h). The payoffs are:

$$\textbf{User:} \quad \mathbb{E}_{x\sim D}[\mathbb{1}\{h(\phi_h(x)) = y\}] \tag{4}$$

$$\textbf{System:} \quad \mathbb{E}_{x\sim D}[\mathbb{1}\{h(\phi_h(x)) = 1\}] \tag{5}$$

Note that payoffs differ only in that User seeks *correct* choices, whereas System benefits from *positive* choices. This reveals a clear connection to strategic classification, in which System, who plays first, is interested in *accurate* predictions, and for this it can learn a classifier; and User, who plays second, can manipulate individual inputs (at some cost) to obtain *positive* predictions. Thus, strategic representation can be viewed as a variation on strategic classification, but with roles 'reversed'. Nonetheless, and despite these structural similarities, strategic representation bears key differences: items are discrete (rather than continuous), manipulations are subject to 'hard' set constraints (rather than 'soft', continuous costs), and learning regards set functions (rather than vector functions). These differences lead to distinct questions and unique challenges in learning.

[2]Note that while h takes values in $\mathcal{X}$, v takes values in $\mathcal{X}$. Nonetheless, truthfulness implies that $\mathcal{Z} \subseteq \mathcal{X}$, and so v is well-defined as a choice function over $\mathcal{Z}$.

3 WARM-UP: NAÏVE AND AGNOSTIC USERS

The naïve user. The *naïve user* employs a 'what you see is what you get' policy: given a representation of an item, z, this user estimates the item's value based on z alone, acting 'as if' z were the item itself. Consequently, the naïve user sets $h(z) = \text{sign}(v(z))$, even though v is truly a function of x. The naïve user fails to account for the system's strategic behavior (let alone the fact that $z \subseteq x$ of some actual x).

Despite its naivety, there are conditions under which this user's approach makes sense. Our first result shows that the naïve policy is sensible in settings where the system is *benevolent*, and promotes user interests instead of its own.

Lemma 3.1. *If system plays the* benevolent *strategy:*

$$\phi_h^{\text{benev}}(x) = \underset{z \subseteq x, |z| \in [k_1, k2]}{\text{argmax}} \{\mathbb{1}\{h(z) = \text{sign}(v(x))\},$$

then the naïve approach maximizes user payoff.

Proof in Appendix D. The above lemma is not meant to imply that naïve users *assume* the system is benevolent; rather, it justifies why users having this belief might act in this way. Real systems, however, are unlikely to be benevolent; our next example shows a strategic system can easily manipulate naïve users to receive arbitrarily low payoff.

Example 1. Let $x_1 = \{a_1\}, x_2 = \{a_1, a_2\}, x_3 = \{a_2\}$ with $v(x_1) = +1$ and $v(x_2) = v(x_3) = -1$. Fix $k_1 = k_2 = 1$, and let $D = (\varepsilon/2, 1-\varepsilon, \varepsilon/2)$. Note $\mathcal{Z} = \{a_1, a_2\}$ are the feasible representations. The naïve user assigns $h = (a_1) = +1, h(a_2) = -1$ according to v. For x_2, a strategic system plays $\phi(x_2) = a_1$. The expected payoff to the user is ε.

One reason a naïve user is susceptible to manipulation is because she does not make any use of the data she may have. We next describe a slightly sophisticated user that uses a simple strategy to ensure better payoff.

The agnostic user. The *agnostic user* puts all faith in data; this user does not make assumptions on, nor is she susceptible to, the type of system she plays against. Her strategy is simple: collect data, compute summary statistics, and choose to either always accept or always reject (or flip a coin). In particular, given a sample set $S = \{(x_i, y_i)\}_{i=1}^m$, the agnostic user first computes the fraction of positive examples, $\hat{\mu} := \frac{1}{m}\sum_{i=1}^m y_i$. Then, for some tolerance τ, sets for *all* z, $h(z) = 1$ if $\hat{\mu} \geq 1/2 + \tau$, $h(z) = -1$ if $\hat{\mu} \leq 1/2 - \tau$, and flips a coin otherwise. In Example 1, an agnostic user would choose $h = (-1, -1)$ when m is large, guaranteeing a payoff of at least $\frac{\sqrt{m}(1-\varepsilon/2)}{2+\sqrt{m}} \to (1 - \varepsilon/2)$ as $m \to \infty$. Investing minimal effort, for an appropriate choice of τ, this user's strategy turns out to be quite robust.

Theorem 3.2. *(Informal) Let μ be the true rate of positive examples, $\mu = \mathbb{E}_D[Y]$. Then as m increases, the agnostic user's payoff approaches $\max\{\mu, 1-\mu\}$ at rate $1/\sqrt{m}$.*

Formal statement and proof in Appendix A.1. In essence, the agnostic user guarantee herself the 'majority' rate with rudimentary usage of her data, and in a way that does not depend on how system responds. But this is can be far from optimal; we now turn to the more elaborate *strategic user* who makes more clever use of the data at her disposal.

4 STRATEGIC USERS WHO LEARN

A strategic agent acknowledges that the system is strategic, and anticipates that representations are chosen to maximize her own engagement. Knowing this, the strategic user makes use of her previous experiences, in the form of a labeled data set $S = \{(x_i, y_i)\}_{i=1}^m$, to learn a choice function $\hat{h}$ from some function class H that optimizes her payoff (given that the system is strategic). Cast as a learning problem, this is equivalent to minimizing the expected classification error on strategically-chosen representations:

$$h^* = \underset{h \in H}{\text{argmin}}\, \mathbb{E}_D[\mathbb{1}\{h(\phi_h(x)) \neq y\}]. \tag{6}$$

Since the distribution D is unknown, we follow the conventional approach of empirical risk minimization (ERM) and optimize the empirical analog of Eq. (6):

$$\hat{h} = \underset{h \in H}{\text{argmin}} \frac{1}{m} \sum_{i=1}^m h(\phi_h(x_i)) \neq y_i). \tag{7}$$

Importantly since every $z_i = \phi_h(x_i)$ is a set, H must include *set functions* $h : \mathcal{Z} \to \{\pm 1\}$, and any algorithm for optimizing Eq. (7) must take this into account. In Sections 4.1 and 4.2, we characterize the complexity of a user's choice function and relate its complexity to that of v, and in Section 4.3 give an algorithm that computes $\hat{h}$, the empirical minimizer, for a hypothesis class of a given complexity.

4.1 Complexity classes of set functions

Ideally, a learning algorithm should permit flexibility in choosing the complexity of the class of functions it learns (e.g., the degree of a polynomial kernel, the number of layers in a neural network), as this provides means to trade-off running time with performance and to reduce overfitting. In this section we propose a hierarchy of set-function complexity classes that is appropriate for our problem.

Throughout we will denote by $\Gamma_k(z)$ all subsets of z having size at most k, i.e:

$$\Gamma_k(z) = \{z' \in 2^E \,:\, z' \subseteq z, |z'| \leq k\}.$$

We start by defining k order functions over the representation space. These functions are completely determined by weights placed on subsets of size at most k (see also [6] and Sec. 1.1).

Definition 4.1. We say the function $h : \mathcal{Z} \to \{\pm 1\}$ is of *k-order* if there exists real weights on sets of cardinality at most k, $\{w(z') : z' \in \Gamma_k(z)\}$, such that

$$h(z) = \text{sign}\left(\sum_{z' \in \Gamma_k(z)} w(z')\right).$$

It is possible that a function $h : \mathcal{Z} \to \{-1, 1\}$ may not be a k-order function for any $k \leq k_2$. But in the context of optimizing Eq. (7), we show that working with k-order functions is sufficiently general. That is, for learning purposes, any set function h can be linked to a matching k-order function h' (for some k) through how it operates on strategic inputs.

Lemma 4.2. *For any $h : \mathcal{Z} \to \{\pm 1\}$, there exists $k \leq k_2$ and a corresponding k-order function h' such that:*

$$h(\phi_h(x)) = h'(\phi_{h'}(x)).$$

Proof in Appendix E. Lemma 4.2 shows that, for learning purposes, any set function h can be linked to a matching k-order function h' (for some k) through how it operates on strategic inputs. We henceforth focus on k-order functions. The proof of Lemma 4.2 is constructive, and the construction itself turns out to be highly useful. In particular, the proof constructs h' having a particular form of *binary* basis weights, $w(z) \in \{a_-, a_+\}$, which we assume from now on are fixed (for every k). Hence, every function h has a corresponding binary-weighted k-order function h', which motivates the following definition of functions and function classes.

Definition 4.3. We say a k-order function h with basis weights $\boldsymbol{w}$ is *binary-weighted* if:

$$w(z) \begin{cases} \in \{a_-, a_+\} & \forall z = k \\ = a_- & \forall z < k \end{cases}$$

for some fixed choice of $a_- \in (-1, 0)$ and $a_+ > \sum_{i\in[k]} \binom{n}{i}$, and denote the class of all binary-weighted k-order functions as:

$$H_k = \{h \,:\, h \text{ is a binary-weighted } k\text{-order function}\}.$$

The classes $\{H_k\}_k$ will serve as complexity classes for our learning algorithm; the user provides k as input, and Alg outputs an $\hat{h} \in H_k$ that minimizes the empirical loss[3]. As we will show, using k as a complexity measure provides the user direct control over the tradeoff between estimation and approximation error, as well as over the running time.

Next, we show that the $\{H_k\}_k$ classes are strictly nested. This will be important for our analysis of approximation error, as it will let us reason about the connection between the learned $\hat{h}$ and the target function v (proof in Appendix E).

Lemma 4.4. *For all k, $H_{k-1} \subseteq H_k$ and $H_k \setminus H_{k-1} \neq \emptyset$.*

Denote $Z_\ell = \{z \in 2^E \,:\, |z| = \ell\}$, and note that all feasible representations can be partitioned as $\mathcal{Z} = Z_{k_1} \uplus \ldots \uplus Z_{k_2}$. We refer to functions that operate on single-sized sets as *restricted functions*. Our next result shows that choice functions in H_k can be represented by restricted functions over Z_k that are 'lifted' to operate on the entire $\mathcal{Z}$ space. This will allow us to work only with sets of size exactly k in the algorithm given in Sec. 4.3.

Lemma 4.5. *For each $h \in H_k$ with weights $\boldsymbol{w}$ there exists a corresponding $g : Z_k \to \{\pm 1\}$ such that $h = \text{lift}(g)$, where:*

$$\text{lift}(g)(z) = \begin{cases} 1 & \text{if } k \leq |z| \text{ and} \\ & \exists z' \subseteq z, |z'| = k \text{ s.t. } g(z') = 1 \\ -1 & \text{o.w.} \end{cases}$$

Further, for all $z \in Z_k$ $g(z) = 1$ if $w(z) = a_+ > 0$, and $g(z) = -1$ otherwise.

Proof. Let $h \in H_k$ with weight function $\boldsymbol{w}$. Since $h \in H_k$, $\boldsymbol{w}$ defines weights over sets of size at most k, such that either $w(z) \in (-1, 0)$ or $w(z) > \sum_{i\in[k]} \binom{n}{i}$, and

$$h(z) = \text{sign}\left(\sum_{z'\in\Gamma_k(z)} w(z')\right).$$

[3]Assuming the empirical error is zero.

Define $g : Z_k \to \{-1, 1\}$ such that for a $z \in Z_k$, $g(z) = 1$ if $w(z) > 0$ and $g(z) = -1$ otherwise. From the definition of H_k and $\text{lift}(g)$, for all $z \in \mathcal{Z}$ such that $|z| < k$, $h(z) = \text{lift}(g)(z) = -1$. Further, for all $z \in \mathcal{Z}$ such that $|z| \geq k$,

$$\begin{aligned} h(z) = 1 &\iff \exists z' \subseteq z \text{ such that } |z'| = k \text{ and } w(z') = a_+ > 0 \\ &\quad \text{(from the choice of } a_+) \\ &\iff \exists z' \subseteq z \text{ such that } |z'| = k \text{ and } g(z') = 1 \\ &\quad \text{(from the definition of } g) \\ &\iff \text{lift}(g)(z) = 1 \text{ (from the definition of lift}(g)). \end{aligned}$$

□

Note that H_n includes all binary-weighted set functions, but since representations are of size at most k_2, it suffices to consider only $k \leq k_2$. Importantly, k can be set lower than k_1; for example, H_1 is the class of threshold modular functions, and H_2 is the class of threshold pairwise functions. The functions we consider are parameterized by their weights, $\boldsymbol{w}$, and so any k-order function has at most $|\boldsymbol{w}| = \sum_{i=0}^{k} \binom{q}{i}$ weights. In this sense, the choice of k is highly meaningful. Now that we have defined our complexity classes, we turn to discussing how they can be optimized over.

4.2 Learning via reduction to induced functions

The simple structure of functions in H_k makes them good candidates for optimization. But the main difficulty in optimizing the empirical error in Eq. (7) is that the choice of h does not only determine the error, but also determines the inputs on which errors are measured (indirectly through the dependence of ϕ_h on h). To cope with this challenge, our approach is to work with *induced* functions that already have the system's strategic response encoded within, which will prove useful for learning. Additionally, as they operate directly on x (and not z), they can easily be compared with v, which will become important in Sec. 5.

Definition 4.6. For a class H, its *induced class* is:

$$F_H \triangleq \{f : \mathcal{X} \to \{\pm 1\} \,:\, \exists h \in H \text{ s.t. } f(x) = h(\phi_h(x))\}$$

The induced class F_H includes for every $h \in H$ a corresponding function that already has ϕ_h integrated in it. We use $F_k = F_{H_k}$ to denote the induced class of H_k. For every h, we denote its induced function by f_h. Whereas h functions operate on z, induced functions operate directly on x, with each f_h accounting internally for how the system strategic responds to h on each x.

Our next theorem provides a key structural result: induced functions inherit the weights of their k-order counterparts.

Theorem 4.7. *For any $h \in H_k$ with weights $\boldsymbol{w}$:*

$$h(z) = \text{sign}\left(\sum_{z'\in\Gamma_k(z)} w(z')\right),$$

its induced $f_h \in F_k$ can be expressed using the same weights, $\boldsymbol{w}$, but with summation over subsets of x, i.e.:

$$f_h(x) = \text{sign}\left(\sum_{z\in\Gamma_k(x)} w(z)\right).$$

Proof. Since $h \in H_k$, $\boldsymbol{w}$ satisfies the following two properties (see Definition 4.3):

(1) either $w(z) = a_- \in (-1, 0)$ or $w(z) = a_+ > \sum_{i\in[k]} \binom{n}{i}$,

(2) $w(z) = a_-$ for all z having $|z| < k$.

Further, from the definition of f_h, we have $f_h(x) = h(\phi_h(x))$. This implies

$$f_h(x) = 1 \iff \exists z \in \mathcal{Z},\ z \subseteq x \text{ such that } h(z) = 1\,.$$

From the the above two properties of the weights function, we have

$$h(z) = 1 \iff \exists z' \subseteq z,\ |z'| = k \text{ such that } w(z') = a_+ > 0\,.$$

From the above two equations we conclude that

$$f_h(x) = 1 \iff \exists\, z \subseteq x, |z| = k \text{ such that } w(z) = a_+ > 0\,.$$

Finally, the two properties of $\boldsymbol{w}$ ensure that

$$f_h(x) = \text{sign}\left(\sum\nolimits_{z \in \Gamma_k(x)} w(z)\right)\,.$$

□

Theorem 4.7 is the main pillar on which our algorithm stands: it allows us to construct h by querying the loss *directly*—i.e., without explicitly computing ϕ_h—by working with the induced f_h; this is since:

$$\mathbb{1}\{h(\phi_h(x_i)) \neq y_i\} = \mathbb{1}\{f_h(x_i) \neq y_i\}$$

Thus, through their shared weights, induced functions serve as a bridge between *what* we optimize, and *how*.

4.3 Learning algorithm

We now present our learning algorithm, ALG. The algorithm is exact: it takes as input a training set S and a parameter k. and assuming Y is realizable by an induced function of an $h \in H_k$, that is, $Y \in F_k$, returns an $h \in H_k$ that minimizes the empirical loss (Eq. (7)). If $Y \in F_k$ then for brevity we say Y is realizable. The algorithm constructs h by sequentially computing its weights, $\boldsymbol{w} = \{w(z)\}_{|z| \leq k}$. As per Def. 4.3, only $w(z)$ for z with $|z| = k$ must be learned; hence, weights are sparse, in the sense that only a small subset of them are assigned a_+, while the rest are a_-. Weights can be implemented as a hash table, where $w(z) = a_+$ if z is in the table, and $w(z) = a_-$ if it is not. We prove the correctness of ALG in Theorem 4.8. The proof leverages a property that characterizes the existence of an $h \in H_k$ having zero empirical error (see Lemma E.1 in the proof of Theorem 4.8). The proof of Lemma E.1 uses Lemma 4.5 and Theorem 4.7; Lemma 4.5 is used to connect functions in H_k with functions operating over size k subsets via the lifting operation, and Theorem 4.7 enables the loss to be directly computed for the induced functions using the shared weight structure.

Theorem 4.8. *For any $k \in [k_2]$, if Y is realizable then ALG returns an $\hat{h}$ that minimizes the empirical error.*

Proof in Appendix E. Note that our algorithm is exact: it returns a true minimizer of the empirical 0/1 loss, assuming Y is realizable. Additionally, ALG can be used to identify if there exists an $h \in H_k$ with zero empirical error; at Step 15, for each $x \in S^+$ if there does not exist a $z \in Z_{k,S}$ or $z \in Z^+$ such that $z \subseteq x$ then from Lemma E.1 in Appendix D there does not exist an h with zero empirical error. We note that even for linear classification with 0/1 loss the empirical minimizer can be learnt via solving a linear program if the data is realizable and is NP-hard otherwise [18].

Algorithm: ALG

1 **Input**: $S = \{(x_i, y_i)\}_{i \in [m]}$, $k \in [k_2]$
2 **Pre-compute:**
3 $S^+ = \{x \in S : y = +1\}$, $S^- = \{x \in S : y = -1\}$
4 $Z_{k,S} = \{z : |z| = k, \exists x \in S\ z \subseteq x\}$
5 $\hat{p}(x_i) = \frac{1}{m} \sum_{j \in [m]} \mathbb{1}\{x_i = x_j\} \quad \forall i \in [m]$
6 Fix $a_- \in (-1, 0)$ and $a_+ > \sum_{i \in [1,k]} \binom{n}{i}$
7 **Initialize:**
8 $Z^+ = \emptyset, Z^- = \emptyset, V = \emptyset, \ \ S_z = \emptyset \quad \forall z \in Z_{k,S}$
9 **Run:**
10 **for** $x \in S^-$ **do**
11 **for** z s.t. $z \subseteq x$ and $z \in Z_{k,S}$ **do**
12 $Z^- = Z^- \cup \{z\}, \quad Z_{k,S} = Z_{k,S} \setminus \{z\}$
13 $S_z = S_z \cup \{x\}$
14 **for** $x \in S^+$ **do**
15 **for** $z \subseteq x$ such that $z \in Z_{k,S}$ **do**
16 $Z^+ = Z^+ \cup \{z\}$
17 Set $w(z) = \begin{cases} a_+ & \text{if } z \in Z^+ \\ a_- & \text{o.w. (implicitly)} \end{cases}$ ▷ *implemented as hash table*
18 **return** $\hat{h}(z) = \text{sign}(\sum_{z' \in \Gamma_k(z)} w(z'))$.

Lemma 4.9. *Let n be the size of elements in $\mathcal{X}$, m be the number of samples that the user has, and $k \leq k_2$ be the user's choice of complexity. Then ALG runs in $O(m\binom{n}{k})$ time.*

PROOF. In the first two for loops, for each $x \in S^+$ (or in S^-) the internal for loop runs for $O(\binom{n}{k})$ time. Since $|S| \leq m$, this is a total of at most $O(m\binom{n}{k})$ operations. Similarly, step 20 places weights on at most $m\binom{n}{k}$ subsets, and hence runs in $O(m\binom{n}{k})$ time. Hence, ALG runs in $O(m\binom{n}{k})$ time. □

This runtime is made possible due to several key factors: (i) only k-sized weights need to be learned, (ii) all weights are binary-valued, and (iii) loss queries are efficient in induced space. Nonetheless, when n and k are large, runtime may be significant, and so k must be chosen with care. Fortunately, our results in Sec. 5.1.1 give encouraging evidence that learning with small k—even $k = 1$, for which runtime is $O((mn)^2)$—is quite powerful[4].

In the analysis, the $m\binom{n}{k}$ is made possible only since weights are sparse, and since ALG operates on a finite sample set of size m. Alternatively, if m is large, then this expression can be replaced with $\binom{q}{k}$. This turns out to be necessary; in Appendix A.2 we show that, in the limit, $\binom{q}{k}$ is a lower bound.

5 BALANCE OF POWER

Our final section explores the question: what determines the balance of power between system and users? We begin with the perspective of the user, who has commitment power, but can only minimize the empirical error for any Y on the available sample data. For her, the choice of complexity class k is key in balancing approximation error—how well (in principle) can functions $h \in H_k$ approximate v; and estimation error—how close the empirical payoff of the learned

[4] Assuming Y is realizable.

$\hat{h}$ is to its expected value. Our results give insight into how these types of error trade off as k is varied.

For the system, the important factors are k_1 and k_2, since these determine its flexibility in choosing representations. Since more feasible representation mean more flexibility, it would seem plausible that smaller k_1 and larger k_2 should help the system more. However, our results indicate differently: for system, *smaller* k_2 is better, and the choice of k_1 has limited effect on strategic users. The result for k_2 goes through a connection to the user's choice of k; surprisingly, smaller k turns out to be, in some sense, better for all.

5.1 User's perspective

We begin by studying the effects of k on user payoff. Recall that users aim to minimize the expected error (Eq. (6)):

$$\varepsilon(h) = \mathbb{E}_D[\mathbb{1}\{h(\phi_h(x)) \neq \text{sign}(v(x))\}],$$

but instead minimize the empirical error (Eq. (7)). For reasoning about the expected error of the learned choice function $\hat{h} \in H_k$, a common approach is to decompose it into two error types—*approximation* and *estimation*:

$$\varepsilon(\hat{h}) = \underbrace{\varepsilon(h^*)}_{\text{approx.}} + \underbrace{\varepsilon(\hat{h}) - \varepsilon(h^*)}_{\text{estimation}}, \qquad h^* = \operatorname*{argmin}_{h' \in H_k} \varepsilon(h')$$

Approximation error describes the lowest error obtainable by functions in H_k; this measures the 'expressivity' of H_k, and is independent of $\hat{h}$. For approximation error, we define a matching complexity structure for value functions v, and give several results relating the choice of k and the complexity of v. Estimation error describes how far the learned $\hat{h}$ is from the optimal $h^* \in H_k$, and depends on the data size, m. Here we give a generalization bound based on VC analysis.

5.1.1 User approximation error. To analyze the approximation error, we must be able to relate choice functions h (that operate on representations z) to the target value function v (which operates on items x). To connect the two, we will again use induced functions, for which we now define a matching complexity structure.

Definition 5.1. A function $f : \mathcal{X} \to \{\pm 1\}$ has an *induced complexity* of ℓ if exists a function $g : Z_\ell \to \{\pm 1\}$ s.t.:

$$f(x) = \begin{cases} 1 & \text{if } \exists z \subseteq x, |z| = \ell \text{ and } g(z) = 1 \\ -1 & \text{o.w.} \end{cases}$$

and ℓ is minimal (i.e., there is no such $g' : Z_{\ell-1} \to \{\pm 1\}$).

We show in Lemma 5.2 and Corollary 5.3 that the induced complexity of a function f captures the minimum $k \in [1, n]$ such that f is an induced function of an $h \in H_k$.

Lemma 5.2. *Let $k \leq k_2$. Then for every $h \in H_k$, the induced complexity of the corresponding f_h is $\ell \leq k$.*

Proof. Let $\ell = \min_{k' \in [1,k]}\{\text{there exists a } g : Z_\ell \to \{\pm 1\} \text{ such that } h = \text{lift}(g)\}$. From Lemma 4.5, we know $\ell \leq k$. Further, assume $g : Z_\ell \to \{\pm 1\}$ is such that $h = \text{lift}(g)$. Now, from the definition of f_h, we have for all $x \in \mathcal{X}$, $f_h(x) = 1$ if and only if there exists a $z \in \mathcal{Z}$ such that $z \subseteq x$ and $h(z) = 1$. Since $h = \text{lift}(g)$, $f_h(x) = 1$ if and only if there exists a $z \in Z_\ell$ such that $z \subseteq x$ and $g(z) = 1$. This implies the induced complexity of f_h is $\ell \leq k$. □

Corollary 5.3. *Let $F_k = F_{H_k}$ be the induced function class of H_k, as defined in Def. 4.6. Then:*

$$F_k = \{f : \mathcal{X} \to \{\pm 1\} \ : \ f \text{ has induced complexity } \leq k\}.$$

Proof. From Lemma 5.2, we know that functions in F_k have induced complexity at most k. We show that if f has induced complexity at most k then there is an $h \in H_k$ such that $f = f_h$. Let the induced complexity of f be equal to $\ell \leq k$. Then there exists a $g : Z_\ell \to \{\pm 1\}$ such that

$$f(x) = 1 \iff \exists z \in Z_\ell \text{ such that } z \subseteq x \text{ and } g(z) = 1. \quad (8)$$

Let $h = \text{lift}(g)$. First we show that $f(x) = f_h(x)$ for all $x \in \mathcal{X}$. Since h is a lift of g, if $g(z') = 1$ for a $z' \in Z_\ell$ then for all $z \in \mathcal{Z}$ such that $z' \subseteq z$, we have $h(z) = 1$.

$$h(z) = 1 \iff \exists z' \in Z_\ell \text{ such that } z' \subseteq z \text{ and } g(z') = 1. \quad (9)$$

Hence, from Equations 8 and 9 for all $x \in \mathcal{X}$, $f(x) = 1$ if and only if there exists $z \in \mathcal{Z}$ such that $z \subseteq x$ and $h(z) = 1$. From the definition of induced function, this implies $f(x) = f_h(x)$ for all $x \in \mathcal{X}$.

To show $h \in H_k$, we construct a weight function $\boldsymbol{w}$ on sets of size at most k. For $z \in 2^E$ and $|z| < k$, let $w(z) = a_- \in (-1, 0)$. For $z \in Z_k$, let

$$w(z) \begin{cases} = a_+ > \sum_{i \in [1,k]} \binom{n}{i} & \text{if } \exists z' \subseteq z, |z'| = \ell \text{ and } g(z) = 1 \\ = a_- & \text{o.w.} \end{cases}$$

Now from Equation 9, $h(z) = 1$ if and only if there exists $z' \in Z_\ell$ such that $z' \subseteq z$ and $g(z') = 1$. Hence, from the definition of $\boldsymbol{w}$, $h(z) = 1$ if and only if there exists $z' \in Z_\ell$ such that $z' \subseteq z$ and $w(z') = a_+$. In particular, since $a_+ > \sum_{i=1}^{k} \binom{n}{i}$ and $a_- \in (0, 1)$, we have

$$h(z) = \text{sign}\left(\sum_{z' : z' \subseteq z, |z'| \leq k} w(z')\right).$$

□

We can now turn to considering the effect of k on approximation error. Since the 'worthwhileness' function $Y(x) = \text{sign}(v(x))$ operates on x, we can consider its induced complexity, which we denote by ℓ^* (i.e., $Y \in F_{\ell^*}$). The following result shows that if $\ell^* \leq k$, then H_k is expressive enough to perfectly recover Y.

Theorem 5.4. *If $\ell^* \leq k$ then the approximation error is* 0.

Proof. Since the induced complexity of v is ℓ^*, there is a function $g : Z_{\ell^*} \to \{\pm 1\}$ s.t.:

$$v(x) = \begin{cases} 1 & \text{if } \exists z \subseteq x, |z| = \ell^* \text{ and } g(z) = 1 \\ -1 & \text{o.w.} \end{cases}$$

Let $a_+ > \sum_{i \in [1,k]} \binom{n}{i}$ and $a_- \in (0, 1)$, and define the weight function $\boldsymbol{w}$ on sets of size at most k as follows: a) if $|z| < k$ then let $w(z) = a_-$, b) if $|z| = k$ and there exists a $z' \subseteq z$ such that $g(z') = 1$ then $w(z) = a_+$, and c) if $|z| = k$ and there does not exist a $z' \subseteq z$ such that $g(z') = 1$ then $w(z) = a_-$. Now define h using $\boldsymbol{w}$ as follows:

$$h(z) = \text{sign}\left(\sum_{z' \in \Gamma_k(z)} w(z')\right).$$

We now show that for each $x \in \mathcal{X}$, $h(\phi_h(x)) = f_h(x) = v(x)$ implying $h_k^* = h$. Suppose $f_h(x) = 1$ for an $x \in \mathcal{X}$. Then there exists a $z \in \mathcal{Z}$ such that $z \subseteq x$ and $h(z) = 1$. From Theorem 4.7, and the choice of a_+ and a_- we have that there exists a $z \subseteq x$, $|z| = k$ such

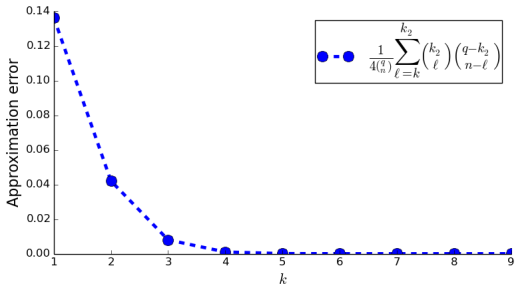

Figure 1: Upper bound on approximation error showing diminishing returns. Parameters: $q = 400, n = 30$ **and** $k_2 = 10$**.**

that $w(z) = a_+$. From the construction of w this implies there exists a $z \subseteq x$, $|z| = \ell^*$ such that $g(z) = a_+$. But from the above definition of v this implies $v(x) = 1$. Similarly, we can argue, if $f_h(x) = -1$ then $v(x) = -1$ for any $x \in \mathcal{X}$. Hence, $h(\phi_h(x)) = v(x)$ for each $x \in \mathcal{X}$ implying $h^*_k = h$ and zero approximation error for h. □

One conclusion from Theorem 5.4 is that if the user knows ℓ^*, then zero error is, in principle, obtainable; another is that there is no reason to choose $k > \ell^*$. In practice, knowing ℓ^* can aid the user in tuning k according to computational (Sec. 4.3) and statistical considerations (Sec. 5.1.2). Further conclusions relate ℓ^* and k_2:

Corollary 5.5. *If $\ell^* \le k_2$ and the distribution D has full support on $\mathcal{X}$, then $k = \ell^*$ is the smallest k that gives zero approximation error.*

Corollary 5.6. *If $\ell^* > k2$, then the approximation error weakly increases with k, i.e., $\varepsilon(h^*_k) \le \varepsilon(h^*_{k-1})$ for all $k \le k_2$. Furthermore, if the distribution D has full support on $\mathcal{X}$ then no k can achieve zero approximation error.*

Proofs in Appendix F. In general, Corollary 5.6 guarantees only weak improvement with k. Next, we show that increasing k can exhibit clear diminishing-returns behavior, with most of the gain obtained at very low k.

Lemma 5.7. *Let D be the uniform distribution over $\mathcal{X}$. Then there is a value function v for which $\varepsilon(h^*_k)$ diminishes convexly with k.*

The proof is constructive and given in Appendix F. We construct a v such that the approximation error $h^*_k \in H_k$ is upper bounded by

$$\varepsilon(h^*_k) \le \frac{1}{4\binom{q}{n}} \sum_{\ell=k}^{k_2} \binom{k_2}{\ell}\binom{q-k_2}{n-\ell}.$$

The diminishing returns property of upper bound is illustrated in Fig. 1. Although Lemma 5.7 describes a special case, we conjecture that the phenomena of diminishing returns applies more broadly.

Our next result shows that learning k_1-order functions can be as powerful as learning subadditive functions; hence, learning with $k = k_1$ is highly expressive. Interestingly, the connection between general (k-order) functions and class of subadditive functions is due to the strategic response mapping, ϕ.

Lemma 5.8. *Consider threshold-subadditive functions:*

$$H_{\mathrm{SA}} = \{\mathrm{sign}(g(z)) \,:\, g \text{ is subadditive on subsets in } \mathcal{Z}\}$$

Then for every threshold-subadditive $h_g \in H_{\mathrm{SA}}$, there is an $h \in H_{k_1}$ for which $h(\phi_h(x)) = h_g(\phi_{h_g}(x))\ \forall x \in \mathcal{X}$.

Proof. Let $h \in H_{\mathrm{SA}}$ with a corresponding $g : \mathcal{Z} \to \mathbb{R}$ such that $h(z) = \mathrm{sign}(g(z))$ for all $z \in \mathcal{Z}$. Choose an $a_+ > \sum_{i=1}^{k_1}\binom{n}{i}$, and $a_- \in (0,1)$. Define a weight function $\boldsymbol{w}$ on sets of size at most k_1 as follows:

$$w(z) = \begin{cases} a_+ & \text{if } |z| = k_1,\ h(z) = 1 \\ a_- & \text{o.w.} \end{cases}$$

Let $h' \in H_{k_1}$ be the function defined by the binary weight $\boldsymbol{w}$ as defined above. We argue that for every $x \in \mathcal{X}$, if $h(\phi_h(x)) = h'(\phi_{h'}(x))$. For every $x \in \mathcal{X}$, $h(\phi_h(x)) = 1$ if and only if there is a $z \in \mathcal{Z}$ and $z \subseteq x$ such that $h(z) = 1$. Since g is sub-additive, we have

$$0 \le g(z) \le \sum_{z' \subseteq z, z' \neq z, z' \in \mathcal{Z}} g(z')\,. \tag{10}$$

A simple recursive argument implies $h(\phi_h(x)) = 1$ if and only if there is a $z \subseteq x$ such that $|z| = k_1$ and $h(z) = 1$, and hence $w(z) = a_+$. Hence, from Theorem 4.7 this implies, $h(\phi_h(x)) = 1$ if and only if $h'(\phi_{h'}(x)) = 1$. □

5.1.2 User estimation error. For estimation error, we give generalization bounds based on VC analysis. The challenge in analyzing functions in H_k is that generalization applies to the *strategic* 0/1 loss, i.e., $\mathbb{1}\{h(\phi_h(x)) \neq y\}$, and so standard bounds (which apply to the standard 0/1 loss) do not hold. To get around this, our approach relies on directly analyzing the VC dimension of the induced class, F_k (a similar approach was taken in Sundaram et al. [19] for SC). This allows us to employ tools from VC theory, which give the following bound.

Theorem 5.9. *For any k and m, given a sample set S of size m sampled from D and labeled by some v, we have*

$$\varepsilon(\hat{h}) - \varepsilon(h^*) \le \sqrt{\frac{C(\binom{q}{k}\log(\binom{q}{k}/\epsilon) + \log(1/\delta)}{m}}$$

w.p. at least $1-\delta$ over S, and for a fixed constant C. In particular, Alg in Sec. 4.3, assuming Y is realizable, returns an $\hat{h} \in H_k$ for which:

$$\varepsilon(\hat{h}) \le \sqrt{\frac{C(\binom{q}{k}\log(\binom{q}{k}/\epsilon) + \log(1/\delta)}{m}}$$

w.p. at least $1-\delta$ over S, and for a fixed constant C.

The proof relies on Theorem 4.7; since h and f_h share weights, the induced F_k can be analyzed as a class of q-variate degree-k multilinear polynomials. Since induced functions already incorporate ϕ, VC analysis for the 0/1 loss can be applied. Note that such polynomials have exactly $\binom{q}{k}$ degrees of freedom; hence the term in the bound.

5.2 System's perspective

The system's expressive power derives from its flexibility in choosing representations z for items x. Since k_1, k_2 determine which representations are feasible, they directly control the system's power to manipulate; and while the system itself may not have direct control over k_1, k_2 (i.e., if they are set by exogenous factors like screen size), their values certainly affect the system's ability to optimize

engagement. Our next result is therefore unintuitive: for system, a smaller k_2 is better (in the worst case), even though it reduces the set of feasible representations. This result is obtained indirectly, by considering the effect of k_2 on the user's choice of k.

Lemma 5.10. *There exists a distribution D and a value function v such that for all $k < k' \leq k_2$, system has higher payoff against the optimal $h^*_k \in H_k$ than against $h^*_{k'} \in H_{k'}$.*

The proof is in Appendix F; it uses the uniform distribution, and the value function from Theorem 5.7. Recalling that the choice of k controls the induced complexity ℓ (Corollary 5.3), and that users should choose k to be no greater than k_2, we can conclude the following:

Corollary 5.11. *For the system, lower k_2 is better (in a worst-case sense).*

Proof in Appendix F. For k_1, it turns out that against strategic users, it is entirely inconsequential. The reason is that payoff to the strategic user is derived entirely from k—which is upper-bounded by k_2, but can be set lower than k_1. This invariance is derived immediately from how functions in H_k are defined, namely that $w(z) = a_-$ for all z with $|z| < k$ (Def. 4.3). This, however, holds when the strategic user chooses to learn over H_k for some k. Consider, alternatively, a strategic user that decides to learn subadditive functions instead. In this case, Theorem 5.8 shows that k_1 determines the users 'effective' k; the smaller k_1, the smaller the subset of subadditive functions that can be learned. Hence, for user, smaller k_1 means worse approximation error. This becomes even more pronounced when facing a naïve user; for her, a lower k_1 means that system now has a large set of representations to choose from; if even one of them has $v(z) = 1$, the system can exploit this to increase its gains. In this sense, as k_1 decreases, payoff to the system (weakly) improves.

6 DISCUSSION

Our analysis of the balance of power reveals a surprising conclusion: for both parties, in some sense, simple choice functions are better. For system, lower k improves its payoff through how it relates to k_2 (Corollary 5.11). For users, lower k is clearly better in terms of runtime (Lemma 4.9) and estimation error (Theorem 5.9), and for approximation error, lower k has certain benefits—as it relates to ℓ^* (Corollary 5.5), and via diminishing returns (Theorem 5.7). Thus, and despite their conflicting interests—to some degree, the incentives of the system and its users align.

But the story is more complex. For users, there is no definitive notion of 'better'; strategic users always face a trade-off, and must choose k to balance approximation, estimation, and runtime. In principle, users are free to choose k at will; but since there is no use for $k > k_2$, a system controlling k_2 de facto controls k as well. This places a concrete restriction on the freedom of users to choose, and inequitably: for small k_2, users whose v has complexity $\leq k_2$ (i.e., having 'simple tastes') are less susceptible to manipulation than users with v of complexity $> k_2$ (e.g., fringe users with eclectic tastes) (Theorem 5.4, Corollaries. 5.5 and 5.6). In this sense, the choice of k_2 also has implications on fairness. We leave the further study of these aspects of strategic representation for future work.

From a purely utilitarian point of view, it is tempting to conclude that systems should always set k_2 to be low. But this misses the broader picture: although systems profit from engagement, users engage only if they believe it is worthwhile to them, and dissatisfied users may choose to leave the system entirely (possibly into the hands of another). Thus, the system should not blindly act to maximize engagement; in reality, it, too, faces a tradeoff.

REFERENCES

[1] Elias Abboud, Nader Agha, Nader H Bshouty, Nizar Radwan, and Fathi Saleh. 1999. Learning threshold functions with small weights using membership queries. In *Proceedings of the twelfth annual conference on Computational learning theory*. 318–322.

[2] Ittai Abraham, Moshe Babaioff, Shaddin Dughmi, and Tim Roughgarden. 2012. Combinatorial auctions with restricted complements. In *Proceedings of the 13th ACM Conference on Electronic Commerce*. 3–16.

[3] Dana Angluin. 1987. Queries and Concept Learning. *Machine Learning* 2, 4 (1987), 319–342.

[4] Maria-Florina Balcan and Nicholas JA Harvey. 2011. Learning submodular functions. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*. 793–802.

[5] Yann Chevaleyre, Ulle Endriss, Sylvia Estivie, and Nicolas Maudet. 2008. Multiagent resource allocation in k -additive domains: preference representation and complexity. *Annals of Operations Research* 163, 1 (2008), 49–62.

[6] Vincent Conitzer, Tuomas Sandholm, and Paolo Santi. 2005. Combinatorial Auctions with k-wise Dependent Valuations. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[7] Jinshuo Dong, Aaron Roth, Zachary Schutzman, Bo Waggoner, and Zhiwei Steven Wu. 2018. Strategic Classification from Revealed Preferences. In *Proceedings of the 2018 ACM Conference on Economics and Computation*. ACM.

[8] Shaddin Dughmi, Nicole Immorlica, Ryan O'Donnell, and Li-Yang Tan. 2015. Algorithmic Signaling of Features in Auction Design. In *Algorithmic Game Theory - 8th International Symposium, SAGT*. Springer, 150–162.

[9] Uriel Feige, Michal Feldman, Nicole Immorlica, Rani Izsak, Brendan Lucier, and Vasilis Syrgkanis. 2015. A unifying hierarchy of valuations with complements and substitutes. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[10] Vitaly Feldman. 2009. On the power of membership queries in agnostic learning. *The Journal of Machine Learning Research* 10 (2009), 163–182.

[11] Nika Haghtalab, Nicole Immorlica, Brendan Lucier, Markus Mobius, and Divyarthi Mohan. 2021. *Persuading with Anecdotes.* Technical Report. National Bureau of Economic Research.

[12] Moritz Hardt, Nimrod Megiddo, Christos H. Papadimitriou, and Mary Wootters. 2016. Strategic Classification. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*.

[13] Meena Jagadeesan, Celestine Mendler-Dünner, and Moritz Hardt. 2021. Alternative microfoundations for strategic classification. In *Proceedings of the 38th International Conference on Machine Learning, ICML*.

[14] Emir Kamenica and Matthew Gentzkow. 2011. Bayesian Persuasion. *American Economic Review* 101, 6 (2011).

[15] Anilesh K Krishnaswamy, Haoming Li, David Rein, Hanrui Zhang, and Vincent Conitzer. 2021. Classification with Strategically Withheld Data. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[16] John Miller, Smitha Milli, and Moritz Hardt. 2020. Strategic classification is causal modeling in disguise. In *Proceedings of the 37th International Conference on Machine Learning, ICML*. PMLR, 6917–6926.

[17] Nir Rosenfeld, Kojin Oshiba, and Yaron Singer. 2020. Predicting choice with set-dependent aggregation. In *Proceedings of the 37th International Conference on Machine Learning, ICML*.

[18] Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding machine learning: From theory to algorithms.* Cambridge university press.

[19] Ravi Sundaram, Anil Vullikanti, Haifeng Xu, and Fan Yao. 2021. PAC-learning for strategic classification. In *Proceedings of the 38th International Conference on Machine Learning, ICML*.

[20] Hanrui Zhang and Vincent Conitzer. 2021. Incentive-aware pac learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[21] Tijana Zrnic, Eric Mazumdar, Shankar Sastry, and Michael Jordan. 2021. Who Leads and Who Follows in Strategic Classification? *Advances in Neural Information Processing Systems* 34 (2021).

A ADDITIONAL RESULTS

A.1 Agnostic User

Theorem A.1 stated below is the formal version of Theorem 3.2 in Section 3. Theorem A.1 shows that given a a large enough sample size m, an agnostic user's payoff would approach $\max\{\mu, 1-\mu\}$, where $\mu = \mathbb{E}_D[y]$.

Theorem A.1. *Let $\frac{2}{2+\sqrt{m}} \le \delta < 1/8$ and $\tau = \frac{\delta}{2(1-\delta)} + \sqrt{\frac{2\log(1/\delta)}{m}}$, then agnostic user's expected payoff guarantee is given by*

$$\begin{cases} \ge (1-\delta)(1-\mu) & \text{if } \widehat{\mu} \le 1/2 - \tau \\ \ge (1-\delta)\mu & \text{if } \widehat{\mu} \ge 1/2 + \tau \\ = 1/2 & \text{Otherwise} \end{cases}$$

Before we prove the theorem, we state Hoeffeding's inequality, which is a well known result from probability theory.

Lemma A.2. *Let $S_m = \sum_{i=1}^{m} X_i$ be the sum of m i.i.d. random variables with $X_i \in [0,1]$ and $\mu = \mathbb{E}[X_i]$ for all $i \in [m]$, then*

$$\mathbb{P}\left(\frac{S_m}{m} - \mu \ge \varepsilon\right) \le e^{-2m\varepsilon^2} \quad \text{and,} \quad \mathbb{P}\left(\frac{S_m}{m} - \mu \le -\varepsilon\right) \le e^{-2m\varepsilon^2}.$$

We will use the following equivalent form of the above inequality. Let $\delta := e^{-2m\varepsilon^2}$ i.e. $\varepsilon = \sqrt{\frac{2\log(1/\delta)}{m}}$ and $\widehat{\mu} = \frac{S_m}{m}$. Then we have with probability at-least $(1-\delta)$ we have

$$\mu \le \widehat{\mu} + \sqrt{\frac{2\log(1/\delta)}{m}} \text{ and,} \tag{11}$$

$$\mu \ge \widehat{\mu} - \sqrt{\frac{2\log(1/\delta)}{m}} \tag{12}$$

Now we are ready to give the proof of Theorem A.1

Proof of Theorem A.1. We begin with the following supporting lemma.

Lemma A.3. *Let $\frac{2}{2+\sqrt{m}} \le \delta < 1/8$, then $\tau < 1/2$.*

Proof. The proof follows from following sequence of inequalities,

$$\frac{2}{2+\sqrt{m}} < \delta \iff m > 4(1/\delta - 1)^2 \implies m > 4(1/\delta - 1)\log(1/\delta) \iff \frac{\delta}{2(1-\delta)} > \frac{2\log(1/\delta)}{m}$$

Let $\gamma = \frac{\delta}{2(1-\delta)}$. We have $\tau = \gamma + \sqrt{\gamma}$ which is an increasing function of δ, so we the maximum is achieved at $\delta = 1/8$ and is given by $1/\sqrt{14} + 1/14 < 1/2$. This completes the proof of the lemma. □

From, lemma A.3 we have that $1/2 + \tau < 1$, hence there is a non-trivial range i.e. $\widehat{\mu} \in [1/2+\tau, 1]$ where user assigns $h(z) = +1$ for all z with probability 1. Similarly, when $\widehat{\mu} \in [0, 1/2-\tau]$ user assigns $h(z) = -1$ for all z with probability 1. We will consider three cases separately.

Case 1 ($\widehat{\mu} \in [1/2+\tau, 1]$): From Hoeffding's inequality (Eq. 12) we have that with probability at-least $(1-\delta)$,

$$\mu \ge \widehat{\mu} - \sqrt{\frac{2\log(1/\delta)}{m}}$$
$$\implies \mu \ge 1/2 + \frac{\delta}{2(1-\delta)} = \frac{1}{2(1-\delta)}$$

Hence, with probability at-least $(1-\delta)$ an agnostic user will get a payoff of μ. Hence, the expected payoff in this case is at-least $(1-\delta)\mu \ge 1/2 \ge (1-\delta)(1-\mu)$.

Case 2 ($\widehat{\mu} \in [0, 1/2-\tau]$): Similar to Case 1 here we use tail bound given by Hoeffding's inequality (Eq. 11) to get with probability at-least $(1-\delta)$,

$$\mu \le \widehat{\mu} + \sqrt{\frac{2\log(1/\delta)}{m}}$$
$$\implies \mu \le 1/2 - \frac{\delta}{2(1-\delta)} = \frac{1-2\delta}{2(1-\delta)}$$

Hence, $(1-\mu) \ge \frac{1}{2(1-\delta)}$. The agnostic user guarantees for the payoff of $(1-\mu)$ with probability at-least $(1-\delta)$ in this case. Hence we have the payoff of $(1-\delta)(1-\mu) \ge 1/2 \ge (1-\delta)\mu$ in this case.

Case 3 ($\widehat{\mu} \in (1/2-\tau, 1/2+\tau)$): Finally, in this case, the agnostic user chooses $h(z) = 1$ for all $z \in \mathcal{Z}$ with probability $1/2$ and $h(z) = -1$ for all $z \in \mathcal{Z}$ with probability $1/2$. Hence, the expected payoff is given by $\frac{1}{2}\mu + \frac{1}{2}(1-\mu) = 1/2$ irrespective of the true mean μ of positive samples. □

A.2 Runtime Lower Bound

As given in Lemma 4.9, the runtime of our algorithm is $m(\binom{n}{k})$. We argued in Section 4 that is made possible only since weights are sparse, and since the algorithm operates on a finite sample set of size m. If m is large, then this expression can be replaced with $\binom{q}{k}$. We now show that, in the limit (or under full information), the dependence on $\binom{q}{k}$ is necessary. The conclusion from Lemma A.4 is that to find the loss minimizer, any algorithm must traverse at least all such h; since there exist $\binom{q}{k}$ such functions, this is a lower bound. This is unsurprising; H_k is tightly related to the class of multilinear polynomials, whose degrees of freedom are exactly $\binom{q}{k}$.

Lemma A.4. *Consider a subclass of H_k composed of choice functions h which have $w(z) = a_+$ for exactly one z with $|z| = k$, and $w(z) = a_-$ otherwise. Then, for every such h, there exists a corresponding v, such that h is a unique minimizer (within this subclass) of the error w.r.t. v.*

Proof. Let z_1 and z_2 be distinct k size subsets, and let $a_- \in (0, 1)$ and $a_+ > \sum_{i\in[1,k]} \binom{n}{i}$. Further, let $\boldsymbol{w}_i$, $i \in [1, 2]$ be a weight function that assigns a_+ to z_i and a_- to all other subsets of size at most k. Let h_1 and h_2 be two function in H_k defined by the binary weighted functions $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ respectively. Observe that for $v_i = f_{h_i}$ the approximation error (see (Eq. (6))) of h_i is zero. Hence, to prove the lemma it is sufficient to show that $f_{h_1} \neq f_{h_2}$.

Suppose $f_{h_1} = f_{h_2}$. Since $z_1 \neq z_2$, there exists an $x \in \mathcal{X}$ such that $z_1 \subseteq x$ but $z_2 \subseteq x$. From Theorem 4.7 and the choice of a_+ and a_-, this implies $f_{h_1}(x) = 1$ but $f_{h_2}(x) = -1$, and hence, a contradiction. □

B ADDITIONAL RELATED WORK

Learning set functions. Concept learning refers to learning a binary function over hypercubes [3] through a query access model. Abboud et al. [1] provide a lower bound on *membership queries* to exactly learn a threshold function over sets where each element has small integer valued weights. Our learning framework admits large weights and has only a sample access in contrast with the query access studied in this literature. Feldman [10] show that the problem of learning set functions with sample access is computationally hard. However, we show (see Section 5) that the strategic setting is more nuanced; a more complex representations are disadvantageous for both user and the system. In other words, it is in the best interest of system to choose smaller (and much simpler) representations. A by-now classic work in learning theory studies the learnability from data of submodular (non-threshold) set functions [4]. Though we consider learning subadditive functions in this work, an extension to submodular valuations is a natural extension. Learning set functions is in general hard, even for certain subcalsses such as submodular functions. Rosenfeld et al. [17] show that it's possible to learn certain parameterized subclasses of submodular functions, when the goal is to use them for optimization. But this refers to learning over approximate proxy losses; whereas in our work, we show that learning is possible directly over the 0/1 loss.

Hierarchies of set functions. Conitzer et al. [6] (and independently, Chevaleyre et al. [5]) suggest a notion of k-wise dependent valuations, to which our Definition 4.1 is related. We also allow up to k-wise dependencies, but our valuations need not be positive and we focus on their sign (an indication whether an item is acceptable or not). Our set function valuations are also over item attributes rather than multiple items. Despite the differences, the definitions have a shared motivation: Conitzer et al. [6] believe that this type of valuation is likely to arise in many economic scenarios, especially since due to cognitive limitations, it might be difficult for a player to understand the inter-relationships between a large group of items. Hierarchies of valuations with limited dependencies/synergies have been further studied by Abraham et al. [2], Feige et al. [9] under the title 'hypergraph valuations'. These works focus on monotone valuations that have only positive weights for every subset, and are thus mathematically different than ours.

C MISSING PROOF FROM SECTION 2

Observation 2.1. *Every best-response $z \in \phi_h(x)$ induces the same value in the user's objective function (Eq. (3)).*

Proof. The proof follows from the definition of best response (Eq. 2). Let $z_1, z_2 \in \phi_h(x)$. Then since ϕ_h consists of only best response, we have either $h(z_1) = h(z_2) = 1$, or $h(z_1) = h(z_2) = -1$. Hence, $h(z_1) = \text{sign}(v(x))$ if and only if $h(z_2) = \text{sign}(v(x))$ for any $z_1, z_2 \in \phi_h(x)$. □

D MISSING PROOF FROM SECTION 3 AND AN ADDITIONAL EXAMPLE

D.1 Naïve users and Benevolent system

Lemma 3.1. *If system plays the* benevolent *strategy:*

$$\phi_h^{\text{benev}}(x) = \underset{z\subseteq x, |z|\in[k_1,k2]}{\text{argmax}} \{\mathbb{1}\{h(z) = \text{sign}(v(x))\},$$

then the naïve approach maximizes user payoff.

Proof. Since a naïve user plays $h(z) = \text{sign}(v(z))$, for each $x \in \mathcal{X}$ the payoff of the user is maximized if in response the system plays a $z \subseteq x$ such that $\text{sign}(v(z)) = \text{sign}(v(x))$. Observe that, if there exists a $z \in \mathcal{Z}$ and $z \subseteq x$, such that $\text{sign}(v(z)) = \text{sign}(v(x))$ then $z \in \phi_h^{\text{benev}}(x)$ and consequently the user's payoff is maximized for such an x. Conversely, if there exists no $z \in \mathcal{Z}$ and $z \subseteq x$, such that $\text{sign}(v(z)) = \text{sign}(v(x))$, then no truthful system can ensure more than zero utility for such a x. Hence, a benevolent system maximizes the utility of a naïve user. □

Now, we present an additional example to show how a naïve users choice function can be manipulated by the strategic system and as a consequence, user may obtain arbitrarily small payoff against a strategic system.

Example 2. Let $x_1 = \{a_1, a_2\}, x_2 = \{a_1, a_3\}, x_3 = \{a_1, a_4\}, x_4 = \{a_2, a_3\}, x_5 = \{a_3, a_4\}$ with $\text{sign}(v(x_1)) = \text{sign}(v(x_5)) = \text{sign}(v(a_2)) = \text{sign}(v(a_4)) = +1$ and $\text{sign}(v(x_2)) = \text{sign}(v(x_3)) = \text{sign}(v(x_4)) = \text{sign}(v(a_1)) = \text{sign}(v(a_3)) = -1$. Further, let $k_1 = k_2 = 1$ with $z_i = a_i$ as representations and a distribution $D = (\frac{\varepsilon}{4}, \frac{\varepsilon}{4}, 1-\varepsilon, \frac{\varepsilon}{4}, \frac{\varepsilon}{4})$ supported over $(x_1, x_2, x_3, x_4, x_5)$.

A unique truthful representation for this instance is $h = (-1, +1, -1, +1)$. A strategic agent can manipulate a naïve agent into non-preferred choices by using a representation $(a_2, a_1, a_4, a_2, a_4)$ for $(x_1, x_2, x_3, x_4, x_5)$. Note here that a naïve agent expected z_1 as a representation for x_3 since $h(z_1) = \text{sign}(v(x_3)) = -1$ and $h(z_4) = +1 \neq \text{sign}(v(x_3))$. However, a strategic agent chose a_4 as under given h we have $h(a_4) = 1$. A naïve users payoff in this case is reduced to ε which can be arbitrarily small.

E MISSING PROOFS FROM SECTION 4

Lemma 4.2. *For any $h : \mathcal{Z} \to \{\pm 1\}$, there exists $k \leq k_2$ and a corresponding k-order function h' such that:*

$$h(\phi_h(x)) = h'(\phi_{h'}(x)) .$$

Proof. Define k as follows: if $h(z) = -1$ for all $z \in \mathcal{Z}$ then $k = k_1$, and otherwise

$$k = \min_{k' \in [k_1, k_2]} \{\exists z \text{ such that } |z| = k' \text{ and } h(z) = 1, \text{ but for all } z' \subset z \text{ and } z' \in \mathcal{Z}, h(z') = -1\}.$$

Define h' as follows: For $|z| < k$, $h'(z) = -1$; for $|z| \geq k$

$$h'(z) = 1 \text{ if } \exists z' : |z'| = k, z' \subseteq z \text{ and } h(z') = 1;$$
$$h'(z) = -1 \text{ otherwise.}$$

First, we argue that h' defined as above satisfies $h'(\phi_{h'}(x)) = h(\phi_h(x))$ for all $x \in \mathcal{X}$. Suppose $h(\phi_h(x)) = 1$. Then there exists $z \in \mathcal{Z}$ such that $z \subseteq x$ and $h(z) = 1$. From the choice of k, we may assume without loss of generality that $|z| = k$. Further, from the construction of h', we have $h'(z) = 1$, and hence $h'(\phi_{h'}(x)) = h(\phi_h(x)) = 1$. Now suppose $h(\phi_h(x)) = -1$. Then for all $z \subseteq x$ we have $h(z) = -1$. In particular, for all $z \subseteq x$ such that $|z| = k$ we have $h(z) = -1$. This implies for all $z \subseteq x$ such that $|z| \geq k$ we have $h'(z) = -1$. This is because if there exists $z \subseteq x$ such that $|z| \geq k$ and $h'(z) = 1$ then from the definition of h' there exists a $z' \subseteq z \subseteq x$ such that $|z'| = k$, and $h(z') = 1$ (a contradiction). Additionally, from definition, for all $z \subseteq x$ such that $|z| < k$ we have $h'(z) = -1$. Hence, if $h(\phi_h(x)) = -1$ then $h'(\phi_{h'}(x)) = -1$.

Now, we show that h' is a k-order function. Let $a_- \in (-1, 0)$ and $w(z) = a_-$ for all z such that $|z| \leq k$ and $h'(z) = -1$. Further, for all z such that $|z| = k$, if $h'(z) = 1$ then let $w(z) = a_+ > \sum_{i \in [k]} \binom{n}{i}$. For all $z \in \mathcal{Z}$, if $|z| < k$ then by construction of h', we have $h'(z) = -1$, and since for all $z' \in \Gamma_k(z)$, $w(z') = a_- < 0$ we have $\sum_{z' \in \Gamma_k(z)} w(z') < 0$. Hence, for all $z \in \mathcal{Z}$, if $|z| < k$

$$h'(z) = \text{sign}\left(\sum\nolimits_{z' \in \Gamma_k(z)} w(z')\right) = -1.$$

Similarly, for all $z \in \mathcal{Z}$, if $|z| \geq k$ then by construction of h', we have $h'(z) = 1$ if and only if there exists a $z' \subseteq z$, and $z' = |k|$ such that $h'(z) = h(z) = 1$. In particular, if $|z| \geq k$ and $h'(z) = 1$ then there exists a $z' \subseteq z$, and $|z'| = k$ such that $w(z') = a_+$. Since $a_+ > \sum_{i \in [k]} \binom{n}{i}$, $a_- \in (-1, 0)$, and $k_2 \leq n$, we have if $|z| \geq k$ and $h'(z) = 1$ then

$$h'(z) = \text{sign}\left(\sum\nolimits_{z' \in \Gamma_k(z)} w(z')\right) = 1.$$

Finally, if $|z| \geq k$ and $h'(z) = -1$ then from the definition of h' there does not exists a $z' \subseteq z$, and $|z'| = k$ such that $w(z') = a_+$. Since $a_- \in (-1, 0)$, we have if $|z| \geq k$ and $h'(z) = -1$ then

$$h'(z) = \text{sign}\left(\sum\nolimits_{z' \in \Gamma_k(z)} w(z')\right) = -1.$$

□

Lemma 4.4. *For all k, $H_{k-1} \subseteq H_k$ and $H_k \setminus H_{k-1} \neq \emptyset$.*

Proof. Arbitrarily choose $u \subset E$ (recall E is the ground set) such that $|u| = k$, and let $w(u) = a_{k,+} > \sum_{i \in [k]} \binom{n}{i}$.[5] Also for all $z \neq u$ and $|z| \leq k$, let $w(z) = a_- \in (-1, 0)$. Let $h : \mathcal{Z} \to \{\pm 1\}$ be defined as follows

$$h(z) = \text{sign}\left(\sum\nolimits_{z' \in \Gamma_k(z)} w(z')\right)$$

From the definition of H_k, we have $h \in H_k$. We show that $h \notin H_{k-1}$. First, observe that for all $z \in \mathcal{Z}$ $h(z) = 1$ if and only if $u \subseteq z$. Suppose $h \in H_{k-1}$. Then there is a weight function w' on sets of size at most $k-1$ such that either $w'(z) = a_- \in (-1, 0)$ or $w_z = a_{k-1,+} > \sum_{i \in [k-1]} \binom{n}{i}$, and

$$h(z) = \text{sign}\left(\sum\nolimits_{z' \in \Gamma_{k-1}(z)} w'(z')\right)$$

[5] Here we wish to distinguish between a_+ for k and $k-1$ and hence we use $a_{k,+}$ instead of a_+.

Let $z \in \mathcal{Z}$ be such that $u \subseteq z$. This implies $h(z) = 1$. Hence there exist a $u' \subseteq z$ such that $|u'| = k-1$ and $w'(u') = a_{k-1,+}$. Let $\tilde{z} \in \mathcal{Z}$ be such that $u' \subseteq \tilde{z}$ but $u \not\subseteq \tilde{z}$. Such a $\tilde{z}$ exists because $u \cap u' \neq u$. Further, as $u \not\subseteq \tilde{z}$, we have $h(\tilde{z}) = -1$. But since $u' \subseteq \tilde{z}$, we have from the choice of $a_{k-1,+}$ and a_-

$$\sum\nolimits_{z' \in \Gamma_{k-1}(\tilde{z})} w'(z') \;>\; 0$$
$$\Rightarrow \operatorname{sign}\left(\sum\nolimits_{z' \in \Gamma_{k-1}(\tilde{z})} w'(z')\right) \;=\; h(\tilde{z}) = \; 1.$$

This gives a contradiction. Hence, $h \notin H_{k-1}$. □

Theorem 4.8. *For any $k \in [k_2]$, if Y is realizable then Alg returns an $\hat{h}$ that minimizes the empirical error.*

Proof. Throughout, for ease of notation, we use $x \in S$ to denote $x \in \{x_1, \ldots, x_m\}$. Let $Z_k = \{z : |z| = k, \exists x \in S \; z \subseteq x\}$. Recall $Z_{k,S}$ is equal to Z_k at the beginning of the algorithm. Also, for each $z \in Z_k$, let $\mathcal{X}_z = \{x \in S \mid z \subseteq x\}$. The following lemma characterizes the training set for which there exists an $h \in H_k$ with zero empirical error.

Lemma E.1. *There exists an $h \in H_k$ with zero empirical error if and only if for all $x \in S^+$ there exists a $z \in Z_k$ and $z \subseteq x$ such that $z \not\subseteq x'$ for all $x' \in S^-$.*

Proof. Suppose there exists an $h \in H_k$ with zero empirical error. Since $h \in H_k$, from Lemma 4.5 we have there exists a $g : Z_k \to \{\pm 1\}$ such that $h = \text{lift}(g)$. We state the following observation, and its proof follows from the definition of $\text{lift}(g)$.

Observation E.2. *(1) For every $z \in \mathcal{Z}$ such that $h(z) = 1$ there is a $z' \in Z_k$ such that $z' \subseteq z$ and $g(z') = 1$,*
(2) For every $z \in \mathcal{Z}$ such that $h(z) = -1$, it must be that for every $z' \in Z_k$, $z' \subseteq z$ we have $g(z') = -1$.

Further, as the empirical error for h is zero, we have the following observation.

Observation E.3. *(1) For every $x \in S^+$ there is a $z \in \mathcal{Z}$ and $z \subseteq x$ such that $h(z) = 1$,*
(2) For every $x \in S^-$, it must be that for every $z \in \mathcal{Z}$, $z \subseteq x$ we have $h(z) = -1$.

Proof. For every $x \in S^+$, since the empirical error is zero, we have $h(\phi_h(x)) = 1$. From the definition of ϕ_h, this implies there is a $z \in \mathcal{Z}$ and $z \subseteq x$ such that $h(z) = 1$. Similarly, for every $x \in S^-$, since the empirical error is zero, we have $h(\phi_h(x)) = -1$. Again from the definition of ϕ_h, it must be that for every $z \in \mathcal{Z}$, $z \subseteq x$ we have $h(z) = -1$. □

Hence, from Observations E.2 and E.3, we have for every $x \in S^+$ there is a $z \in Z_k$, $z \subseteq x$ such that $g(z) = 1$. Similarly, for every $x \in S^-$ it must be that for every $z \in Z_k$, $z \subseteq x$ we have $g(z) = -1$. Hence, for every $x \in S^+$ there exists a $z \in Z_k$ and $z \subseteq x$ such that $z \not\subseteq x'$ for all $x' \in S^-$.

Conversely, suppose for every $x \in S^+$ there exists a $z \in Z_k$ and $z \subseteq x$ such that $z \not\subseteq x'$ for all $x' \in S^-$. Then define $g : Z_k \to \{\pm 1\}$ as follows: a) for all $z \in Z_k$ such that $z \subseteq x$ for an $x \in S^-$, let $g(z) = -1$, b) for all $z \in Z_k$, such that $z \subseteq x$ for an $x \in S^+$ and $z \not\subseteq x'$ for an $x' \in S^-$, let $g(z) = 1$, c) for all $z \in Z_k$, such that $z \not\subseteq x$ for any $x \in S$, let $g(z) = -1$. From the supposition, we have that for every $x \in S^+$, there is a $z \in Z_k$ and $z \subseteq x$ such that $g(z) = 1$. Now define $h = \text{lift}(g)$. To show that the empirical error of h is zero, it is sufficient to show that for every $x \in S^-$ $h(\phi_h(x)) = -1$, and for every $x \in S^+$ $h(\phi_h(x)) = 1$. Let $x \in S^-$. From the definition of g, for every $z \in Z_k$ such that $z \subseteq x$, $g(z) = -1$. Hence, from the definition of lift, we have for every $z \in \mathcal{Z}$ such that $z \subseteq x$, $h(z) = -1$. Now from the definition of best response, we have $h(\phi_h(x)) = -1$. Similarly, if $x \in S^+$ from our supposition and the definition of g, we have there exists a $z \in Z_k$ such that $z \subseteq x$ and $g(z) = 1$. Hence, from the definition of lift, there exists a $z \in \mathcal{Z}$ such that $z \subseteq x$ and $h(z) = 1$. Finally, from the definition of best response , we have $h(\phi_h(x)) = 1$. □

Now, if $Y \in F_k$ then there exists an $h \in H_k$ such that the induced function of h is equal to Y, that is, $f_h = Y$. This implies there exists an $h \in H_k$ which attains zero empirical error on the training set. Since empirical error is always non-negative, such an h minimizes the empirical error in this case. Hence, from Lemma E.1, it follows that if Y is realizable then for all $x \in S^+$ at Step 15 of Alg there is either a $z \in Z^+$ and $z \subseteq x$, or $z \in Z_{k,S}$ and $z \subseteq x$. Now, observe that at the beginning of Step 20, set Z^+ satisfies the following:

$$z \in Z^+ \iff \exists\, x \in S^+ \text{ such that } z \subseteq x \text{ and } \nexists x' \in S^- \text{ such that } z \subseteq x'. \tag{13}$$

Further, at Step 20, for a $z \in Z_k$, $w(z) = a_+$ if $z \in Z^+$. This implies

$$w(z) = a_+ \iff \exists\, x \in S^+ \text{ such that } z \subseteq x \text{ and } \nexists x' \in S^- \text{ such that } z \subseteq x'. \tag{14}$$

Also, from Theorem 4.7, the induced function $f_{\hat{h}}$ corresponding to the returned $\hat{h}$ is given as

$$f_{\hat{h}}(x) = \operatorname{sign}\left(\sum\nolimits_{z \in \Gamma_k(x)} w(z)\right). \tag{15}$$

To complete the proof of theorem, we show that $f_{\hat{h}}(x_i) = y_i$ for every $x_i \in S$. Suppose $x \in S^-$. Then from Equations and 13 and 14, for every $z \subseteq x$ and $|z| \leq k$ we have $w(z) = a_- < 0$, and hence from Equation 15 for $f_{\hat{h}}$ we have $f_{\hat{h}}(x) = y = -1$. Similarly, suppose $x \in S^+$. Then from Equation 14, there exists $z \subseteq x$, $|z| = k$ such that $w(z) = a_+$. Hence, from Equation 15, and noting that $a_+ > \sum_{i \in [k]} \binom{n}{i}$ and $a_- \in (-1, 0)$ we have $f_{\hat{h}}(x) = y = 1$. □

F MISSING PROOFS FROM SECTION 5

Corollary 5.5. *If $\ell^* \leq k_2$ and the distribution D has full support on $\mathcal{X}$, then $k = \ell^*$ is the smallest k that gives zero approximation error.*

Proof. In the proof of Theorem 5.4, we show that for $k = \ell^*$, we have zero approximation error. Hence, to prove the corollary it is sufficient to show that for a $k < \ell^*$ the approximation error is not zero. Suppose there is an $h \in H_k$ such that $\varepsilon(h) = 0$ and $k < \ell^*$. Since the distribution D has full support, this implies $f_h(x) = v(x)$ for all $x \in \mathcal{X}$. Hence, the induced complexity of v is at most $k < \ell^*$ giving a contradiction. □

Corollary 5.6. *If $\ell^* > k2$, then the approximation error weakly increases with k, i.e., $\varepsilon(h_k^*) \leq \varepsilon(h_{k-1}^*)$ for all $k \leq k_2$. Furthermore, if the distribution D has full support on $\mathcal{X}$ then no k can achieve zero approximation error.*

Proof. The approximation error weakly decreases because $H_{k-1} \subseteq H_k$ for all $k \leq k_2$. Also, from the proof of Corollary 5.5, it is clear that no k can achieve zero approximation error. □

Lemma 5.7. *Let D be the uniform distribution over $\mathcal{X}$. Then there is a value function v for which $\varepsilon(h_k^*)$ diminishes convexly with k.*

Proof. We construct a v such that the approximation error for $h_k^* \in H_k$ is as given below

$$\varepsilon(h_k^*) = \frac{1}{4\binom{q}{n}} \sum_{\ell=k}^{k_2} \binom{k_2}{\ell}\binom{q-k_2}{n-\ell}.$$

It is easy to see that $\varepsilon(h_k^*)$ diminishes convexly with k (see Fig. 1). We choose k_2 elements $e_1, e_2, \ldots, e_{k_2} \in E$ (the ground set), and let z_e be the k_2 size subset consisting of these k_2 elements. For a $v : \mathcal{X} \to \mathbb{R}$, let $\mathcal{X}_v^+ = \{x \in \mathcal{X} | \operatorname{sign}(v(x)) = 1\}$ and $\mathcal{X}_v^- = \{x \in \mathcal{X} | \operatorname{sign}(v(x)) = -1\}$. We first show that there exists a v with the following two properties:

(1) if $x \in \mathcal{X}_v^+$ then there exists a $z \subseteq z_e$ such that $z \subseteq x$.
(2) For $k \in [1, k_2]$, let $\mathcal{X}_k = \{x \in \mathcal{X} | \exists z \subseteq z_e, |z| = k, \text{ and } z \subseteq x\}$. Then $\mathcal{X}_v^+ \cap \mathcal{X}_k = \frac{3}{4}(\sum_{\ell=k}^{k_2} \binom{k_2}{\ell}\binom{q-k_2}{n-\ell})$, for every $k \in [1, k_2]$.
(3) For every $z \subseteq z_e$, let $\mathcal{X}_z = \{x \in \mathcal{X} \mid z \subseteq x\}$. Then $|\mathcal{X}_v^+ \cap \mathcal{X}_z| = \frac{3}{4}\binom{q-k}{n-k}$, where $|z| = k$.

We construct such a v iteratively. We begin by making the following observation.

Observation F.1. *For each $k \in [1, k_2]$, $|\mathcal{X}_k| = \sum_{\ell=k}^{k_2} \binom{k_2}{\ell}\binom{q-k_2}{n-\ell}$.*

Proof. Recall $\mathcal{X}$ consists of size n subsets of E. For a $k \in [1, k_2]$ we wish to choose n size subsets of E that contain a $z \subseteq z_e$, $|z| = k$. This equivalent to choosing a fixed $\ell \geq k$ size subset of z_e and then choosing the remaining $n - \ell$ elements from the $q - k_2$ elements (not part of z_e) in E. For every $\ell \geq k$ we can choose ℓ size subset of z_e in $\binom{k_2}{\ell}$ ways, and for each such choice we can choose the remaining $n - \ell$ elements in $\binom{q-k_2}{n-\ell}$ ways. Since, this holds for any $\ell \in [k, k_2]$, we have $|\mathcal{X}_k| = \sum_{\ell=k}^{k_2} \binom{k_2}{\ell}\binom{q-k_2}{n-\ell}$. □

Constructing v: The idea is to iteratively add elements in $\mathcal{X}$ to $\mathcal{X}_v^+$, that is, iteratively determine the $x \in \mathcal{X}$ such that $\operatorname{sign}(v(x)) = 1$. In the first round, we arbitrarily choose $\frac{3}{4}\binom{q-k_2}{n-k_2}$ from $\mathcal{X}_{k_2}$ and add it to $\mathcal{X}_v^+$, and the remaining $\frac{1}{4}\binom{q-k_2}{n-k_2}$ are added to $\mathcal{X}_v^-$. At round k, assume we have constructed a v satisfying the above three properties for $k' > k$, that is,

(1) if $x \in \mathcal{X}_v^+$ then there exists a $z \subseteq z_e$ such that $z \subseteq x$.
(2) $\mathcal{X}_v^+ \cap \mathcal{X}_{k'} = \frac{3}{4}(\sum_{\ell=k'}^{k_2} \binom{k_2}{\ell}\binom{q-k_2}{n-\ell})$, for every $k' \in [k+1, k_2]$.
(3) For every $z \subseteq z_e$, let $\mathcal{X}_z = \{x \in \mathcal{X} \mid z \subseteq x\}$. Then $|\mathcal{X}_v^+ \cap \mathcal{X}_z| = \frac{3}{4}\binom{q-k'}{n-k'}$, where $|z| = k' > k$.

Hence, at round k, we have $\binom{k_2}{k}\binom{q-k_2}{n-k}$ elements in $\mathcal{X}_k$ which are not yet in $\mathcal{X}_v^+$ or $\mathcal{X}_v^-$. From these elements in $\mathcal{X}_k$, for every k size subset $z \subseteq z_e$ we arbitrarily choose $\frac{3}{4}\binom{q-k_2}{n-k}$ elements containing z and add the remaining $\frac{1}{4}\binom{q-k_2}{n-k}$ elements to $\mathcal{X}_v^-$. Now, observe that v satisfies the first two properties for every $k' \in [k, k_2]$ after this procedure. We argue v satisfies the third property for any $z \subseteq z_e$, such that $|z| = k$. The n size sets in $\mathcal{X}$ containing a $z \subseteq z_e$, such that $|z| = k$, can be partitioned into sets containing different $\ell >= k$ size subsets of z_e. In particular, we have the following combinatorial equality

$$\binom{q-k'}{n-k'} = \sum_{\ell=k'}^{k_2} \binom{k_2-k'}{\ell-k'}\binom{q-k_2}{n-\ell}$$

In the above expression, $\binom{q-k_2}{n-\ell}$ corresponds to the number of n size sets that contain only a a specific $\ell \geq k'$ size subset of z_e. Since our iterative procedure ensures from each such partition at least $\frac{3}{4}$ fraction of x is added to $\mathcal{X}_v^+$, we have that v satisfies the third property.

Optimal $h^* \in H_k$: From the construction of v, it is clear that the optimal $h^* \in H_k$ for the above constructed v, for any $k \in [1, k_2]$ satisfies the following: for every $z \in \mathcal{Z}$, $h^*(z) = 1$ if and only if there exists a $z' \subseteq z_e$, $|z'| = k$, and $z' \subseteq z$. Further as D is the uniform distribution, for such an h^*:

$$\varepsilon(h_k^*) = \frac{1}{4\binom{q}{n}} \sum_{\ell=k}^{k_2} \binom{k_2}{\ell}\binom{q-k_2}{n-\ell}.$$

□

Theorem 5.9. *For any k and m, given a sample set S of size m sampled from D and labeled by some v, we have*

$$\varepsilon(\hat{h}) - \varepsilon(h^*) \leq \sqrt{\frac{C(\binom{q}{k}\log(\binom{q}{k}/\epsilon) + \log(1/\delta)}{m}}$$

w.p. at least $1-\delta$ over S, and for a fixed constant C. In particular, Alg in Sec. 4.3, assuming Y is realizable, returns an $\hat{h} \in H_k$ for which:

$$\varepsilon(\hat{h}) \leq \sqrt{\frac{C(\binom{q}{k}\log(\binom{q}{k}/\epsilon) + \log(1/\delta)}{m}}$$

w.p. at least $1-\delta$ over S, and for a fixed constant C.

Proof. We first argue that the VC dimension of H_k is at most $\binom{q}{k}$. Let $d = \sum_{i\in[1,k]}\binom{n}{i}$, index the vectors in $\{0,1\}^d$ by $z \subseteq E$ (the ground set), such that $|z| \leq k$. Then each $z \in \mathcal{Z}$ can be represented by a binary vector $e_z \in \{0,1\}^d$, with the entry indexed by a z' being 1 if and only if $z' \subseteq z$. Further, let $w \in \{a_-, a_+\}^d$ be a binary weighted vector with a_- and a_+ as in Def. 4.3. Then from the definition of H_k, for each $h \in H_k$, there is a $w_h \in \{a_-, a_+\}^d$ such that a) $h(z) = \text{sign}(\langle w, e_z\rangle)$ for all $z \in Z$, and b) the entry of w indexed by a z' with $|z'| < k$ is a_-. From this we observe that the VC dimension of H_k is at most $\binom{q}{k}$, since each $h \in H_k$ is decided by the realization of binary weights on entries indexed by the $\binom{q}{k}$ sets. Now the first part of the theorem follows by noting that the first bound is the agnostic PAC generalization guarantee for an algorithm minimizing the empirical error in the standard classification setting with VC dimension at most $\binom{q}{k}$. To prove the second part, we have $Y \in F_k$, and hence the approximation error is zero, that is, $\varepsilon(h^*) = 0$ (from Lemma E.1). Further, Alg minimizes the empirical error (Theorem 4.8), and returns an $\hat{h}$ with zero empirical error. □

Lemma 5.10. *There exists a distribution D and a value function v such that for all $k < k' \leq k_2$, system has higher payoff against the optimal $h_k^* \in H_k$ than against $h_{k'}^* \in H_{k'}$.*

Proof. The v is constructed as in the proof Lemma 5.7. We recall notations from the proof of Lemma 5.7: z_e is a k_2 size subset. Further, in the proof of Lemma 5.7 we argued that for $k \in [1, k_2]$, h_k^* is such that for all $z \in \mathcal{Z}$, $h^*(z) = 1$ if and only if there exists a k size $z' \subseteq z$ which is also a subset of z_e.

Now let $k, k' \in [1, k_2]$ such that $k < k'$. Since D is the uniform distribution, to show system's utility is more for k compared to k' it is sufficient to show that

$$\sum_{x\in\mathcal{X}} \mathbb{1}\{h_k^*(\phi_{h_k^*}(x)) = 1\} > \sum_{x\in\mathcal{X}} \mathbb{1}\{h_{k'}^*(\phi_{h_{k'}^*}(x)) = 1\}$$

From the proof of Lemma 5.7 and Theorem 4.7, it follows that

$$\sum_{x\in\mathcal{X}} \mathbb{1}\{h_k^*(\phi_{h_k^*}(x)) = 1\} = \sum_{x\in\mathcal{X}} \mathbb{1}\{f_{h_k^*}(x) = 1\} = \sum_{\ell=k}^{k_2}\binom{k_2}{\ell}\binom{q-k_2}{n-\ell}$$

Similarly,

$$\sum_{x\in\mathcal{X}} \mathbb{1}\{h_{k'}^*(\phi_{h_{k'}^*}(x)) = 1\} = \sum_{\ell=k'}^{k_2}\binom{k_2}{\ell}\binom{q-k_2}{n-\ell}$$

Since $k < k'$, we have

$$\sum_{x\in\mathcal{X}} \mathbb{1}\{f_{h_k^*}(x) = 1\} = \sum_{\ell=k}^{k_2}\binom{k_2}{\ell}\binom{q-k_2}{n-\ell} > \sum_{\ell=k'}^{k_2}\binom{k_2}{\ell}\binom{q-k_2}{n-\ell}$$

implying system's utility is more for k compared to k'. □

Corollary 5.11. *For the system, lower k_2 is better (in a worst-case sense).*

Proof. In Lemma 5.10, we showed there exists a user with v such that for all $k, k' \in [1, k_2]$ and $k < k'$, the system has better utility against the optimal choice function in H_k than in $H_{k'}$. Since the choice of k the user can make is bounded by k_2, a lower k_2 maximizes the worst-case payoff to the system. □