Refinement as a Service: Algorithmic Predictor Refinement

Wei Tang
Chinese University of Hong Kong
weitang@cuhk.edu.hk

Hanrui Zhang
Chinese University of Hong Kong
hanrui@cse.cuhk.edu.hk

Abstract

Prediction aggregation aims to combine information from multiple predictors into a more informative one. We study this question in the setting of calibrated predictors, where each prediction must equal the conditional expectation of the quantity being predicted given the predictor's signal. Given several calibrated input predictors and the feature distribution, but not the underlying Bayes probabilities, we ask when one can construct refined calibrated predictors that preserve the information in the original predictors and cannot be further refined using the available information.

We formulate calibrated predictors as signaling schemes and define refinement through feature-independent garblings: a predictor refines another if its signal can simulate the other's signal. Constructibility is characterized through observable linear information: each signal corresponds to a vector over the feature space, and a new signal is constructible exactly when its vector lies in the linear span of the input signal vectors. Under this formulation, we establish a sharp algorithmic picture. For deterministic output predictors, bilateral refinement admits a polynomial-time algorithm based on a bipartite graph between the two input signal partitions, while refinement with an arbitrary number of input predictors is NP-hard. In contrast, when randomized output predictors are allowed, we give a polynomial-time algorithm for any number of input predictors by decomposing constructible signal vectors into extreme rays of the associated polyhedral cone.

1 Introduction

Prediction aggregation (or forecast aggregation) is the process of combining multiple predictions into a single prediction in a principled way (see, e.g., Stone, 1961; Bates and Granger, 1969; Gneiting et al., 2007; Ranjan and Gneiting, 2010). The idea is that the aggregate prediction contains information inherent in all input predictions, and is therefore more "informative" than any single input prediction. This very idea, instantiated in different ways, lies behind fundamental methodologies and tools in different branches of computer science / machine learning: In supervised learning, boosting techniques are used to turn weak learners into strong ones (Schapire, 1990; Freund, 1995; Friedman et al., 2000); in online learning, multiplicative weights algorithms achieve low regret against the best fixed expert (Littlestone and Warmuth, 1994; Cesa-Bianchi and Lugosi, 2006); in crowdsourcing and computational social choice, voting and aggregation methods combine multiple individuals' opinions into a single, supposedly representative opinion, based on which social decisions can be made (Raykar et al., 2010; Brandt et al., 2016); even in LLM architectures, mixture of experts (MoE) models show drastically improved efficiency and scalability while achieving similar, or even better performance compared to dense models (Jacobs et al., 1991; Dai et al., 2024; Liu et al., 2024). All this highlights the power of the clean idea underlying prediction aggregation, which motivates our study in this paper.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

At the same time, the above also hints at the *immensity* of this very idea: It is deeply rooted in so many different areas, where it manifests in so many different technical forms, to the extent that any serious technical discussion of the idea must refrain from talking about the idea itself — instead, one must begin by putting the idea into a specific, technical context. The context of our investigation is *refinement of calibrated predictors*. Below we explain the setup and the keywords ("predictors", "calibrated", and "refinement") using a realistic example.

Example: click-through rate (CTR) prediction in online advertising. A user visits a platform (e.g., Google or Meta), which triggers the display of an ad. The platform must decide which ad to display from a set of candidates, and the decision depends on the predicted click-through rates (CTRs) of these ads, which are the predicted probabilities that the user would click on each of these ads. The platform sets up a *predictor* for each candidate ad, which maps the user's features (region, language, OS, historical activities on the same platform, etc.) to a numerical prediction. Typically, this predictor is a machine learning model trained on historical data. Needless to say, the use of such predictors is omnipresent in all kinds of applications.

In online advertising, we want the predictors to be *calibrated*, meaning that conditioning on a particular prediction made by a predictor, the probability that the user would click on the ad is precisely the predicted value (Dawid, 1982; Foster and Vohra, 1997). In other words, the prediction is always equal to the conditional expectation of the true CTR given the prediction.[1] This ensures that the ads are properly ranked, and the payments that the advertisers are charged are correctly computed (McMahan et al., 2013; He et al., 2014; Google Ads Help). Calibration is also desired in many other scenarios, e.g., medical diagnosis and weather forecasting.

The need for *refinement* of predictors arises when different platforms hold complementary information about the same prediction task. To make this concrete, suppose there are three equally likely user types, A, B, and C. Two platforms hold calibrated CTR predictors f_1 and f_2 for the same ad,[2] where f_1 pools Types A and B at prediction 0.3 and separates Type C at prediction 0.7, while f_2 separates Type A at prediction 0.1 and pools Types B and C at prediction 0.6. By calibration, these predictions identify the average CTRs of the corresponding groups. In particular, since Types A and B have average CTR 0.3, while Type A alone has average CTR 0.1, Type B must have average CTR $0.3 \times 2 - 0.1 = 0.5$. Thus, combining the two predictors yields the fully refined calibrated predictor with predictions $0.1, 0.5, 0.7$ for Types A, B, and C, respectively. This refinement preserves the information in both input predictors and makes strictly finer predictions, thereby giving a *Pareto improvement*. Refinement, however, is not always possible. For example, with four equally likely types A, B, C, and D, suppose f_1 pools $\{\mathrm{A}, \mathrm{B}\}$ at 0.3 and $\{\mathrm{C}, \mathrm{D}\}$ at 0.7, while f_2 pools $\{\mathrm{A}, \mathrm{C}\}$ at 0.4 and $\{\mathrm{B}, \mathrm{D}\}$ at 0.6. Although the two predictors contain different information, neither can be further refined using only the information contained in the other. This leads us to the main questions addressed in the paper:

> Given a collection of input calibrated predictors, when is refinement possible using the information contained in the other predictors? More generally, how can we compute maximally refined predictors, so that no further refinement is possible given all available information?

1.1 Results and Techniques

Predictors as signaling schemes and constructible refinements. To facilitate our study, we identify each calibrated predictor with the signaling scheme it induces.[3] Once a signal is fixed, its numerical prediction is pinned down by calibration: it must equal the conditional expectation of the label given that signal. This viewpoint is consistent with the standard informational interpretation in forecast aggregation literature, where reported forecasts are typically treated as posterior beliefs under the forecasters' information. Our focus is different: we ask *when* and *how* the information contained in several calibrated predictors can be combined into a constructible refinement (see, e.g., Stone, 1961; DeGroot and Fienberg, 1983; Clemen, 1989; Arieli et al., 2018; Guo et al., 2025 and Section B for more discussions). Under this perspective, the numerical values carry no additional primitive information beyond the signal structure itself. In the deterministic case, a pre-

[1] Calibration does not require feature-level accuracy; for example, a predictor that always outputs the population-average CTR is calibrated.

[2] Note that predictors for different ads generally should not be combined, since they target different quantities.

[3] Unless stated otherwise, all predictors discussed in this work are calibrated.

dictor partitions the feature space into signal cells; in the randomized case, it maps each feature to a distribution over signals. Under this representation, refinement has a natural information-theoretic meaning. Given two predictors f and g, we say that g refines f if the signal generated by f can be simulated from the signal generated by g through a feature-independent post-processing rule. Intuitively, g contains at least as much information as f, because one can always recover f's signal from g's signal by garbling. Our goal is therefore to find, for each input predictor f_i, a constructible predictor g_i that refines f_i and cannot be further refined within the constructible class.

It remains to clarify constructibility. Semantically, a predictor is constructible if the calibrated value associated with each of its signals can be uniquely determined from the input predictors, without knowing the underlying Bayes probabilities. To make this criterion operable, we represent each signal by a vector over the feature space: for deterministic predictors, these vectors are 0-1 indicators of signal cells; for randomized predictors, they record the probabilities with which each feature generates the signal. The key observation is linear-algebraic: a new signal is constructible precisely when its vector can be expressed as a linear combination of the signal vectors of the input predictors. We formalize this characterization in Section 3; a worked example is deferred to Section A. In Section G, we also provide a more in-depth comparison of our (signal-level) notion of calibration and its prediction-level counterpart.

The above representation gives an operable way to check whether candidate predictors are constructible and whether they refine the inputs. The main algorithmic challenge is to find constructible predictors that maximally refine the input predictors. We study this problem separately for deterministic and randomized outputs.

Refinement into deterministic predictors. We first consider the restriction in which the output predictors must be deterministic. This is natural when randomized predictions are infeasible, e.g., when one needs a stable prediction for each feature. Our results establish a dichotomy between bilateral and multilateral refinement:

Theorem 1.1 (informal)**.** *Restricted to deterministic predictors:*

- *When there are only* 2 *input predictors, there is a polynomial-time algorithm for predictor refinement.*
- *With an arbitrary number* k *of input predictors, predictor refinement is* NP*-hard.*

Technically, the bilateral case has a clean graph-theoretic structure. Given two deterministic input predictors, we build a bipartite overlap graph whose vertices are the signal cells of the two predictors, and where an edge indicates that two cells intersect on at least one feature. Refinement is governed by the connected components that arise after removing a cell from this graph: these components determine how that cell can be split into constructible branch pieces. Iterating this decomposition yields the finest constructible deterministic refinement of each input predictor in polynomial time.

The hardness result for multilateral refinement shows that this graph structure is special to the bilateral case. With an arbitrary number of input predictors, deciding whether a target cell admits a nontrivial constructible split becomes a higher-dimensional linear-algebraic problem. We show that this problem can encode exact covering: one must combine the input signal vectors so that irrelevant coordinates cancel out exactly while the remaining support lies inside the target cell. This leads to NP-hardness via a reduction from a variant of exact cover.

Refinement into randomized predictors. We then investigate the unrestricted problem, where output predictors are allowed to be randomized. In this regime, we obtain a clean positive result:

Theorem 1.2 (informal)**.** *When randomization is allowed, there is a polynomial-time algorithm for predictor refinement with any number of input predictors.*

Technically, randomization turns the problem into a geometric decomposition problem over nonnegative signal vectors. The constructible signals form a polyhedral cone, and a predictor is maximally refined once each of its signal vectors lies on an extreme ray of this cone. Our algorithm decomposes every input signal vector into such extreme-ray pieces. The decomposition can be found in polynomial time. Starting from a target signal vector, the algorithm repeatedly uses linear feasibility to search for a nonzero constructible vector with strictly smaller support. If no such vector exists, the current vector spans an extreme ray; if one exists, it yields a valid split. Each split strictly decreases a support-size potential, which bounds the total number of iterations polynomially.

Application: refinement as a service. Our algorithmic results can be used directly when participants share their predictors and run one of our refinement algorithms. They also suggest a service-oriented interpretation, motivated by the growing use of machine-learning services and prediction APIs (Tramèr et al., 2016; Bommasani et al., 2021; Google Cloud). A platform may hold a highly optimized general-purpose predictor, while an agent may hold a specialized predictor for a particular domain. The platform can run our algorithm to incorporate all constructible information from its predictor into the agent's specialized predictor, producing a refined predictor that is more informative while preserving the agent's domain-specific expertise. Conversely, upon agreement, the platform can refine its own predictor using the agent's information, or add the agent's predictor to its available bag of signal vectors. In this sense, refinement can be viewed as a service for improving calibrated predictors without requiring access to the underlying Bayes probabilities.

2 Preliminary

We begin by introducing the basic notation for the *predictor refinement* problem in the binary classification setting. Let $\mathcal{X}$ be a finite feature space, and let $\mathcal{Y} = \{0, 1\}$ be the binary label space. The data are generated according to a joint distribution $\mathcal{D} \in \Delta(\mathcal{X} \times \mathcal{Y})$.

A possibly randomized predictor $f = (\Sigma_f, T_f, p_f)$ consists of a signal space Σ_f, a stochastic kernel $T_f : \mathcal{X} \to \Delta(\Sigma_f)$, and a prediction map $p_f : \Sigma_f \to [0, 1]$. The kernel $T_f(\cdot \mid x)$ specifies, for each feature x, a distribution over signals, with $T_f(s \mid x)$ denoting the conditional probability of realizing signal s given the feature realization $X = x$; the prediction map then assigns each realized signal $s \in \Sigma_f$ a numerical prediction $p_f(s) \in [0, 1]$. Given a feature realization $X = x$, the predictor first draws a signal $s \sim T_f(\cdot \mid x)$ and then assigns a prediction $p_f(s)$ to this signal. Together with the data distribution $\mathcal{D}$, the predictor f induces a joint distribution over predictions and labels, denoted by $\Gamma_f \in \Delta([0, 1] \times \mathcal{Y})$. Let F_f denote the CDF of the induced marginal distribution of the predictions p. Throughout this work, we focus on perfectly calibrated predictors. Formally, a predictor f is perfectly calibrated if the following condition holds:

$$\mathbb{E}_{(p,Y)\sim\Gamma_f}[Y \mid p] = p, \quad \text{for every } p \in \mathsf{supp}(F_f) \,.$$

As we can see, under perfect calibration w.r.t. the data distribution $\mathcal{D}$, the prediction assigned to each signal is uniquely determined once the signal space Σ_f and kernel T_f are fixed. Consequently, we often omit the prediction map and write $f = (\Sigma_f, T_f)$. A predictor f is deterministic if $T_f(\cdot \mid x)$ is a point mass for every feature $x \in \mathcal{X}$.

We formulate predictors as signal schemes, rather than simply as maps from features to numerical predictions, because calibrated numerical values alone may collapse signals that are informationally distinct. Since refinement is meant to compare the information carried by predictors, we keep realized signals as the primitive objects and recover their numerical values only through calibration. For example, suppose two features x_1 and x_2 both have calibrated value $1/2$. Consider one predictor that uses two distinct signals, one for x_1 and one for x_2, and another predictor that pools $\{x_1, x_2\}$ into a single signal. Both predictors always output the same numerical prediction, $1/2$. However, they are not informationally equivalent: the first predictor can simulate the second by forgetting which signal was realized, while the second cannot recover the distinction between x_1 and x_2. Thus, representing predictors only by their numerical predictions would identify these two predictors, whereas the signal-based representation correctly distinguishes their refinement content.

Informational dominance. With the above signal-based representation in place, we now introduce the information-theoretic dominance relation used throughout the paper to compare predictors, which we call informational dominance.

Definition 2.1 (Informational dominance)**.** *Let $f = (\Sigma_f, T_f), g = (\Sigma_g, T_g)$ be two (possibly) randomized predictors. We say that f informationally dominates g, denoted by $f \succ g$, if and only if the following two conditions hold:*

- *there exists a stochastic kernel $H : \Sigma_f \to \Delta(\Sigma_g)$ such that, for every feature $x \in \mathcal{X}$, we have $T_g(\cdot \mid x) = \sum_{s\in\Sigma_f} H(\cdot \mid s)\, T_f(s \mid x)$.*
- *there does not exist any stochastic kernel $H : \Sigma_g \to \Delta(\Sigma_f)$ such that, for every feature $x \in \mathcal{X}$, we have $T_f(\cdot \mid x) = \sum_{s\in\Sigma_g} H(\cdot \mid s)\, T_g(s \mid x)$.*

Equivalently, conditional on $X = x$, one may first draw the hidden signal s_f from $T_f(\cdot \mid x)$, and then pass s_f through the feature-independent noisy channel H to obtain a new signal s_g; the resulting conditional law of s_g is exactly $T_g(\cdot \mid x)$.

The predictor-refinement problem. Fix the feature marginal distribution $\lambda \in \Delta(\mathcal{X})$ induced by the data distribution $\mathcal{D}$, and fix k predictors $(f_j)_{j\in[k]}$. A predictor g is said to be constructible from λ and $(f_j)_{j\in[k]}$ if it can be determined solely from this information; that is, if there exists a mapping $\mathcal{A}$ such that $g = \mathcal{A}(\lambda, (f_j)_{j\in[k]})$. Equivalently, g can be computed using only the information contained in λ and $(f_j)_{j\in[k]}$, without access to the unknown Bayes probabilities $\{\mathbb{P}[Y = 1 \mid x]\}_{x\in\mathcal{X}}$. We write $\mathcal{G}$ for the set of all such constructible predictors. When there is no ambiguity, we suppress the dependence of $\mathcal{G}$ on λ and $(f_j)_{j\in[k]}$ in the notation. We next define the subset of constructible predictors that are undominated under our informational dominance.

Definition 2.2 (Undominated predictors)**.** *Fix a feature distribution λ and fix k predictors $(f_j)_{j\in[k]}$. Let $\mathcal{G}_\succ \subseteq \mathcal{G}$ be the set of undominated constructible predictors: $\mathcal{G}_\succ \triangleq \{g \in \mathcal{G} :$ there does not exist $g^\dagger \in \mathcal{G}$ such that $g^\dagger \succ g\}$.*

The set $\mathcal{G}_\succ$ includes all constructible predictors that are not dominated, within the class $\mathcal{G}$. We also denote by $\mathcal{G}^{\text{DET}} \subset \mathcal{G}$ the set of all constructible deterministic predictors, and let $\mathcal{G}^{\text{DET}}_\succ \triangleq \{g \in \mathcal{G}^{\text{DET}} :$ there does not exist $g^\dagger \in \mathcal{G}^{\text{DET}}$ such that $g^\dagger \succ g\}$ be the set of all deterministic informationally-undominated predictors. We are now ready to formalize the predictor refinement problem.

PREDICTORREFINE:
Input: a feature distribution $\lambda = (\lambda_i)_{i\in[n]}$ and k predictors $(f_j)_{j\in[k]}$.
Question: output a collection of predictors $(g_j)_{j\in[k]}$ such that for every $j \in [k]$,
- if there exists a predictor $g \in \mathcal{G}_\succ$ such that $g \succ f_j$, then $g_j \in \mathcal{G}_\succ$ and $g_j \succ f_j$;
- otherwise $g_j = f_j$.

Equivalently, the problem PREDICTORREFINE asks, for each initial predictor f_i, to return an undominated strict improvement whenever such an improvement exists, and to return the original predictor f_i unchanged otherwise. Importantly, the aggregator observes only the feature distribution λ and the initial predictors $(f_j)_{j\in[k]}$, and does not observe the underlying Bayes probabilities.

Additional notations. We write the feature space as $\mathcal{X} = \{x_i\}_{i\in[n]}$, and let $\lambda = (\lambda_i)_{i\in[n]}$ where $\mathbb{P}[X = x_i] = \lambda_i$ for $i \in [n]$ is the probability of feature $X = x_i$. For each feature x_i, we let $q_i \triangleq \mathbb{P}[Y = 1 \mid X = x_i] \in [0, 1]$ denote its Bayes probability for the label $Y = 1$. For nonzero vector $\mathbf{v} \in \mathbb{R}^n_+$, we slightly abuse notation and define $\lambda(\mathbf{v}) \triangleq \sum_{i\in[n]} \lambda_i \mathbf{v}_i, Y(\mathbf{v}) \triangleq \sum_{i\in[n]} \lambda_i q_i \mathbf{v}_i, \mu(\mathbf{v}) \triangleq \frac{Y(\mathbf{v})}{\lambda(\mathbf{v})}$. For nonempty $S \subseteq [n]$, we also write $\mathbf{1}_S$ as the set-indicator vector.

3 Observable Linear Information

In this section, we recast the signal-kernel representation for the predictors from Section 2 into an equivalent linear form (see Section 3.1), which gives rise to a formal definition of constructible predictors from observable information $(\lambda, (f_j)_{j\in[k]})$ (see Section 3.2).

3.1 Representing Predictors via Bag-of-Vectors

In this subsection, we reformulate the predictors and the informational dominance via the bag-of-vectors. Throughout this section, by a *bag* we mean a finite multiset.

Definition 3.1 (Raw bag-of-vectors representation)**.** *Let $f = (\Sigma_f, T_f)$ be a finitely supported predictor. Relabel the signal space as $\Sigma_f = [m_f]$ where $m_f \triangleq |\Sigma_f|$. For each $\ell \in [m_f]$, define the signal vector $\mathbf{v}_{f,\ell} \in \mathbb{R}^n_+$ by $\mathbf{v}_{f,\ell}(i) \triangleq T_f(\ell \mid x_i)$ for $i \in [n]$. The multiset $\mathcal{V}^{\text{raw}}(f) \triangleq \{\mathbf{v}_{f,\ell} : \ell \in [m_f]\}$ is called the* raw bag-of-vectors *representation of predictor f.*

Since the signal kernel $T_f(\cdot \mid x_i)$ is a probability distribution on $[m_f]$ for each $i \in [n]$, the vectors in $\mathcal{V}^{\text{raw}}(f)$ satisfy $\mathbf{v}_{f,\ell} \in \mathbb{R}^n_+$ for all $\ell \in [m_f]$ and $\sum_{\ell\in[m_f]} \mathbf{v}_{f,\ell} = \mathbf{1}$. Conversely, any finite

bag $\{\mathbf{v}_1, \ldots, \mathbf{v}_m\} \subseteq \mathbb{R}^n_+$ satisfying $\sum_{\ell \in [m]} \mathbf{v}_\ell = \mathbf{1}$ defines a finitely supported signal kernel: one simply takes signal space $[m]$ and sets $T(\ell \mid x_i) \triangleq \mathbf{v}_\ell(i)$ for $i \in [n]$, $\ell \in [m]$. The coordinatewise nonnegativity and summation condition ensure that $T(\cdot \mid x_i)$ is a probability distribution on $[m]$ for every $i \in [n]$. Thus, up to relabeling of signals, finitely supported signal kernels are in one-to-one correspondence with finite bags of nonnegative vectors whose coordinatewise sum is $\mathbf{1}$.

The raw bag-of-vectors representation contains a redundancy: two nonzero vectors on the same positive ray encode the same informative signal, up to an independent random split. Accordingly, we normalize out this redundancy.

Definition 3.2 (Ray equivalence and reduced bag)**.** *For nonzero vectors $\mathbf{v}, \mathbf{v}^\dagger \in \mathbb{R}^n_+$, we write $\mathbf{v} \sim_{\mathrm{ray}} \mathbf{v}^\dagger$ iff $\mathbf{v}^\dagger = c\,\mathbf{v}$ for some $c > 0$. Let $\mathcal{V}^{\mathrm{raw}}(f) = \{\mathbf{v}_{f,\ell}\}_{\ell \in [m_f]}$. Let $\mathcal{R}_1, \ldots, \mathcal{R}_L$ be the distinct $\sim_{\mathrm{ray}}$-equivalence classes. For each $t \in [L]$, define the merged vector $\bar{\mathbf{v}}_{f,t} \triangleq \sum_{\mathbf{v}_{f,\ell} \in \mathcal{R}_t} \mathbf{v}_{f,\ell}$. The reduced bag-of-vectors representation of f is $\mathcal{V}(f) \triangleq \{\bar{\mathbf{v}}_{f,t} : t \in [L]\}$.*

If $\mathbf{v}^\dagger = c\,\mathbf{v}$ for some $c > 0$, then $\mu(\mathbf{v}^\dagger) = \mu(\mathbf{v})$. Thus, all vectors in one ray class carry the same numeric prediction. Moreover, passing from $\mathcal{V}^{\mathrm{raw}}(f)$ to $\mathcal{V}(f)$ only merges signals that differ by an independent random split, so this canonicalization does not change the informational-dominance content of the predictor. From now on, whenever we write $\mathcal{V}(f)$, we always mean the reduced bag-of-vectors representation.

Informational dominance in bag-of-vectors form. With the above reformulation, we can also define informational dominance in bag-of-vectors form.

Definition 3.3 (Informational dominance in bag form)**.** *Let $\mathcal{V}(f) = \{\mathbf{v}_{f,\ell}\}_{\ell \in [m_f]}$ and $\mathcal{V}(g) = \{\mathbf{v}_{g,r}\}_{r \in [m_g]}$ be the reduced bags of two predictors. We say that f informationally dominates g, denoted by $f \succ g$, if and only if the following two conditions hold:*

- *there exists a row-stochastic matrix $\mathbf{H} = (\mathbf{H}_{\ell,r})_{\ell \in [m_f],\, r \in [m_g]} \in [0,1]^{m_f \times m_g}$ such that, for every $r \in [m_g]$, we have $\mathbf{v}_{g,r} = \sum_{\ell \in [m_f]} \mathbf{H}_{\ell,r}\, \mathbf{v}_{f,\ell}$.*
- *there does not exist any row-stochastic matrix $\mathbf{H} = (\mathbf{H}_{r,\ell})_{r \in [m_g],\, \ell \in [m_f]} \in [0,1]^{m_g \times m_f}$ such that, for every $\ell \in [m_f]$, we have $\mathbf{v}_{f,\ell} = \sum_{r \in [m_g]} \mathbf{H}_{r,\ell}\, \mathbf{v}_{g,r}$.*

Equivalently, if $\mathbf{V}_f \in \mathbb{R}^{m_f \times n}_+$ and $\mathbf{V}_g \in \mathbb{R}^{m_g \times n}_+$ denote the matrices whose rows are the vectors in $\mathcal{V}(f)$ and $\mathcal{V}(g)$, respectively, then the first condition above is equivalent to $\mathbf{V}_g = \mathbf{H}^\top \mathbf{V}_f$.

Lemma 3.1. *Definition 3.3 is equivalent to the signal-kernel definition of informational dominance in Definition 2.1.*

3.2 Constructible Predictors from Observable Linear Information

In this section, we define the set of constructible predictors. Fix the input predictors $(f_j)_{j \in [k]}$, and let $(\mathcal{V}(f_j))_{j \in [k]}$ where $\mathcal{V}(f_j) = \{\mathbf{v}_{j,\ell}\}_{\ell \in [m_j]}$ denote their reduced bags where $m_j \triangleq |\Sigma_{f_j}|$. Define $N \triangleq \sum_{j \in [k]} m_j$, and define the unknown vector $\mathbf{y} \triangleq (\lambda_1 q_1, \ldots, \lambda_n q_n)^\top \in \mathbb{R}^n$.

Definition 3.4 (Signal-feature matrix)**.** *Define the signal-feature matrix $\mathbf{M} \in \mathbb{R}^{N \times n}_+$ whose rows are the signal vectors of the input predictors. Equivalently, if row (j,ℓ) corresponds to the vector $\mathbf{v}_{j,\ell}$, then $\mathbf{M}_{(j,\ell),i} \triangleq \mathbf{v}_{j,\ell}(i)$ for all $j \in [k]$, $\ell \in [m_j]$, $i \in [n]$. Let the vector $\mathbf{b} \triangleq \mathbf{M}\mathbf{y} \in \mathbb{R}^N$ where its (j,ℓ)-th component is $\mathbf{b}_{(j,\ell)} = Y(\mathbf{v}_{j,\ell}) = \mu(\mathbf{v}_{j,\ell})\, \lambda(\mathbf{v}_{j,\ell})$.[4] Thus, the aggregator's information about the unknown vector $\mathbf{y}$ is exactly the linear observation $\mathbf{M}\mathbf{y} = \mathbf{b}$.*

For the fixed inputs $(\lambda, (f_j)_{j \in [k]})$, the signal-wise label totals $Y(\mathbf{v}_{j,\ell})$ are exactly the linear functionals of the unknown vector $\mathbf{y} = (\lambda_1 q_1, \ldots, \lambda_n q_n)^\top$ that are directly determined by the observed predictors. Collecting these quantities over all input signals yields a linear system $\mathbf{M}\mathbf{y} = \mathbf{b}$; we call the pair $(\mathbf{M}, \mathbf{b})$, or equivalently the equation $\mathbf{M}\mathbf{y} = \mathbf{b}$, the *observable linear information* induced by $(\lambda, (f_j)_{j \in [k]})$.

[4] This scalar is determined by the observed inputs $(\lambda, (f_j)_{j \in [k]})$ since $\mu(\mathbf{v}_{j,\ell})$ is the prediction attached to the signal $\mathbf{v}_{j,\ell}$ and $\lambda(\mathbf{v}_{j,\ell})$ is computable from λ and $\mathbf{v}_{j,\ell}$.

If predictor f_j is deterministic, then for each feature x_i there exists a unique signal in Σ_{f_j} that is realized with probability one. Define the cell

$$A_{j,\ell} \triangleq \{\, i \in [n] : T_{f_j}(\ell \mid x_i) = 1 \,\}, \quad \ell \in [m_j] \,.$$

Because f_j is deterministic, the nonempty sets $A_{j,1}, \ldots, A_{j,m_j}$ form a partition of $[n]$. We call this partition the *signal partition* of f_j and denote it by $\Pi_{f_j} = \{A_{j,\ell}\}_{\ell \in [m_j]}$.

Remark 3.5. *If predictor f_j is deterministic with signal partition $\Pi_{f_j} = \{A_{j,\ell}\}_{\ell \in [m_j]}$, then $\mathbf{v}_{j,\ell} = \mathbf{1}_{A_{j,\ell}}$ for every $\ell \in [m_j]$. Thus, in the deterministic special case, Definition 3.4 reduces to a matrix with $0/1$ rows.*

We identify the row space of $\mathbf{M}$ with a subspace of $\mathbb{R}^n$:

$$\textsc{Row}(\mathbf{M}) \triangleq \{\mathbf{M}^\top \alpha : \alpha \in \mathbb{R}^N\} = \textsc{span}\{\mathbf{v}_{j,\ell} : j \in [k],\ \ell \in [m_j]\} \subseteq \mathbb{R}^n.$$

Definition 3.6 (Observable vector)**.** *A vector $\mathbf{v} \in \mathbb{R}^n$ is called* observable *if the value $Y(\mathbf{v}) = \mathbf{v}^\top \mathbf{y}$ is uniquely determined by the observed data, equivalently by $\mathbf{b} = \mathbf{M}\mathbf{y}$.*

Proposition 3.2. *A vector $\mathbf{v} \in \mathbb{R}^n$ is* observable *if and only if $\mathbf{v} \in \textsc{Row}(\mathbf{M})$.*

Corollary 3.3. *A nonempty subset $S \subseteq [n]$ is* observable *if and only if $\mathbf{1}_S \in \textsc{Row}(\mathbf{M})$.*

Constructible predictors from observable information. We are now ready to formally define constructible predictors.

Definition 3.7 (Constructible predictors)**.** *A finitely supported predictor g is said to be* constructible *from observable information if its reduced bag can be written as $\mathcal{V}(g) = \{\mathbf{v}_{g,1}, \ldots, \mathbf{v}_{g,m_g}\}$ for some finite family of vectors satisfying: (i) $\mathbf{v}_{g,r} \in \textsc{Row}(\mathbf{M}) \cap \mathbb{R}^n_+ \setminus \{\mathbf{0}\}$ for every $r \in [m_g]$; (ii) $\sum_{r \in [m_g]} \mathbf{v}_{g,r} = \mathbf{1}$ coordinatewise. The prediction attached to the signal $\mathbf{v}_{g,r}$ is $\mu(\mathbf{v}_{g,r})$. We write $\mathcal{G}$ for the set of all such predictors.*

Condition (i) ensures that each signal of g is identifiable from the observable linear information. Condition (ii) ensures that the family $\{\mathbf{v}_{g,r}\}_{r \in [m_g]}$ defines a valid stochastic kernel via $T_g(r \mid x_i) \triangleq \mathbf{v}_{g,r}(i)$.

Remark 3.8. *Definition 3.7 is the formal notion of constructibility used in the remainder of the paper. In particular, whenever we write $\mathcal{G}$, we mean the family of predictors constructible from observable linear information in the sense of Definition 3.7.*

Corollary 3.4 (Constructible deterministic predictors)**.** *Let g be a deterministic predictor, and let $\Pi_g = \{A_1, \ldots, A_{m_g}\}$ be its signal partition. Then $g \in \mathcal{G}^{\text{DET}}$ if and only if $\mathbf{1}_{A_r} \in \textsc{Row}(\mathbf{M})$ for every $r \in [m_g]$. In that case, the prediction on the signal cell A_r is $\mu(A_r)$.*

Remark 3.9. *For deterministic predictors under informational dominance, the primitive objects are the signal cells in Π_g. Distinct signal cells are* not *merged merely because they display the same numeric prediction. This is the interpretation that will be used in Section 4.*

4 Hardness and Efficient Algorithms for Deterministic Predictors

In this section, we study the search problem PREDICTORREFINE restricted to deterministic inputs and deterministic outputs; that is, the input predictors $(f_j)_{j \in [k]}$ and the output predictors $(g_j)_{j \in [k]}$ are all required to be deterministic predictors.

Fix a feature distribution $\lambda = (\lambda_i)_{i \in [n]}$ and deterministic input predictors $(f_j)_{j \in [k]}$. Recall that $\mathcal{G}^{\text{DET}} \subseteq \mathcal{G}$ denotes the set of constructible deterministic predictors, and $\mathcal{G}^{\text{DET}}_{\succ} \subseteq \mathcal{G}^{\text{DET}}$ denotes the set of constructible deterministic predictors that are undominated by any other constructible deterministic predictors. By Corollary 3.4, a deterministic predictor g belongs to $\mathcal{G}^{\text{DET}}$ if and only if $\mathbf{1}_A \in \textsc{Row}(\mathbf{M})$ for every signal cell $A \in \Pi_g$. In that case, the prediction on cell A is the calibrated value $\mu(\mathbf{1}_A)$. By Remark 3.9, the primitive objects in this section are the signal cells in Π_g. With this simplification in place, informational dominance for deterministic predictors can be characterized entirely in terms of the refinement relation between their signal partitions.

Lemma 4.1 (Dominance for deterministic predictors)**.** *Let g, h be deterministic predictors, and let Π_g and Π_h denote their signal partitions. Then we have $g \succ h$ if and only if Π_g* strictly refines *Π_h, that is, for every cell $A \in \Pi_g$, there exists a cell $A' \in \Pi_h$ such that $A \subseteq A'$, and moreover $\Pi_g \neq \Pi_h$.*

4.1 NP-Hardness for Solving PREDICTORREFINE

To establish the hardness of the search problem PREDICTORREFINE, we first consider the following decision problem IFREFINABLE as a sub-problem toward solving PREDICTORREFINE.

IFREFINABLE:
Input: a feature distribution $\lambda = (\lambda_i)_{i\in[n]}$ and k deterministic predictors $(f_j)_{j\in[k]}$.
Question: does there exist a predictor $g \in \mathcal{G}_{\succ}^{\text{DET}}$ such that $g \succ f_1$?

Theorem 4.2. *The decision problem* IFREFINABLE *is* NP*-hard.*

Corollary 4.3. *The search problem* PREDICTORREFINE*, restricted to deterministic input predictors and deterministic output predictors, is* NP*-hard.*

At a high level, the proof of Theorem 4.2 proceeds by a reduction from the following NP-complete problem, which we denote by X3C3 (see, e.g., Garey and Johnson, 2002).

Exact 3-cover with multiplicity 3 (X3C3):
Input: a set $\mathcal{U} = \{u_1, \ldots, u_m\}$ and a collection $\mathcal{C} = \{C_1, \ldots, C_d\}$ of 3-element subsets of $\mathcal{U}$ such that every element $u_r \in \mathcal{U}$ belongs to exactly three sets, and m is a multiple of 3.
Question: does there exist an exact cover of $\mathcal{U}$?

The problem X3C3 encodes the incidence matrix $\mathbf{X} \in \{0,1\}^{m\times d}$ into the observable linear space $\text{ROW}(\mathbf{M})$. The construction creates two feature blocks $\mathcal{X} = \mathcal{X}_1 \sqcup \mathcal{X}_2$, takes the first predictor to have partition $\Pi_{f_1} = \{\mathcal{X}_1, \mathcal{X}_2\}$, and for each set $\mathcal{C}_j$ introduces a two-cell predictor built from $R_j = \{v_j\} \cup \{w_r : u_r \in \mathcal{C}_j\}$. As a result, by Lemma 4.1 and Corollary 3.4, a strict deterministic improvement of f_1 is equivalent to splitting one of the two cells $\mathcal{X}_1$ or $\mathcal{X}_2$ by a nonempty proper observable subset S, namely a subset satisfying $\mathbf{1}_S \in \text{ROW}(\mathbf{M})$. The key structural step is then to characterize exactly when such subsets exist: Lemma D.2 shows that $\mathcal{X}_2$ admits no nontrivial observable subset, while Lemma D.1 shows that for $S \subseteq \mathcal{X}_1$ with indicator vector $\mathbf{z} \in \{0,1\}^d$, the condition $\mathbf{1}_S \in \text{ROW}(\mathbf{M})$ is equivalent to $\mathbf{Xz} = \beta\mathbf{1}_m$ for some scalar β. The four possible values $\beta \in \{0,1,2,3\}$ then separate the two trivial cases from the two exact-cover cases, and Lemma D.3 packages this into the equivalence between an exact cover of the X3C3 instance, the existence of some constructible predictor $g \in \mathcal{G}^{\text{DET}}$ with $g \succ f_1$, and the existence of an undominated one $g^\star \in \mathcal{G}_{\succ}^{\text{DET}}$ with $g^\star \succ f_1$. The missing proofs and algorithms are deferred to Section D.

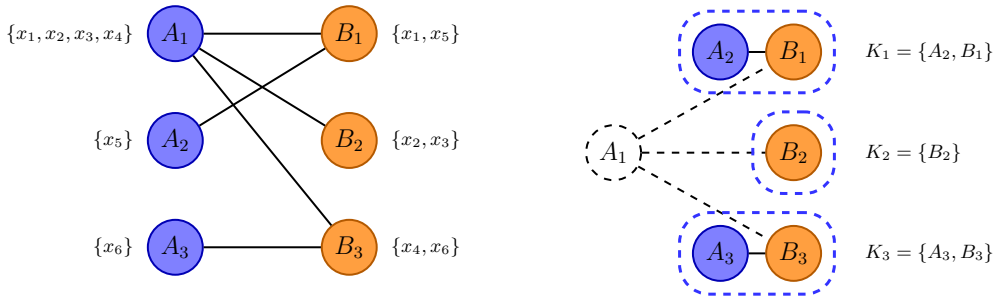
4.2 Efficient Algorithm for Solving PREDICTORREFINE with $k = 2$

In this subsection, we show that there exists a polynomial-time algorithm that can solve the deterministic version of problem PREDICTORREFINE when $k = 2$.

Theorem 4.4. *Assume $k = 2$. Then there exists a polynomial-time algorithm (Algorithms 1 and 2) that solves the deterministic version of* PREDICTORREFINE*.*

We write $\Pi_{f_1} = \{A_{1,\ell}\}_{\ell\in[m_1]}, \Pi_{f_2} = \{A_{2,r}\}_{r\in[m_2]}$ for the signal partitions of the two deterministic input predictors f_1 and f_2. By Remark 3.5, the rows of the signal-feature matrix $\mathbf{M}$ are exactly the indicator vectors $\mathbf{1}_{A_{1,\ell}}$ for $\ell \in [m_1]$ and $\mathbf{1}_{A_{2,r}}$ for $r \in [m_2]$. For notational simplicity, for every input signal cell $A \in \Pi_{f_1} \cup \Pi_{f_2}$, define its observable cell label mass by $L(A) \triangleq Y(\mathbf{1}_A) = \mu(\mathbf{1}_A)\,\lambda(\mathbf{1}_A)$ which is determined by the observed input predictor and the feature distribution.

Define the overlap graph G as the bipartite graph with vertex set $\mathcal{I}(G) \triangleq \Pi_{f_1} \sqcup \Pi_{f_2}$ and with an edge between $A_{1,\ell}$ and $A_{2,r}$ iff $A_{1,\ell} \cap A_{2,r} \neq \emptyset$ (see Figure 1a for an illustration). At a high level, the proof of Theorem 4.4 exploits the fact that when $k = 2$, the observable structure inside a fixed input cell A is completely controlled by the overlap graph G between the signal partitions Π_{f_1} and Π_{f_2}. For each cell A on one side, we remove the vertex A from the overlap graph G and use the connected components K of $G - A$ to decompose the cell A into its K-branches A_K (see Figure 1b for an illustration). Lemma E.1 shows that these branch pieces are exactly the right atomic objects for refinement: part (i) shows that they partition the cell A, part (ii) shows that each A_K is observable and that its label mass can be computed from the observable cell label masses $L(\cdot)$, and part (iii) shows that every observable subset $S \subseteq A$ must be a union of such branch pieces.



(a) Original overlap graph G (edge iff the two cells intersect).

(b) Connected components of $G - A_1$.

Figure 1: Illustration of Algorithm 1. In this example, the two deterministic predictors f_1 and f_2 induce signal partitions over the feature set. We write each signal cell by the feature indices it contains: $\Pi_{f_1} = \{A_1, A_2, A_3\}$, where $A_1 = \{x_1, x_2, x_3, x_4\}$, $A_2 = \{x_5\}$, and $A_3 = \{x_6\}$; and $\Pi_{f_2} = \{B_1, B_2, B_3\}$, where $B_1 = \{x_1, x_5\}$, $B_2 = \{x_2, x_3\}$, and $B_3 = \{x_4, x_6\}$. Figure 1a shows the overlap graph G, whose vertices are the signal cells of f_1 and f_2, and where an edge indicates a nonempty intersection between two signal cells. The cell A_1 is an articulation vertex. Figure 1b shows that $G - A_1$ has three connected components: $K_1 = \{A_2, B_1\}$, $K_2 = \{B_2\}$, and $K_3 = \{A_3, B_3\}$. By Algorithm 1, the cell A_1 is therefore split into the branch pieces $A_1^{K_1} = A_1 \cap B_1 = \{x_1\}$, $A_1^{K_2} = A_1 \cap B_2 = \{x_2, x_3\}$, and $A_1^{K_3} = A_1 \cap B_3 = \{x_4\}$. The remaining signal cells $A_2 = \{x_5\}$ and $A_3 = \{x_6\}$ are unchanged. Hence the finest constructible deterministic refinement of f_1 is $\widehat{\Pi}_1 = \big\{\{x_1\}, \{x_2, x_3\}, \{x_4\}, \{x_5\}, \{x_6\}\big\}$.

Consequently, if we refine every cell $A \in \Pi_{f_j}$ into all of its branch pieces, we obtain a canonical partition, which we denote by $\widehat{\Pi}_j$, which is the finest constructible deterministic refinement of Π_{f_j}. Lemma 4.1 then converts this structural statement into the desired dominance conclusion: if $\widehat{\Pi}_j \neq \Pi_{f_j}$, the associated predictor $g_j^\star$ strictly informationally-dominates f_j; if $\widehat{\Pi}_j = \Pi_{f_j}$, then no constructible deterministic predictor can strictly informationally-dominate f_j. Algorithms 1 and 2 simply implement this canonical refinement and test which of these two cases occurs (see Figure 1 for an example and illustration). We defer the missing proofs and algorithms to Section E.

5 Efficient Algorithms for Randomized Predictors

In this section, we show that once randomized outputs are allowed, the problem PREDICTORREFINE becomes polynomial-time solvable. For complexity statements, we assume that all entries of λ, all entries of the input signal vectors, and all input predictions are rational numbers encoded in binary.

Theorem 5.1. *There exists an algorithm (see Algorithm 5) that runs in time polynomial in the input size and solves* PREDICTORREFINE.

At a high level, Algorithm 5 works by refining each input reduced bag $\mathcal{V}(f_j) = \{\mathbf{v}_{j,\ell}\}_{\ell \in [m_j]}$ into a finer bag of constructible signal vectors that still sums to the same row. The key geometric observation is that all constructible signal vectors lie in the same cone $\mathcal{M}$, which we refer to as the constructible signal cone, and that the atomic directions of this cone are its extreme rays:

$$\mathcal{M} \triangleq \textsc{Row}(\mathbf{M}) \cap \mathbb{R}^n_+ .$$

Starting from an input row $\mathbf{v}_{j,\ell} \in \mathcal{M} \setminus \{\mathbf{0}\}$, the algorithm repeatedly decomposes $\mathbf{v}_{j,\ell}$ into finer vectors in $\mathcal{M}$ until every remaining piece lies on an extreme ray of $\mathcal{M}$. Such a refinement can only increase information, because one can always forget the finer signal and map each refined piece back to its parent signal through a feature-independent garbling. The geometric part of the argument is carried out in Lemmas F.1 to F.3, which show that extreme-ray pieces can be found via support-minimality and obtained in polynomial time through linear-feasibility tests. The final step is Lemma F.4: once every row of a reduced bag lies on an extreme ray of $\mathcal{M}$, the resulting constructible predictor already belongs to $\mathcal{G}_\succ$. Consequently, for each input predictor f_j, the algorithm either leaves $\mathcal{V}(f_j)$ unchanged, which certifies that $f_j \in \mathcal{G}_\succ$, or produces a strict extreme-ray refinement, which yields a constructible predictor $g_j \in \mathcal{G}_\succ$ with $g_j \succ f_j$. We defer the missing proofs to Section F.

6 Conclusion and Future Research

In this paper, we propose and study the problem of algorithmic predictor refinement. We formulate an abstract, signal-based framework that captures the technical essence of the problem. Based on the framework, we present comprehensive algorithmic results for refinement into both deterministic and randomized predictors: the deterministic problem admits an efficient algorithm when there are 2 input predictors, and becomes intractable with multiple input predictors; the randomized problem admits an efficient algorithm for any number of input predictors. For future research, several interesting questions remain open. In particular, it remains unclear whether a proper notion of approximation can help, particularly in the deterministic problem with multiple input predictors. It also remains unclear whether there exist efficient algorithms for the deterministic problem with any *fixed* number of input predictors.

References

János Aczél and Carl Wagner. A characterization of weighted arithmetic means. *SIAM Journal on Algebraic Discrete Methods*, 1(3):259–260, September 1980.

Rui Ai, Yuqi Pan, David Simchi-Levi, Milind Tambe, and Haifeng Xu. Beyond majority voting: LLM aggregation by leveraging higher-order information. arXiv preprint arXiv:2510.01499, 2025.

Itai Arieli, Yakov Babichenko, and Rann Smorodinsky. Robust forecast aggregation. *Proceedings of the National Academy of Sciences*, 115(52):E12135–E12143, December 2018.

John M Bates and Clive WJ Granger. The combination of forecasts. *Journal of the operational research society*, 20(4):451–468, 1969.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Robert F. Bordley. A multiplicative formula for aggregating probability assessments. *Management Science*, 28(10):1137–1148, October 1982.

Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D Procaccia. *Handbook of computational social choice*. Cambridge University Press, 2016.

Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

Robert T Clemen. Combining forecasts: A review and annotated bibliography. *International journal of forecasting*, 5(4):559–583, 1989.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1280–1297, 2024.

A Philip Dawid. The well-calibrated bayesian. *Journal of the American statistical Association*, 77 (379):605–610, 1982.

Henrique De Oliveira, Yuhta Ishii, and Xiao Lin. Robust merging of information. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, EC '21, pages 341–342, New York, NY, USA, 2021. ACM.

Morris H DeGroot and Stephen E Fienberg. The comparison and evaluation of forecasters. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 32(1-2):12–22, 1983.

Franz Dietrich and Christian List. Probabilistic opinion pooling. In Alan Hájek and Christopher Hitchcock, editors, *The Oxford Handbook of Probability and Philosophy*, pages 519–542. Oxford University Press, Oxford, UK, 2016.

Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

Philip Ernst, Robin Pemantle, Ville Satopää, and Lyle Ungar. Bayesian aggregation of two forecasts in the partial information framework. *Statistics & Probability Letters*, 119:170–180, December 2016.

Dean P Foster and Rakesh V Vohra. Calibrated learning and correlated equilibrium. *Games and Economic Behavior*, 21(589):40–55, 1997.

Yoav Freund. Boosting a weak learning algorithm by majority. *Information and computation*, 121 (2):256–285, 1995.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2): 337–407, 2000.

Rafael Frongillo, Yiling Chen, and Ian Kash. Elicitation for aggregation. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, volume 29 of *AAAI '15*, Austin, Texas, 2015. AAAI Press.

Rafael Frongillo, Mary Monroe, Eric Neyman, and Bo Waggoner. Robust forecast aggregation via additional queries. arXiv preprint arXiv:2512.05271, 2025.

Michael R Garey and David S Johnson. *Computers and intractability*, volume 29. wh freeman New York, 2002.

Christian Genest. A characterization theorem for externally Bayesian groups. *The Annals of Statistics*, 12(3):1100–1105, September 1984.

Tilmann Gneiting, Fadoua Balabdaoui, and Adrian E Raftery. Probabilistic forecasts, calibration and sharpness. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 69(2): 243–268, 2007.

Google Ads Help. About expected click-through rate in google ads. Google Ads Help. URL https://support.google.com/sa360/answer/9351019. Accessed: May 2, 2026.

Google Cloud. Get online inferences from a custom trained model. https://docs.cloud.google.com/vertex-ai/docs/predictions/get-online-predictions. Accessed: May 2, 2026.

Yongkang Guo and Yuqing Kong. Robust aggregation with adversarial experts. In *Proceedings of the ACM Web Conference 2025*, WWW '25. ACM, 2025.

Yongkang Guo, Jason D. Hartline, Zhihuan Huang, Yuqing Kong, Anant Shah, and Fang-Yi Yu. Algorithmic robust forecast aggregation. In *Proceedings of the 26th ACM Conference on Economics and Computation*, EC '25, New York, NY, USA, 2025. ACM.

Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, et al. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the eighth international workshop on data mining for online advertising*, pages 1–9, 2014.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

Yuqing Kong, Shu Wang, and Ying Wang. The surprising benefits of base rate neglect in robust aggregation. In *Proceedings of the 25th ACM Conference on Economics and Computation*, EC '24, pages 1131–1131, New York, NY, USA, 2024. ACM.

Gilat Levy and Ronny Razin. Combining forecasts in the presence of ambiguity over correlation structures. *Journal of Economic Theory*, 199:105075, January 2022.

Nick Littlestone and Manfred K Warmuth. The weighted majority algorithm. *Information and computation*, 108(2):212–261, 1994.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

H Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. Ad click prediction: a view from the trenches. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1222–1230, 2013.

Eric Neyman and Tim Roughgarden. Are you smarter than a random expert? The robust aggregation of substitutable signals. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, EC '22, pages 990–1012, New York, NY, USA, 2022. ACM.

Asa B. Palley and Jack B. Soll. Extracting the wisdom of crowds when information is shared. *Management Science*, 65(5):2291–2309, May 2019.

Yuqi Pan, Zhaohua Chen, and Yuqing Kong. Robust decision aggregation with second-order information. In *Proceedings of the ACM Web Conference 2024*, pages 77–88, 2024.

Drazen Prelec. A Bayesian truth serum for subjective data. *Science*, 306(5695):462–466, October 2004. doi: 10.1126/science.1102081.

Dražen Prelec, H. Sebastian Seung, and John McCoy. A solution to the single-question crowd wisdom problem. *Nature*, 541(7638):532–535, January 2017.

Roopesh Ranjan and Tilmann Gneiting. Combining probability forecasts. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 72(1):71–91, 2010.

Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. *Journal of machine learning research*, 11(4), 2010.

Ville A. Satopää, Jonathan Baron, Dean P. Foster, Barbara A. Mellers, Philip E. Tetlock, and Lyle H. Ungar. Combining multiple probability predictions using a simple logit model. *International Journal of Forecasting*, 30(2):344–356, April 2014.

Ville A. Satopää, Robin Pemantle, and Lyle H. Ungar. Modeling probability forecasts via information diversity. *Journal of the American Statistical Association*, 111(516):1623–1633, December 2016.

Robert E Schapire. The strength of weak learnability. *Machine learning*, 5(2):197–227, 1990.

Mervyn Stone. The opinion pool. *The Annals of Mathematical Statistics*, pages 1339–1342, 1961.

Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction {APIs}. In *25th USENIX security symposium (USENIX Security 16)*, pages 601–618, 2016.

Juntao Wang, Yang Liu, and Yiling Chen. Forecast aggregation via peer prediction. In *Proceedings of the 9th AAAI Conference on Human Computation and Crowdsourcing*, volume 9 of *HCOMP '21*, pages 131–142. AAAI Press, 2021.

Tom Wilkening, Marcellin Martinie, and Piers D. L. Howe. Hidden experts in the crowd: Using meta-predictions to leverage expertise in single-question prediction problems. *Management Science*, 68(1):487–508, January 2022. doi: 10.1287/mnsc.2020.3934.

David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

A Missing Example in Section 1

Consider the two predictors in our first introductory example (in their abstract forms): $f_1 : \{\{\mathrm{A}, \mathrm{B}\} \mapsto \text{Signal 1}, \{\mathrm{C}\} \mapsto \text{Signal 2}\}$, $f_2 : \{\{\mathrm{A}\} \mapsto \text{Signal 3}, \{\mathrm{B}, \mathrm{C}\} \mapsto \text{Signal 4}\}$. We already know that the conditional mean on all 3 feature vectors can be uniquely determined from f_1 and f_2. To formalize this fact, we write both input predictors as bags of vectors. For f_1, we write $(1, 1, 0)$ for Signal 1, because the corresponding subset contains A ("Feature 1") and B ("Feature 2"), but not C ("Feature 3"); similarly, we write $(0, 0, 1)$ for Signal 2. Accordingly, f_1 is represented by the associated bag of vectors $\mathcal{V}(f_1) = \{(1, 1, 0), (0, 0, 1)\}$, where each vector corresponds to one prediction, and the sum of all vectors is the all-one vector. Similarly, f_2 is represented by $\mathcal{V}(f_2) = \{(1, 0, 0), (0, 1, 1)\}$. Note that a predictor, modulo permutations of the signals (which is immaterial), corresponds bijectively to its bag-of-vectors representation.

As before, assume that the 3 features are equally frequent in the population (which is in fact without loss of generality). Now, to see why the conditional expectations on all features (and in particular, on feature B) are uniquely determined, simply observe that they can be pinned down by solving the following (overdetermined but guaranteed to be consistent) linear system:

$$\begin{aligned}(1, 1, 0) \cdot \mathbf{p} &= \|(1, 1, 0)\|_1 \cdot p_1 \\ (0, 0, 1) \cdot \mathbf{p} &= \|(0, 0, 1)\|_1 \cdot p_2 \\ (1, 0, 0) \cdot \mathbf{p} &= \|(1, 0, 0)\|_1 \cdot p_3 \\ (0, 1, 1) \cdot \mathbf{p} &= \|(0, 1, 1)\|_1 \cdot p_4,\end{aligned}$$

where $\mathbf{p} = (p_A, p_B, p_C)^\top$ is the conditional expectations on individual features, and $p_1, \ldots, p_4$ are the conditional expectations corresponding to the predictions (Signal 1, …, Signal 4). The first two equations correspond to $\mathcal{V}(f_1)$, and the last two correspond to $\mathcal{V}(f_2)$. p_B in particular can be uniquely determined, because the vector $(0, 1, 0)$ is in the linear span of $\mathcal{V}(f_1) \cup \mathcal{V}(f_2)$. Accordingly, the predictor $g : \{\{\mathrm{A}\} \mapsto \text{Signal 5}, \{\mathrm{B}\} \mapsto \text{Signal 6}, \{\mathrm{C}\} \mapsto \text{Signal 7}\}$ is constructible, because all 3 vectors corresponding to the 3 predictions, $(1, 0, 0)$, $(0, 1, 0)$, and $(0, 0, 1)$, are in the span of $\mathcal{V}(f_1) \cup \mathcal{V}(f_2)$. Note that in the above reasoning, the specific values of $p_1, \ldots, p_4$ are entirely irrelevant, which is also why we abstract them away in our definition of predictors.

B Further Related Work

Prediction aggregation, or forecast aggregation, and closely related formulations have been studied across several communities, including machine learning, statistics, economics, and theoretical computer science. Early foundational work includes opinion pooling and forecast combination (Stone, 1961; Bates and Granger, 1969), as well as classical reviews of forecast combination (Clemen, 1989). The broader literature has since developed along different methodological directions.

In machine learning, aggregation is often studied through ensemble methods, where multiple predictors/classifiers are combined to improve predictive performance. Classical examples include stacking, bagging, boosting, and random forests (Wolpert, 1992; Breiman, 1996; Schapire, 1990; Freund, 1995; Freund and Schapire, 1997; Friedman et al., 2000; Breiman, 2001; Dietterich, 2000). A related online-learning perspective studies prediction with expert advice, where multiplicative-weights-type algorithms combine experts' predictions while achieving low regret relative to the best fixed expert (Littlestone and Warmuth, 1994; Cesa-Bianchi and Lugosi, 2006).

In statistics, forecast aggregation has been studied both axiomatically and probabilistically. The axiomatic literature imposes desiderata on aggregation rules, such as unanimity preservation and variants of independence, and characterizes rules such as linear or externally Bayesian pooling under corresponding assumptions (Aczél and Wagner, 1980; Genest, 1984; Dietrich and List, 2016). The probabilistic and Bayesian literature instead models how forecasts are generated from underlying signals and then derives aggregation rules by Bayesian updating or parametric estimation; see, for example, Satopää et al. (2014); Frongillo et al. (2015); Ernst et al. (2016); Satopää et al. (2016); Ranjan and Gneiting (2010).

A common feature of much of this literature is that the reported forecasts are interpreted as probabilistic beliefs induced by the forecasters' information (Bordley, 1982; Gneiting et al., 2007; Satopää et al., 2016; Arieli et al., 2018; Guo et al., 2025). Under this interpretation, a forecast is calibrated

with respect to the forecaster's signal: it represents the posterior mean of the target conditional on the information available to that forecaster. Our focus on calibrated predictors is therefore aligned with this standard informational view of forecasts.

In economics and theoretical computer science, a recent line of work takes a robust perspective and asks for aggregators that perform well across large classes of possible information structures, especially when the correlation structure among experts' signals is unknown or misspecified (Arieli et al., 2018; De Oliveira et al., 2021; Levy and Razin, 2022; Neyman and Roughgarden, 2022; Kong et al., 2024; Guo et al., 2025; Guo and Kong, 2025; Frongillo et al., 2025). A complementary strand leverages second-order or higher-order information, such as agents' beliefs about others' answers, to improve aggregation in finite populations (Prelec, 2004; Prelec et al., 2017; Palley and Soll, 2019; Wang et al., 2021; Wilkening et al., 2022; Pan et al., 2024; Ai et al., 2025).

Our work differs from these lines in both its objective and its information constraint. Rather than designing a rule that maps several reported forecasts into a single aggregate forecast, or evaluating such a rule under a loss or regret criterion, we study when and how calibrated predictors can be refined using only the observable information contained in the input predictors, without access to the underlying Bayes probabilities. This leads to a distinct constructibility problem: the goal is to move from numerical aggregation to information refinement, and to characterize and compute maximally informative constructible refinements under an information-theoretic dominance relation.

C Missing Proofs in Section 3

Proof of Lemma 3.1. We write the signal spaces $\Sigma_f = [m_f]$ and $\Sigma_g = [m_g]$, and let $\mathcal{V}(f) = \{\mathbf{v}_{f,\ell}\}_{\ell\in[m_f]}$ and $\mathcal{V}(g) = \{\mathbf{v}_{g,r}\}_{r\in[m_g]}$ be the corresponding reduced bags.

Assume first that $f \succ g$ in the sense of Definition 2.1. Then there exists a stochastic kernel $H : \Sigma_f \to \Delta(\Sigma_g)$ such that

$$T_g(r \mid x_i) = \sum\nolimits_{\ell\in[m_f]} H(r \mid \ell)\, T_f(\ell \mid x_i) \quad \text{for all } i \in [n],\ r \in [m_g]\ .$$

Define the matrix $\mathbf{H}$ with the entry $\mathbf{H}_{\ell,r} \triangleq H(r \mid \ell)$. Then $\mathbf{H} = (\mathbf{H}_{\ell,r})$ is row-stochastic, and for every $r \in [m_g]$ and $i \in [n]$,

$$\mathbf{v}_{g,r}(i) = T_g(r \mid x_i) = \sum\nolimits_{\ell\in[m_f]} \mathbf{H}_{\ell,r}\, T_f(\ell \mid x_i) = \sum\nolimits_{\ell\in[m_f]} \mathbf{H}_{\ell,r}\, \mathbf{v}_{f,\ell}(i)\ .$$

Thus, the first condition in Definition 3.3 holds. The second condition is translated identically. Thus $f \succ g$ also holds in bag form.

Conversely, suppose Definition 3.3 holds. Define a stochastic kernel $H : \Sigma_f \to \Delta(\Sigma_g)$ by $H(r \mid \ell) \triangleq \mathbf{H}_{\ell,r}$. Then, for every $i \in [n]$ and $r \in [m_g]$,

$$T_g(r \mid x_i) = \mathbf{v}_{g,r}(i) = \sum\nolimits_{\ell\in[m_f]} \mathbf{H}_{\ell,r}\, \mathbf{v}_{f,\ell}(i) = \sum\nolimits_{\ell\in[m_f]} H(r \mid \ell)\, T_f(\ell \mid x_i)\ .$$

So g is obtained from f by a feature-independent garbling. Again, the nonexistence of a reverse row-stochastic matrix is exactly the nonexistence of a reverse garbling kernel. Thus, the two definitions are equivalent. □

Proof of Proposition 3.2. Assume first that $\mathbf{v} \in \textsc{Row}(\mathbf{M})$. Then $\mathbf{v} = \mathbf{M}^\top \alpha$ for some $\alpha \in \mathbb{R}^N$, and therefore

$$Y(\mathbf{v}) = \mathbf{v}^\top \mathbf{y} = \alpha^\top \mathbf{M}\mathbf{y} = \alpha^\top \mathbf{b}\ ,$$

which is determined by $\mathbf{b}$.

Conversely, suppose $\mathbf{v} \notin \textsc{Row}(\mathbf{M})$. Since $\textsc{Row}(\mathbf{M}) = (\ker \mathbf{M})^\perp$, there exists $\mathbf{z} \in \ker \mathbf{M}$ such that $\mathbf{v}^\top \mathbf{z} \neq 0$. Fix any $\mathbf{y}^0 \in \mathbb{R}^n$ satisfying $\mathbf{M}\mathbf{y}^0 = \mathbf{b}$. Then, for every scalar $\beta \in \mathbb{R}$,

$$\mathbf{M}(\mathbf{y}^0 + \beta\mathbf{z}) = \mathbf{M}\mathbf{y}^0 + \beta\, \mathbf{M}\mathbf{z} = \mathbf{b}\ .$$

Thus, $\mathbf{y}^0$ and $\mathbf{y}^0 + \beta\mathbf{z}$ yield the same observation $\mathbf{b}$. However,

$$\mathbf{v}^\top(\mathbf{y}^0 + \beta\mathbf{z}) = \mathbf{v}^\top\mathbf{y}^0 + \beta\, \mathbf{v}^\top\mathbf{z}$$

depends non-trivially on β because $\mathbf{v}^\top\mathbf{z} \neq 0$. Thus $Y(\mathbf{v})$ is not uniquely determined by $\mathbf{b}$. □

Proof of Corollary 3.3. Apply Proposition 3.2 to the vector $\mathbf{1}_S$. □

Proof of Corollary 3.4. If predictor g is deterministic, then each feature produces exactly one signal with probability one. Thus, the reduced bag of predictor g is

$$\mathcal{V}(g) = \{\mathbf{1}_{A_1}, \ldots, \mathbf{1}_{A_{m_g}}\},$$

where $\Pi_g = \{A_1, \ldots, A_{m_g}\}$ is a partition of $[n]$. Thus, Definition 3.7 is equivalent to the condition that $\mathbf{1}_{A_r} \in \textsc{Row}(\mathbf{M})$ for every $r \in [m_g]$, and the attached prediction is $\mu(\mathbf{1}_{A_r})$. □

D Missing Proofs in Section 4.1

Proof of Lemma 4.1. Suppose first that $g \succ h$. By Definition 2.1, there exists a feature-independent garbling from predictor g to predictor h.

Fix any signal cell $A \in \Pi_g$. If two features $x_i, x_{i'} \in A$, then they induce the same signal of g. After passing through the same feature-independent garbling kernel, they therefore induce the same distribution over signals of h. Since h is deterministic, each of x_i and $x_{i'}$ induces a point mass on a single signal of h. Thus, these point masses must coincide, so x_i and $x_{i'}$ lie in the same signal cell of h. Therefore every cell of Π_g is contained in a cell of Π_h, and thus Π_g refines Π_h.

We next show that the refinement is strict. If the signal partition Π_h also refined Π_g, then applying the same argument with the roles of g and h reversed would imply that there exists a feature-independent garbling from h to g, contradicting the strictness in Definition 2.1. Thus, Π_h does not refine Π_g, and therefore Π_g strictly refines Π_h.

Conversely, suppose that Π_g strictly refines Π_h. Then every cell of Π_g is contained in a unique cell of Π_h. Define a feature-independent garbling from g to h by mapping each signal of g deterministically to the signal of h corresponding to that containing cell. This yields a feature-independent garbling from g to h.

It remains to show that there is no feature-independent garbling from h to g. If such a garbling existed, then by the first part of the proof, every cell of Π_h would be contained in a cell of Π_g; that is, Π_h would refine Π_g. This contradicts the assumption that Π_g strictly refines Π_h. Thus, we have $g \succ h$. □

Proof of Theorem 4.2. Let $\mathbf{X} \in \{0,1\}^{m \times d}$ be the incidence matrix of the X3C3 instance, where $\mathbf{X}_{r,j} = 1$ if and only if $u_r \in C_j$. Since every column and every row of $\mathbf{X}$ has sum 3, double-counting incidences gives

$$\sum\nolimits_{r \in [m]} \sum\nolimits_{j \in [d]} \mathbf{X}_{r,j} = 3d = 3m,$$

and hence $d = m$.

Feature distribution and initial predictors. We construct an instance of IFREFINABLE as follows. Create $n = d + m = 2m$ features and partition the feature space into two blocks

$$\mathcal{X} = \mathcal{X}_1 \sqcup \mathcal{X}_2, \quad \mathcal{X}_1 \triangleq \{v_1, \ldots, v_d\}, \quad \mathcal{X}_2 \triangleq \{w_1, \ldots, w_m\}.$$

Let the feature distribution be uniform: $\lambda_i = 1/n$ for every $i \in [n]$. We define $k = d + 1 = m + 1$ deterministic input predictors $(f_j)_{j \in [k]}$.

- Predictor f_1 has signal partition $\Pi_{f_1} = \{\mathcal{X}_1, \mathcal{X}_2\}$, and predicts $\mu(\mathbf{1}_A)$ on each signal cell $A \in \Pi_{f_1}$.
- For each $j \in [d]$, let $R_j \triangleq \{v_j\} \cup \{\, w_r : u_r \in C_j \,\}$, and let the predictor f_{j+1} have signal partition $\Pi_{f_{j+1}} = \{R_j, R_j^c\}$ and predict $\mu(\mathbf{1}_A)$ on each signal cell $A \in \Pi_{f_{j+1}}$.

Finally, enumerate the n features arbitrarily as $x_1, \ldots, x_n$, and let $q_i \leftarrow (n+1)^{-i}$ be the Bayes probability for the i-th feature.

Row-space description. Let $\mathbf{M}$ be the signal-feature matrix from Definition 3.4 for the constructed input predictors. Since all input predictors are deterministic, Remark 3.5 implies that the rows of $\mathbf{M}$

are exactly the indicator vectors of the initial signal cells. Because each signal partition $\Pi_{f_{j+1}}$ has signal cells R_j and R_j^c, and $\mathbf{1}_{R_j^c} = \mathbf{1}_{\mathcal{X}_1} + \mathbf{1}_{\mathcal{X}_2} - \mathbf{1}_{R_j}$, we obtain

$$\textsc{Row}(\mathbf{M}) = \textsc{span}\left\{\mathbf{1}_{\mathcal{X}_1}, \mathbf{1}_{\mathcal{X}_2}, \mathbf{1}_{R_1}, \ldots, \mathbf{1}_{R_d}\right\} . \tag{1}$$

We now characterize the observable subsets inside $\mathcal{X}_1$ and $\mathcal{X}_2$.

Lemma D.1. *Let $S \subseteq \mathcal{X}_1$, and let $\mathbf{z} \in \{0,1\}^d$ be the indicator vector of S, namely $\mathbf{z}_j = 1$ if and only if $v_j \in S$. Then we have $\mathbf{1}_S \in \textsc{Row}(\mathbf{M})$ if and only if $\mathbf{X}\mathbf{z} = \beta \mathbf{1}_m$ for some scalar $\beta \in \mathbb{R}$.*

Proof. Assume first that $\mathbf{1}_S \in \textsc{Row}(\mathbf{M})$. By Eqn. (1), there exist scalars $a, b, \alpha_1, \ldots, \alpha_d$ such that

$$\mathbf{1}_S = a\mathbf{1}_{\mathcal{X}_1} + b\mathbf{1}_{\mathcal{X}_2} + \sum\nolimits_{j\in[d]} \alpha_j \mathbf{1}_{R_j} . \tag{2}$$

Evaluating Eqn. (2) at the feature v_j gives

$$\mathbf{z}_j = a + \alpha_j, \quad \text{so} \quad \alpha_j = \mathbf{z}_j - a . \tag{3}$$

Now evaluate Eqn. (2) at the feature w_r. Since $w_r \notin \mathcal{X}_1$, $w_r \in \mathcal{X}_2$, and $w_r \in R_j$ if and only if $u_r \in C_j$, we obtain

$$0 = b + \sum\nolimits_{j\in[d]} \mathbf{X}_{r,j}\alpha_j .$$

Substituting Eqn. (3) yields

$$0 = b + \sum\nolimits_{j\in[d]} \mathbf{X}_{r,j}(\mathbf{z}_j - a) = b + (\mathbf{X}\mathbf{z})_r - a\sum\nolimits_{j\in[d]} \mathbf{X}_{r,j} .$$

Each row of $\mathbf{X}$ has sum 3, so $(\mathbf{X}\mathbf{z})_r = 3a - b$ for every $r \in [m]$. Hence $\mathbf{X}\mathbf{z} = (3a-b)\mathbf{1}_m$.

Conversely, suppose that $\mathbf{X}\mathbf{z} = \beta\mathbf{1}_m$ for some scalar $\beta \in \mathbb{R}$. Then

$$\sum\nolimits_{j\in[d]} \mathbf{z}_j \mathbf{1}_{R_j} - \beta\mathbf{1}_{\mathcal{X}_2} \in \textsc{Row}(\mathbf{M}) .$$

At the coordinate corresponding to v_j, this vector equals $\mathbf{z}_j$. At the coordinate corresponding to w_r, it equals $(\mathbf{X}\mathbf{z})_r - \beta = 0$. Thus the vector is exactly $\mathbf{1}_S$, so $\mathbf{1}_S \in \textsc{Row}(\mathbf{M})$. □

Lemma D.2. *If $S \subseteq \mathcal{X}_2$ and $\mathbf{1}_S \in \textsc{Row}(\mathbf{M})$, then either $S = \emptyset$ or $S = \mathcal{X}_2$.*

Proof. We write

$$\mathbf{1}_S = a\mathbf{1}_{\mathcal{X}_1} + b\mathbf{1}_{\mathcal{X}_2} + \sum\nolimits_{j\in[d]} \alpha_j \mathbf{1}_{R_j} \tag{4}$$

for some scalars $a, b, \alpha_1, \ldots, \alpha_d$. Evaluating Eqn. (4) at v_j yields $0 = a + \alpha_j$ for every j, because $S \subseteq \mathcal{X}_2$. Thus $\alpha_j = -a$ for every j. Evaluating Eqn. (4) at w_r then gives

$$(\mathbf{1}_S)_{w_r} = b + \sum\nolimits_{j\in[d]} \mathbf{X}_{r,j}\alpha_j = b - a\sum\nolimits_{j\in[d]} \mathbf{X}_{r,j} = b - 3a .$$

Thus, every coordinate of $\mathbf{1}_S$ on $\mathcal{X}_2$ has the same value. Since $\mathbf{1}_S$ is a $\{0,1\}$-vector, either all these coordinates are 0 or all are 1. Therefore $S = \emptyset$ or $S = \mathcal{X}_2$. □

Lemma D.3. *The following statements are equivalent:*

1. *the* X3C3 *instance admits an exact cover;*
2. *there exists a predictor $g \in \mathcal{G}^{\text{DET}}$ such that $g \succ f_1$;*
3. *there exists a predictor $g^\star \in \mathcal{G}^{\text{DET}}_{\succ}$ such that $g^\star \succ f_1$.*

Proof. We first prove the equivalence between (1) and (2).

Suppose that there exists a predictor $g \in \mathcal{G}^{\text{DET}}$ such that $g \succ f_1$. By the deterministic special case of dominance, the signal partition Π_g strictly refines $\Pi_{f_1} = \{\mathcal{X}_1, \mathcal{X}_2\}$. Therefore at least one of the two cells $\mathcal{X}_1$ or $\mathcal{X}_2$ contains a nonempty proper subset that is a signal cell of g. Because $g \in \mathcal{G}^{\text{DET}}$, every

signal cell of g is observable by Corollary 3.4. Thus at least one of $\mathcal{X}_1$ or $\mathcal{X}_2$ contains a nonempty proper observable subset.

Conversely, suppose that one of the two cells in Π_{f_1} contains a nonempty proper observable subset. That is, suppose there exist $A \in \{\mathcal{X}_1, \mathcal{X}_2\}$ and a nonempty proper subset $S \subsetneq A$ such that $\mathbf{1}_S \in \textsc{Row}(\mathbf{M})$. Since A is itself a signal cell of f_1, we also have $\mathbf{1}_A \in \textsc{Row}(\mathbf{M})$. Thus, we have

$$\mathbf{1}_{A \setminus S} = \mathbf{1}_A - \mathbf{1}_S \in \textsc{Row}(\mathbf{M}) \, .$$

Thus the partition obtained from $\{\mathcal{X}_1, \mathcal{X}_2\}$ by replacing A with the two cells S and $A \setminus S$ has all signal-cell indicators in $\textsc{Row}(\mathbf{M})$. By Corollary 3.4, this partition defines a predictor $g \in \mathcal{G}^{\text{DET}}$. Its signal partition strictly refines Π_{f_1}, so the deterministic special case of dominance implies $g \succ f_1$.

We have shown that statement (2) is equivalent to the existence of a nonempty proper observable subset inside $\mathcal{X}_1$ or $\mathcal{X}_2$. By Lemma D.2, the cell $\mathcal{X}_2$ has no nonempty proper observable subset. Therefore statement (2) holds if and only if there exists a nonempty proper subset $S \subsetneq \mathcal{X}_1$ such that $\mathbf{1}_S \in \textsc{Row}(\mathbf{M})$.

Now apply Lemma D.1. Statement (2) is equivalent to the existence of a vector $\mathbf{z} \in \{0,1\}^d$ that is neither $\mathbf{0}$ nor $\mathbf{1}_d$ and satisfies $\mathbf{X}\mathbf{z} = \beta \mathbf{1}_m$ for some scalar $\beta \in \mathbb{R}$. Because each row of $\mathbf{X}$ has exactly three 1's and $\mathbf{z} \in \{0,1\}^d$, every entry of $\mathbf{X}\mathbf{z}$ belongs to $\{0,1,2,3\}$. Thus, we have $\beta \in \{0,1,2,3\}$. We now analyze the following four cases.

- *Case* $\beta = 0$. Then $\mathbf{X}\mathbf{z} = \mathbf{0}$. Multiplying on the left by $\mathbf{1}_m^\top$ gives

  $$0 = \mathbf{1}_m^\top \mathbf{X}\mathbf{z} = 3\, \mathbf{1}_d^\top \mathbf{z} \, ,$$

  because every column of $\mathbf{X}$ has sum 3. Hence $\mathbf{1}_d^\top \mathbf{z} = 0$, so $\mathbf{z} = \mathbf{0}$, contradicting the fact that S is nonempty.
- *Case* $\beta = 3$. Then $\mathbf{X}\mathbf{z} = 3\mathbf{1}_m$. Multiplying on the left by $\mathbf{1}_m^\top$ gives

  $$3m = \mathbf{1}_m^\top \mathbf{X}\mathbf{z} = 3\, \mathbf{1}_d^\top \mathbf{z} \, .$$

  Since $d = m$, we obtain $\mathbf{1}_d^\top \mathbf{z} = d$. As $\mathbf{z} \in \{0,1\}^d$, this implies $\mathbf{z} = \mathbf{1}_d$, contradicting the fact that S is a proper subset of $\mathcal{X}_1$.
- *Case* $\beta = 1$. Then $\mathbf{X}\mathbf{z} = \mathbf{1}_m$. Equivalently, every element of $\mathcal{U}$ belongs to exactly one selected set. Hence the selected family $\{C_j : \mathbf{z}_j = 1\}$ is an exact cover of $\mathcal{U}$.
- *Case* $\beta = 2$. Then

  $$\mathbf{X}(\mathbf{1}_d - \mathbf{z}) = \mathbf{X}\mathbf{1}_d - \mathbf{X}\mathbf{z} = 3\mathbf{1}_m - 2\mathbf{1}_m = \mathbf{1}_m \, .$$

  Thus, the complementary family $\{C_j : \mathbf{z}_j = 0\}$ is an exact cover of $\mathcal{U}$.

We have thus shown that (1) and (2) are equivalent.

It remains to prove the equivalence between (2) and (3). The implication $(3) \Rightarrow (2)$ is immediate. For the converse, assume that there exists $g \in \mathcal{G}^{\text{DET}}$ such that $g \succ f_1$. Define

$$\mathcal{G}_g^{\text{DET}} \triangleq \{h \in \mathcal{G}^{\text{DET}} : \ \Pi_h \text{ refines } \Pi_g\} \, .$$

By Corollary 3.4, every predictor in $\mathcal{G}^{\text{DET}}$ is uniquely determined by its signal partition, and there are only finitely many set partitions of $[n]$. Hence $\mathcal{G}_g^{\text{DET}}$ is finite. Choose $g^\star \in \mathcal{G}_g^{\text{DET}}$ with the maximum possible number of signal cells.

We claim that $g^\star \in \mathcal{G}_\succ^{\text{DET}}$. If not, there exists some $h \in \mathcal{G}^{\text{DET}}$ such that $h \succ g^\star$. By the deterministic special case of dominance, Π_h strictly refines $\Pi_{g^\star}$. Since $\Pi_{g^\star}$ refines Π_g, it follows that Π_h also refines Π_g, so $h \in \mathcal{G}_g^{\text{DET}}$. But strict refinement increases the number of signal cells, contradicting the maximal choice of $g^\star$. Thus $g^\star \in \mathcal{G}_\succ^{\text{DET}}$.

Finally, since $\Pi_{g^\star}$ refines Π_g and Π_g strictly refines Π_{f_1}, the partition $\Pi_{g^\star}$ also strictly refines Π_{f_1}. Thus, again by the deterministic special case of dominance, we have $g^\star \succ f_1$. This proves $(2) \Rightarrow (3)$, and completes the proof. □

It remains to verify that the reduction has polynomial output size. The combinatorial part is immediate: $n = d + m = 2m$, $k = d + 1 = m + 1$, and every input predictor has exactly two signal

cells. It remains to bound the encoding length of the numerical predictions of the input predictors. Fix any initial signal cell A. Because $\lambda_i = 1/n$ for every i,

$$\mu(\mathbf{1}_A) = \frac{Y(\mathbf{1}_A)}{\lambda(\mathbf{1}_A)} = \frac{\sum_{i\in A}\lambda_i q_i}{\sum_{i\in A}\lambda_i} = \frac{1}{|A|}\sum\nolimits_{i\in A}(n+1)^{-i} = \frac{1}{|A|(n+1)^n}\sum\nolimits_{i\in A}(n+1)^{n-i}\,.$$

Thus $\mu(\mathbf{1}_A)$ is a rational number whose numerator and denominator are both at most $n(n+1)^n$. Thus, each such value has bit-length $O(n\log n)$. Therefore the whole constructed instance of IFREFINABLE has polynomial size.

By Lemma D.3, the X3C3 instance is a YES-instance if and only if the constructed instance of IFREFINABLE is a YES-instance. Therefore X3C3 reduces in polynomial time to IFREFINABLE. Since X3C3 is NP-complete, the problem IFREFINABLE is NP-hard. □

Proof of Corollary 4.3. Assume that there exists a polynomial-time algorithm for PREDICTORREFINE, restricted to deterministic inputs and deterministic outputs. Given an instance of IFREFINABLE, run this algorithm on the same input and inspect the first output predictor g_1. If $g_1 = f_1$, output NO; otherwise output YES.

We claim that this procedure is correct. If the answer to IFREFINABLE is YES, then by definition there exists some predictor $h \in \mathcal{G}_{\succ}^{\mathrm{DET}}$ such that $h \succ f_1$. By the specification of PREDICTORREFINE, the algorithm must therefore return a predictor $g_1 \in \mathcal{G}_{\succ}^{\mathrm{DET}}$ such that $g_1 \succ f_1$. In particular, $g_1 \neq f_1$.

Conversely, if the answer to IFREFINABLE is NO, then there exists no predictor $h \in \mathcal{G}_{\succ}^{\mathrm{DET}}$ such that $h \succ f_1$. Again by the specification of PREDICTORREFINE, the algorithm must return $g_1 = f_1$.

Thus a polynomial-time algorithm for the deterministic version of PREDICTORREFINE would yield a polynomial-time algorithm for IFREFINABLE. Since IFREFINABLE is NP-hard by Theorem 4.2, the deterministic version of PREDICTORREFINE is NP-hard under polynomial-time Turing reductions. □

E Missing Proofs in Section 4.2

Lemma E.1 (Branch decomposition of one input cell)**.** *Fix a cell $A \in \Pi_{f_1}$. For each connected component K of $G - A$, define the K-branch of the cell A as follows:*

$$A_K \triangleq A \cap \left(\bigcup\nolimits_{A'\in\mathcal{I}(K)\cap\Pi_{f_2}} A'\right)\,.$$

Then the following statements hold.

(i) The nonempty sets A_K form a partition of A.

(ii) For every connected component K of $G - A$,

$$\mathbf{1}_{A_K} = \sum\nolimits_{A'\in\mathcal{I}(K)\cap\Pi_{f_2}} \mathbf{1}_{A'} - \sum\nolimits_{A^\dagger\in\mathcal{I}(K)\cap\Pi_{f_1}} \mathbf{1}_{A^\dagger}\,. \tag{5}$$

Consequently, $\mathbf{1}_{A_K} \in \mathrm{ROW}(\mathbf{M})$, so A_K is observable by Corollary 3.3. Moreover, its observable cell label mass is given by

$$Y(\mathbf{1}_{A_K}) = \sum\nolimits_{A'\in\mathcal{I}(K)\cap\Pi_{f_2}} L(A') - \sum\nolimits_{A^\dagger\in\mathcal{I}(K)\cap\Pi_{f_1}} L(A^\dagger)\,. \tag{6}$$

(iii) If $S \subseteq A$ and $\mathbf{1}_S \in \mathrm{ROW}(\mathbf{M})$, then there exists a subcollection $\mathcal{T}$ of the connected components of $G - A$ such that $S = \bigcup_{K\in\mathcal{T}} A_K$.

The completely symmetric statements hold after interchanging the roles of Π_{f_1} and Π_{f_2}.

Proof of Lemma E.1. We prove the three statements in order.

Proof of statement (i). Fix a cell $A \in \Pi_{f_1}$. For each feature index $i \in A$, let $A_2(i)$ denote the unique cell of Π_{f_2} that contains i. Since $A_2(i)$ is adjacent to A in G, the vertex $A_2(i)$ belongs to a unique connected component K of $G - A$, and then $i \in A_K$. Thus the union of the nonempty sets A_K is all of A.

If $K_1 \neq K_2$ are two connected components of $G - A$, then $\mathcal{I}(K_1) \cap \Pi_{f_2}$ and $\mathcal{I}(K_2) \cap \Pi_{f_2}$ are disjoint. Since the cells of Π_{f_2} are pairwise disjoint, the sets A_{K_1} and A_{K_2} are disjoint. Hence the nonempty sets A_K form a partition of A.

Proof of statement (ii). Fix a connected component K of $G - A$. We verify Eqn. (5) coordinatewise. Fix a feature index $i \in [n]$, and let $A_1(i)$ and $A_2(i)$ be the unique cells of Π_{f_1} and Π_{f_2}, respectively, that contain i. There are four cases.

- If $A_1(i) = A$ and $A_2(i) \in \mathcal{I}(K)$, then the right-hand side of Eqn. (5) equals 1 at coordinate i, and by definition $i \in A_K$.
- If $A_1(i) = A$ and $A_2(i) \notin \mathcal{I}(K)$, then the right-hand side of Eqn. (5) equals 0 at coordinate i, and $i \notin A_K$.
- If $A_1(i) \neq A$ and $A_1(i) \in \mathcal{I}(K)$, then the edge $(A_1(i), A_2(i))$ lies in $G - A$. Since K is a connected component of $G - A$, this implies $A_2(i) \in \mathcal{I}(K)$ as well. Thus the right-hand side of Eqn. (5) equals $1 - 1 = 0$ at coordinate i. Also $i \notin A$, hence $i \notin A_K$.
- If $A_1(i) \neq A$ and $A_1(i) \notin \mathcal{I}(K)$, then again the edge $(A_1(i), A_2(i))$ lies in $G - A$, so $A_2(i) \notin \mathcal{I}(K)$. Hence the right-hand side of Eqn. (5) equals 0 at coordinate i, and $i \notin A_K$.

These cases exhaust all possibilities, so Eqn. (5) holds. Because the right-hand side is a linear combination of rows of $\mathbf{M}$, we obtain $\mathbf{1}_{A_K} \in \textsc{Row}(\mathbf{M})$. Applying the linear functional $Y(\cdot)$ to Eqn. (5) yields Eqn. (6).

Proof of statement (iii). Assume that $S \subseteq A$ and $\mathbf{1}_S \in \textsc{Row}(\mathbf{M})$. By Remark 3.5, the rows of $\mathbf{M}$ are exactly the vectors $\mathbf{1}_{A'}$ for $A' \in \Pi_{f_1} \cup \Pi_{f_2}$. Hence there exist coefficients $(\alpha_{A'})_{A' \in \Pi_{f_1}}$ and $(\beta_{A''})_{A'' \in \Pi_{f_2}}$ such that

$$\mathbf{1}_S = \sum\nolimits_{A' \in \Pi_{f_1}} \alpha_{A'} \mathbf{1}_{A'} + \sum\nolimits_{A'' \in \Pi_{f_2}} \beta_{A''} \mathbf{1}_{A''} \,. \tag{7}$$

Now let $(A^\dagger, A')$ be any edge of G with $A^\dagger \neq A$. Every feature in the nonempty intersection $A^\dagger \cap A'$ lies outside A, whereas $S \subseteq A$. Therefore the left-hand side of Eqn. (7) vanishes on $A^\dagger \cap A'$, and so

$$\alpha_{A^\dagger} + \beta_{A'} = 0 \tag{8}$$

for every such edge $(A^\dagger, A')$. Fix a connected component K of $G - A$.

- Suppose first that $\mathcal{I}(K) \cap \Pi_{f_1} \neq \emptyset$. Choose any cell $A_0^\dagger \in \mathcal{I}(K) \cap \Pi_{f_1}$ and set $c_K \triangleq \alpha_{A_0^\dagger}$. By Eqn. (8), every neighbor A' of $A_0^\dagger$ in K satisfies $\beta_{A'} = -c_K$. Applying Eqn. (8) repeatedly along paths in the connected component shows that every vertex of $\mathcal{I}(K) \cap \Pi_{f_1}$ has coefficient c_K and every vertex of $\mathcal{I}(K) \cap \Pi_{f_2}$ has coefficient $-c_K$. Therefore, for every $A' \in \mathcal{I}(K) \cap \Pi_{f_2}$, the value of $\mathbf{1}_S$ on the whole intersection $A \cap A'$ is

  $$\alpha_A + \beta_{A'} = \alpha_A - c_K,$$

  which depends only on K and not on the particular choice of A'. Hence $\mathbf{1}_S$ is constant on A_K.
- Suppose next that $\mathcal{I}(K) \cap \Pi_{f_1} = \emptyset$. Since $G - A$ is bipartite, every edge of $G - A$ joins a vertex in $\Pi_{f_1} \setminus \{A\}$ to a vertex in Π_{f_2}. Because K contains no vertex of Π_{f_1}, it contains no edge. As K is connected, it must therefore consist of a single isolated vertex $A_0 \in \Pi_{f_2}$. In this case, $A_K = A \cap A_0$. Fix any feature index $i \in A_K$. Since $i \in A$ and $i \in A_0$, while i belongs to no other cell of either partition, Eqn. (7) gives $(\mathbf{1}_S)_i = \alpha_A + \beta_{A_0}$. Thus, we have $\mathbf{1}_S$ is constant on A_K.

Thus, in either case, $\mathbf{1}_S$ is constant on each branch piece A_K. Since $\mathbf{1}_S$ is a $\{0,1\}$-vector, each A_K is either entirely contained in S or entirely disjoint from S. Therefore S is a union of some of the sets A_K. □

Algorithm 1: COMPUTEFINESTREFINEMENT(j) for $j \in \{1,2\}$

Input: Feature distribution $\lambda = (\lambda_i)_{i\in[n]}$ and deterministic predictors f_1, f_2.
Output: A constructible deterministic predictor $g_j^\star \in \mathcal{G}^{\mathrm{DET}}$.

Construct the signal partitions $\Pi_{f_1} = \{A_{1,\ell}\}_{\ell\in[m_1]}$ and $\Pi_{f_2} = \{A_{2,r}\}_{r\in[m_2]}$;
Construct the overlap graph G with vertex set $\Pi_{f_1} \sqcup \Pi_{f_2}$ and edge $(A_{1,\ell}, A_{2,r})$ iff $A_{1,\ell} \cap A_{2,r} \neq \emptyset$;
For every input signal cell $A \in \Pi_{f_1} \cup \Pi_{f_2}$, compute and store $L(A) \leftarrow \mu(\mathbf{1}_A)\,\lambda(\mathbf{1}_A)$;
Set $j^\dagger \leftarrow 3 - j$;
Initialize $\widehat{\Pi}_j \leftarrow \emptyset$;
foreach $A \in \Pi_{f_j}$ **do**
    Compute the connected components of $G - A$;
    foreach *connected component* K *of* $G - A$ **do**
        $C \leftarrow A \cap \left(\bigcup_{A^\dagger \in \mathcal{I}(K)\cap\Pi_{f_{j^\dagger}}} A^\dagger\right)$;
        if $C \neq \emptyset$ **then**
            compute $Y(\mathbf{1}_C) \leftarrow \sum_{A^\dagger \in \mathcal{I}(K)\cap\Pi_{f_{j^\dagger}}} L(A^\dagger) - \sum_{A^\ddagger \in \mathcal{I}(K)\cap\Pi_{f_j}} L(A^\ddagger)$;
            assign the prediction $\mu(\mathbf{1}_C) \leftarrow \frac{Y(\mathbf{1}_C)}{\lambda(\mathbf{1}_C)}$;
            add the cell C to $\widehat{\Pi}_j$;
return the deterministic predictor $g_j^\star$ whose signal partition is $\widehat{\Pi}_j$ and whose prediction on each signal cell $C \in \widehat{\Pi}_j$ is $\mu(\mathbf{1}_C)$;

Algorithm 2: Polynomial-time algorithm for the deterministic version of PREDICTORREFINE when $k = 2$

Input: Feature distribution $\lambda = (\lambda_i)_{i\in[n]}$ and deterministic predictors f_1, f_2.
Output: Deterministic predictors (g_1, g_2).

$g_1^\star \leftarrow$ COMPUTEFINESTREFINEMENT(1);
$g_2^\star \leftarrow$ COMPUTEFINESTREFINEMENT(2);
for $j \in \{1,2\}$ **do**
    if $\Pi_{g_j^\star} = \Pi_{f_j}$ **then**
        $g_j \leftarrow f_j$;
    else
        $g_j \leftarrow g_j^\star$;
return (g_1, g_2);

Proof of Theorem 4.4. We prove the statement for $j = 1$; the proof for $j = 2$ is completely symmetric.

For each cell $A \in \Pi_{f_1}$ and each connected component K of $G - A$, define A_K as in Lemma E.1, and set

$$\widehat{\Pi}_1 \triangleq \{A_K \neq \emptyset :\ A \in \Pi_{f_1},\ K \text{ is a connected component of } G - A\}\,.$$

For each fixed $A \in \Pi_{f_1}$, Lemma E.1(i) shows that the nonempty sets A_K partition A. Since the cells of Π_{f_1} partition $[n]$, it follows that $\widehat{\Pi}_1$ is a partition of $[n]$. By Lemma E.1(ii), every cell $A' \in \widehat{\Pi}_1$ satisfies $\mathbf{1}_{A'} \in \mathrm{ROW}(\mathbf{M})$, and its quantity $Y(\mathbf{1}_{A'})$ is computable from the observed data. Therefore, by Corollary 3.4, the predictor $g_1^\star$ returned by Algorithm 1 belongs to $\mathcal{G}^{\mathrm{DET}}$ and has signal partition $\Pi_{g_1^\star} = \widehat{\Pi}_1$.

We next show that $\widehat{\Pi}_1$ is the finest constructible deterministic refinement of Π_{f_1}. Let $g \in \mathcal{G}^{\mathrm{DET}}$ be any deterministic predictor such that Π_g refines Π_{f_1}. Fix any cell $A' \in \Pi_g$. Since Π_g refines Π_{f_1}, there exists a unique cell $A \in \Pi_{f_1}$ such that $A' \subseteq A$. Because $g \in \mathcal{G}^{\mathrm{DET}}$, Corollary 3.4 gives $\mathbf{1}_{A'} \in \mathrm{ROW}(\mathbf{M})$. Applying Lemma E.1(iii) with $S = A'$ shows that A' is a union of cells of $\widehat{\Pi}_1$.

Since this holds for every $A' \in \Pi_g$, we conclude that

$$\text{if } \Pi_g \text{ refines } \Pi_{f_1}, \text{ then } \widehat{\Pi}_1 \text{ refines } \Pi_g. \tag{9}$$

We now verify the correctness of Algorithm 2 for the first input predictor.

- *Case 1:* $\widehat{\Pi}_1 \neq \Pi_{f_1}$. Then $\Pi_{g_1^\star} = \widehat{\Pi}_1$ strictly refines Π_{f_1}. By Lemma 4.1, this implies $g_1^\star \succ f_1$. We claim that $g_1^\star \in \mathcal{G}_{\succ}^{\text{DET}}$. Suppose not. Then there exists some predictor $g \in \mathcal{G}^{\text{DET}}$ such that $g \succ g_1^\star$. By Lemma 4.1, the partition Π_g strictly refines $\Pi_{g_1^\star} = \widehat{\Pi}_1$. In particular, Π_g refines Π_{f_1}. Applying Eqn. (9) to g yields that $\widehat{\Pi}_1$ refines Π_g, contradicting the strict refinement of $\widehat{\Pi}_1$ by Π_g. Hence $g_1^\star \in \mathcal{G}_{\succ}^{\text{DET}}$.
  Therefore, in this case, Algorithm 2 outputs $g_1 = g_1^\star$, and this predictor satisfies $g_1 \in \mathcal{G}_{\succ}^{\text{DET}}$ and $g_1 \succ f_1$.
- *Case 2:* $\widehat{\Pi}_1 = \Pi_{f_1}$. We claim that no predictor in $\mathcal{G}^{\text{DET}}$ strictly informationally-dominates f_1. Suppose, toward a contradiction, that there exists $g \in \mathcal{G}^{\text{DET}}$ such that $g \succ f_1$. By Lemma 4.1, the partition Π_g strictly refines Π_{f_1}. Applying Eqn. (9) to g gives that $\widehat{\Pi}_1$ refines Π_g. Since $\widehat{\Pi}_1 = \Pi_{f_1}$, this implies that Π_{f_1} refines Π_g, contradicting the strict refinement of Π_{f_1} by Π_g. Thus there exists no $g \in \mathcal{G}^{\text{DET}}$ with $g \succ f_1$; in particular, there exists no $g \in \mathcal{G}_{\succ}^{\text{DET}}$ with $g \succ f_1$. Hence Algorithm 2 correctly outputs $g_1 = f_1$ in this case.

Combining the two cases, the output for the first input predictor satisfies exactly the required specification of PREDICTORREFINE. By symmetry, the same conclusion holds for the second input predictor. Therefore Algorithm 2 solves the deterministic version of PREDICTORREFINE when $k = 2$.

It remains to prove the running-time bound. Let $I \triangleq |\mathcal{I}(G)| = m_1 + m_2$ and let E denote the number of edges of G. Because every feature index belongs to exactly one cell of Π_{f_1} and exactly one cell of Π_{f_2}, each feature contributes to at most one edge of G. Hence $E \leq n$. Also $m_1, m_2 \leq n$, so $I \leq 2n$.

The overlap graph and the quantities $L(A)$ for the input cells can be computed in polynomial time from the input. In one call to Algorithm 1, we compute connected components of $G - A$ once for each of the at most I input cells A; each such computation takes time $O(I+E)$. After the component labels are known, the branch pieces for this fixed A can be formed by scanning the feature indices in A and grouping them according to the connected component containing their opposite-side cell; this takes time $O(n)$. Thus one call to Algorithm 1 runs in time $O(I(I + E + n))$ which is polynomial in the input size because $I, E \leq O(n)$.

Finally, all arithmetic used to compute the quantities $L(A)$, $Y(\mathbf{1}_{A'})$, and $\mu(\mathbf{1}_{A'})$ involves only polynomially many additions, subtractions, multiplications, and divisions of rational numbers whose bit-length is polynomial in the input size. Hence the total bit complexity is polynomial as well. The second call to Algorithm 1 has the same complexity. Therefore Algorithm 2 runs in polynomial time. □

F Missing Proofs in Section 5

Definition F.1 (Extreme ray)**.** *Let* $\mathbf{v} \in \mathcal{M} \setminus \{\mathbf{0}\}$. *We write* $\text{ray}(\mathbf{v}) \triangleq \{\beta\,\mathbf{v} : \beta \geq 0\}$. *The ray* $\text{ray}(\mathbf{v})$ *is called an extreme ray of* $\mathcal{M}$ *if whenever* $\mathbf{v} = \mathbf{v}^\dagger + \mathbf{v}^\ddagger$ *with* $\mathbf{v}^\dagger, \mathbf{v}^\ddagger \in \mathcal{M}$, *then* $\mathbf{v}^\dagger, \mathbf{v}^\ddagger \in \text{ray}(\mathbf{v})$.

Lemma F.1 (Support-minimal vectors span exactly the extreme rays of $\mathcal{M}$)**.** *Let* $\mathbf{v} \in \mathcal{M} \setminus \{\mathbf{0}\}$. *Then the following are equivalent:*

(i) the ray $\text{ray}(\mathbf{v})$ *is extreme in* $\mathcal{M}$*;*

(ii) there exists no $\mathbf{u} \in \mathcal{M} \setminus \{\mathbf{0}\}$ *such that* $\mathsf{supp}(\mathbf{u}) \subsetneq \mathsf{supp}(\mathbf{v})$.

Proof. We first prove $(i) \Rightarrow (ii)$. Assume that $\text{ray}(\mathbf{v})$ is extreme, and suppose toward a contradiction that there exists $\mathbf{u} \in \mathcal{M} \setminus \{\mathbf{0}\}$ such that $\mathsf{supp}(\mathbf{u}) \subsetneq \mathsf{supp}(\mathbf{v})$. Define

$$\beta \triangleq \min_{i \in \mathsf{supp}(\mathbf{u})} \frac{\mathbf{v}_i}{\mathbf{u}_i}\,.$$

This scalar is well-defined and strictly positive because $\mathsf{supp}(\mathbf{u}) \subseteq \mathsf{supp}(\mathbf{v})$ and $\mathbf{u}_i > 0$ for every $i \in \mathsf{supp}(\mathbf{u})$.

By the definition of β, we have $\mathbf{v} - \beta\mathbf{u} \in \mathbb{R}^n_+$. Moreover, since both $\mathbf{v}$ and $\mathbf{u}$ belong to $\textsc{Row}(\mathbf{M})$, we also have

$$\mathbf{v} - \beta\mathbf{u} \in \textsc{Row}(\mathbf{M}) \,.$$

Thus, we have $\mathbf{v} - \beta\mathbf{u} \in \mathcal{M}$. Furthermore, $\mathbf{v} - \beta\mathbf{u} \neq \mathbf{0}$, because every coordinate in $\mathsf{supp}(\mathbf{v}) \setminus \mathsf{supp}(\mathbf{u})$ remains strictly positive. Thus,

$$\mathbf{v} = \beta\mathbf{u} + (\mathbf{v} - \beta\mathbf{u})$$

is a decomposition of $\mathbf{v}$ into two nonzero vectors in $\mathcal{M}$. However, the first summand $\beta\mathbf{u}$ does not belong to $\mathrm{ray}(\mathbf{v})$: it vanishes on $\mathsf{supp}(\mathbf{v}) \setminus \mathsf{supp}(\mathbf{u})$, whereas $\mathbf{v}$ is strictly positive on that set. This contradicts the extremality of $\mathrm{ray}(\mathbf{v})$.

We next prove $(ii) \Rightarrow (i)$. Assume (ii), and consider any decomposition

$$\mathbf{v} = \mathbf{v}^\dagger + \mathbf{v}^\ddagger \quad \text{with } \mathbf{v}^\dagger, \mathbf{v}^\ddagger \in \mathcal{M} \,.$$

If either $\mathbf{v}^\dagger = \mathbf{0}$ or $\mathbf{v}^\ddagger = \mathbf{0}$, then both summands belong to $\mathrm{ray}(\mathbf{v})$, so there is nothing to prove. Hence assume $\mathbf{v}^\dagger, \mathbf{v}^\ddagger \neq \mathbf{0}$.

Since $\mathbf{v}^\dagger, \mathbf{v}^\ddagger \geq \mathbf{0}$ and $\mathbf{v} = \mathbf{v}^\dagger + \mathbf{v}^\ddagger$, we have $\mathsf{supp}(\mathbf{v}^\dagger) \subseteq \mathsf{supp}(\mathbf{v}), \mathsf{supp}(\mathbf{v}^\ddagger) \subseteq \mathsf{supp}(\mathbf{v})$. By condition (ii), neither inclusion can be strict. Thus, we have $\mathsf{supp}(\mathbf{v}^\dagger) = \mathsf{supp}(\mathbf{v})$ and $\mathsf{supp}(\mathbf{v}^\ddagger) = \mathsf{supp}(\mathbf{v})$.

We show that $\mathbf{v}^\dagger \in \mathrm{ray}(\mathbf{v})$. Suppose, toward a contradiction, that $\mathbf{v}^\dagger \notin \mathrm{ray}(\mathbf{v})$. Since both summands have the same support as $\mathbf{v}$, for every $i \in \mathsf{supp}(\mathbf{v})$ we have $0 < \mathbf{v}^\dagger_i < \mathbf{v}_i$. Define

$$\beta^\dagger \triangleq \min_{i \in \mathsf{supp}(\mathbf{v})} \frac{\mathbf{v}^\dagger_i}{\mathbf{v}_i} \in (0,1) \,, \quad \mathbf{z} \triangleq \mathbf{v}^\dagger - \beta^\dagger \mathbf{v} \,.$$

By construction, $\mathbf{z} \geq \mathbf{0}$. Since $\mathbf{v}^\dagger, \mathbf{v} \in \textsc{Row}(\mathbf{M})$, we also have $\mathbf{z} \in \textsc{Row}(\mathbf{M})$, and therefore $\mathbf{z} \in \mathcal{M}$. Moreover, $\mathbf{z} \neq \mathbf{0}$, because otherwise $\mathbf{v}^\dagger = \beta^\dagger \mathbf{v}$, contradicting $\mathbf{v}^\dagger \notin \mathrm{ray}(\mathbf{v})$. Finally, by the definition of $\beta^\dagger$, at least one coordinate of $\mathbf{z}$ is zero, while $\mathbf{z}$ is supported inside $\mathsf{supp}(\mathbf{v})$. Thus, $\mathsf{supp}(\mathbf{z}) \subsetneq \mathsf{supp}(\mathbf{v})$. This contradicts condition (ii). Therefore $\mathbf{v}^\dagger \in \mathrm{ray}(\mathbf{v})$.

The same argument applies to $\mathbf{v}^\ddagger$, so $\mathbf{v}^\ddagger \in \mathrm{ray}(\mathbf{v})$ as well. Thus, $\mathrm{ray}(\mathbf{v})$ is an extreme ray of $\mathcal{M}$. □

Lemma F.1 reduces the problem of finding extreme rays of the constructible signal cone $\mathcal{M}$ to a support-minimality problem. Indeed, by the lemma, a nonzero vector $\mathbf{v} \in \mathcal{M}$ spans an extreme ray if and only if there is no other nonzero vector in $\mathcal{M}$ whose support is a strict subset of $\mathsf{supp}(\mathbf{v})$. This suggests a natural algorithmic approach: rather than searching for extreme rays directly, we may start from a candidate support set $S \subseteq [n]$ and ask whether there exists any nonzero constructible signal vector supported inside S. If such a feasibility question can be answered efficiently, then we can iteratively shrink supports until no further reduction is possible, at which point Lemma F.1 guarantees that the resulting vector lies on an extreme ray. This is precisely the role of the following linear-feasibility program $\textsc{Feasible}(S)$: For every nonempty set $S \subseteq [n]$, consider the LP-feasibility problem

$$\begin{aligned} \text{find} \quad & (\xi, \mathbf{v}) \in \mathbb{R}^N \times \mathbb{R}^n_+ \\ \text{s.t.} \quad & \mathbf{v} = \mathbf{M}^\top \xi \,, \\ & \mathbf{v}_i = 0 \,, \qquad\qquad i \notin S \,, \\ & \textstyle\sum_{i \in S} \mathbf{v}_i = 1 \,. \end{aligned} \tag{$\textsc{Feasible}(S)$}$$

This system is feasible if and only if there exists a vector in $\mathcal{M} \setminus \{\mathbf{0}\}$ whose support is contained in S.

Algorithm 3: FINDEXTREMESIGNAL(S)

Input: A nonempty set $S \subseteq [n]$ such that FEASIBLE(S) is feasible.
Output: A vector $\mathbf{v}^\star \in \mathcal{M} \setminus \{\mathbf{0}\}$ whose ray is extreme.

$S_{\text{cur}} \leftarrow S$;
foreach $i \in S$ **do**
  if $i \in S_{\text{cur}}$ *and* FEASIBLE($S_{\text{cur}} \setminus \{i\}$) *is feasible* **then**
    $S_{\text{cur}} \leftarrow S_{\text{cur}} \setminus \{i\}$;
Solve FEASIBLE(S_{cur}) and return any feasible vector $\mathbf{v}^\star$;

Lemma F.2. *Algorithm 3 is well defined. If program* FEASIBLE(S) *is feasible, then the returned vector* $\mathbf{v}^\star$ *belongs to* $\mathcal{M} \setminus \{\mathbf{0}\}$ *and the ray* $\text{ray}(\mathbf{v}^\star)$ *is extreme in* $\mathcal{M}$.

Proof. Because the loop only deletes one index at a time, it terminates after at most $|S|$ iterations. By construction, program FEASIBLE(S_{cur}) is feasible at the moment the algorithm stops, so the returned vector $\mathbf{v}^\star$ belongs to $\mathcal{M} \setminus \{\mathbf{0}\}$.

We claim that $\mathsf{supp}(\mathbf{v}^\star) = S_{\text{cur}}$. Indeed, $\mathsf{supp}(\mathbf{v}^\star) \subseteq S_{\text{cur}}$ by the constraints of FEASIBLE(S_{cur}). If the inclusion were strict, then program FEASIBLE($\mathsf{supp}(\mathbf{v}^\star)$) would also be feasible, because the same vector $\mathbf{v}^\star$ would satisfy all the constraints after replacing S_{cur} by $\mathsf{supp}(\mathbf{v}^\star)$. Choose any $i \in S_{\text{cur}} \setminus \mathsf{supp}(\mathbf{v}^\star)$. Then $\mathsf{supp}(\mathbf{v}^\star) \subseteq S_{\text{cur}} \setminus \{i\}$, so FEASIBLE($S_{\text{cur}} \setminus \{i\}$) would be feasible, contradicting the stopping rule. Thus, $\mathsf{supp}(\mathbf{v}^\star) = S_{\text{cur}}$.

We next claim that there is no nonzero vector of $\mathcal{M}$ supported on a strict subset of S_{cur}. If such a vector existed, then after normalizing it to have coordinate sum 1 on its support, program FEASIBLE($S^\dagger$) would be feasible for some strict subset $S^\dagger \subsetneq S_{\text{cur}}$. Choose any $i \in S_{\text{cur}} \setminus S^\dagger$. Then FEASIBLE($S_{\text{cur}} \setminus \{i\}$) would be feasible, again contradicting the stopping rule. Thus, signal vector $\mathbf{v}^\star$ satisfies condition (ii) of Lemma F.1, and hence its ray is extreme by that lemma. □

Algorithm 4: EXTREMERAYDECOMPOSE($\mathbf{v}$)

Input: A vector $\mathbf{v} \in \mathcal{M} \setminus \{\mathbf{0}\}$.
Output: A finite bag $\mathcal{B}(\mathbf{v})$ of vectors in $\mathcal{M} \setminus \{\mathbf{0}\}$ whose rays are extreme and whose sum is $\mathbf{v}$.

$\mathbf{r} \leftarrow \mathbf{v}$;
Initialize $\mathcal{B}(\mathbf{v}) \leftarrow \emptyset$, $t \leftarrow 1$;
while $\mathbf{r} \neq \mathbf{0}$ **do**
  $S \leftarrow \mathsf{supp}(\mathbf{r})$;
  $\mathbf{u} \leftarrow$ FINDEXTREMESIGNAL(S);
  $\beta \leftarrow \min_{i \in \mathsf{supp}(\mathbf{u})} \mathbf{r}_i / \mathbf{u}_i$;
  $\mathbf{v}^{(t)} \leftarrow \beta \mathbf{u}$;
  add $\mathbf{v}^{(t)}$ to $\mathcal{B}(\mathbf{v})$;
  $\mathbf{r} \leftarrow \mathbf{r} - \mathbf{v}^{(t)}$;
  $t \leftarrow t + 1$
return $\mathcal{B}(\mathbf{v})$;

Lemma F.3. *For every* $\mathbf{v} \in \mathcal{M} \setminus \{\mathbf{0}\}$, *Algorithm 4 terminates after at most* n *iterations and returns a finite bag* $\mathcal{B}(\mathbf{v})$ *consisting of vectors in* $\mathcal{M} \setminus \{\mathbf{0}\}$, *each spanning an extreme ray of* $\mathcal{M}$, *such that*

$$\sum\nolimits_{\mathbf{v}^\dagger \in \mathcal{B}(\mathbf{v})} \mathbf{v}^\dagger = \mathbf{v}.$$

Proof. Fix one iteration and let $S = \mathsf{supp}(\mathbf{r})$. Because $\mathbf{r} \in \mathcal{M} \setminus \{\mathbf{0}\}$, the LP FEASIBLE($S$) is feasible after scaling $\mathbf{r}$ so that its coordinates on S sum to 1. Thus Algorithm 3 is well defined and returns $\mathbf{u} \in \mathcal{M} \setminus \{\mathbf{0}\}$ spanning an extreme ray of $\mathcal{M}$. Since $\mathsf{supp}(\mathbf{u}) \subseteq S = \mathsf{supp}(\mathbf{r})$, the scalar

$$\beta \triangleq \min_{i \in \mathsf{supp}(\mathbf{u})} \frac{\mathbf{r}_i}{\mathbf{u}_i}$$

is well defined and strictly positive. Therefore $\mathbf{v}^\dagger = \beta \mathbf{u}$ belongs to $\mathcal{M} \setminus \{\mathbf{0}\}$ and spans the same extreme ray as $\mathbf{u}$. Moreover, $\mathbf{r} - \mathbf{v}^\dagger \in \text{ROW}(\mathbf{M})$ and $\mathbf{r} - \mathbf{v}^\dagger \geq \mathbf{0}$, so $\mathbf{r} - \mathbf{v}^\dagger \in \mathcal{M}$. By the definition

of β, at least one coordinate of $\mathbf{r}-\mathbf{v}^\dagger$ is zero on $\mathsf{supp}(\mathbf{u})$. Thus, the support strictly shrinks whenever $\mathbf{r} \neq \mathbf{0}$. Therefore the loop terminates after at most n iterations.

If the output bag is $\mathcal{B}(\mathbf{v}) = \{\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(q)}\}$ in the order they are generated, then the residual vectors satisfy

$$\mathbf{r}^{(1)} = \mathbf{v}\,; \quad \mathbf{r}^{(t+1)} = \mathbf{r}^{(t)} - \mathbf{v}^{(t)} \;\text{ for } t \in [q]\,; \quad \mathbf{r}^{(q+1)} = \mathbf{0}\,.$$

Summing these identities yields $\mathbf{v} = \sum_{t\in[q]} \mathbf{v}^{(t)}$, as claimed. □

The previous lemmas show that every constructible signal vector in the cone $\mathcal{M}$ admits a finite decomposition into extreme-ray pieces. We next connect this geometric decomposition to informational dominance. For this purpose, it is convenient to use the non-strict garbling preorder.

For two finitely supported predictors f and g, we write $f \succeq g$ if there exists a feature-independent stochastic kernel $H : \Sigma_f \to \Delta(\Sigma_g)$ such that, for every feature $x \in \mathcal{X}$, the conditional signal law of g is obtained from that of f by passing through H. By Lemma 3.1, this is equivalent to the existence of the corresponding row-stochastic matrix in bag form. Moreover,

$$f \succ g \quad \Longleftrightarrow \quad f \succeq g \text{ and } g \not\succeq f$$

The preorder $\succeq$ is transitive: if $g_1 \succeq g_2$ via a feature-independent garbling H_{12} and $g_2 \succeq g_3$ via a feature-independent garbling H_{23}, then the composed channel $H_{23} \circ H_{12}$ is again feature-independent and maps the signal law of g_1 directly to that of g_3. We now show that once every row of a reduced bag lies on an extreme ray of $\mathcal{M}$, the resulting constructible predictor is already informationally-undominated.

Lemma F.4 (Extreme-ray bags are informationally-undominated)**.** *Let $g \in \mathcal{G}$ be a constructible predictor, and write its reduced bag as $\mathcal{V}(g) = \{\mathbf{v}_{g,r}\}_{r\in[m_g]}$. Assume that every vector $\mathbf{v}_{g,r}$ spans an extreme ray of $\mathcal{M}$. Then $g \in \mathcal{G}_\succ$. More precisely, for every predictor $h \in \mathcal{G}$, if we have $h \succeq g$, then we have $\mathcal{V}(h) = \mathcal{V}(g)$ as reduced bags, up to reordering.*

Proof of Lemma F.4. Fix a predictor h, let $\mathcal{V}(h) = \{\mathbf{v}_{h,\ell}\}_{\ell\in[m_h]}$ be its reduced bag-of-vectors. Assume that predictor $h \succeq g$. By Lemma 3.1, there exists a row-stochastic matrix $\mathbf{H} = (\mathbf{H}_{\ell,r})_{\ell\in[m_h],\, r\in[m_g]}$ such that signal vector satisfies

$$\mathbf{v}_{g,r} = \sum\nolimits_{\ell\in[m_h]} \mathbf{H}_{\ell,r}\, \mathbf{v}_{h,\ell} \quad \text{for every } r \in [m_g]\,. \tag{10}$$

Fix a target signal index $r \in [m_g]$. Because signal vector $\mathbf{v}_{g,r}$ spans an extreme ray of constructible signal cone $\mathcal{M}$ and each signal vector $\mathbf{v}_{h,\ell}$ belongs to $\mathcal{M}$, every source signal index ℓ with $\mathbf{H}_{\ell,r} > 0$ must satisfy $\mathbf{v}_{h,\ell} \in \mathrm{ray}(\mathbf{v}_{g,r})$. Since $\mathcal{V}(h)$ is reduced, there is at most one such ℓ. Denote it by ℓ_r if it exists. Then Eqn. (10) reduces to

$$\mathbf{v}_{g,r} = \mathbf{H}_{\ell_r,r}\, \mathbf{v}_{h,\ell_r}\,.$$

Thus, the vector $\mathbf{v}_{h,\ell_r}$ lies on the same ray as $\mathbf{v}_{g,r}$.

We claim that row ℓ_r of $\mathbf{H}$ has no positive entry in any other column $r' \neq r$. If $\mathbf{H}_{\ell_r,r'} > 0$ for some $r' \neq r$, then the same argument applied to column r' shows that $\mathbf{v}_{h,\ell_r}$ lies on the same ray as $\mathbf{v}_{g,r'}$. Therefore $\mathbf{v}_{g,r}$ and $\mathbf{v}_{g,r'}$ would lie on the same positive ray, contradicting the fact that $\mathcal{V}(g)$ is reduced. Thus the only positive entry in row ℓ_r is the one in column r. Since every row of $\mathbf{H}$ sums to 1, it follows that $\mathbf{H}_{\ell_r,r} = 1$, and thus, we have $\mathbf{v}_{h,\ell_r} = \mathbf{v}_{g,r}$.

We have shown that every vector in $\mathcal{V}(g)$ appears as a distinct row of $\mathcal{V}(h)$. If $\mathcal{V}(h)$ contained any additional row $\mathbf{v}_{h,\ell}$ not equal to one of these rows, then because row ℓ of $\mathbf{H}$ has sum 1, it would have to satisfy $\mathbf{H}_{\ell,r} > 0$ for some r. By the first paragraph of the proof, $\mathbf{v}_{h,\ell}$ would then lie on the same ray as $\mathbf{v}_{g,r}$, contradicting the reducedness of $\mathcal{V}(h)$. Thus $\mathcal{V}(h) = \mathcal{V}(g)$ up to reordering.

In particular, if some predictor $h \in \mathcal{G}$ satisfied $h \succ g$, then $h \succeq g$ would imply $\mathcal{V}(h) = \mathcal{V}(g)$. Applying Lemma 3.1 again, this yields $g \succeq h$, contradicting $h \succ g$. Thus, no constructible predictor strictly dominates g, so $g \in \mathcal{G}_\succ$. □

Algorithm 5: Solving PREDICTORREFINE for randomized predictors

Input: Feature distribution $\lambda = (\lambda_i)_{i\in[n]}$ and finitely supported predictors $(f_j)_{j\in[k]}$.
Output: Predictors $(g_j)_{j\in[k]}$.

Compute the reduced bags $\mathcal{V}(f_j) = \{\mathbf{v}_{j,\ell}\}_{\ell\in[m_j]}$ for all $j \in [k]$;
Construct the signal-feature matrix $\mathbf{M}$ and the observable vector $\mathbf{b}$ as in Definition 3.4;
for $j \in [k]$ **do**
    initialize an empty bag $\widetilde{\mathcal{V}}_j$;
    foreach $\ell \in [m_j]$ **do**
        run EXTREMERAYDECOMPOSE($\mathbf{v}_{j,\ell}$) and add all returned vectors to $\widetilde{\mathcal{V}}_j$;
    let $\mathcal{V}_j^\star$ be the ray-reduction of $\widetilde{\mathcal{V}}_j$;
    foreach $\mathbf{v} \in \mathcal{V}_j^\star$ **do**
        solve $\mathbf{M}^\top \xi = \mathbf{v}$;
        compute $p(\mathbf{v}) \leftarrow \dfrac{\xi^\top \mathbf{b}}{\lambda(\mathbf{v})}$;
    let $g_j^\star$ be the predictor whose reduced bag is $\mathcal{V}_j^\star$ and whose prediction on each signal $\mathbf{v} \in \mathcal{V}_j^\star$ is $p(\mathbf{v})$;
    if $\mathcal{V}_j^\star = \mathcal{V}(f_j)$ **then**
        $g_j \leftarrow f_j$;
    else
        $g_j \leftarrow g_j^\star$;
return $(g_j)_{j\in[k]}$;

We are now ready to prove Theorem 5.1.

Proof of Theorem 5.1. We first verify that every output constructed by the algorithm is well defined. Fix $j \in [k]$ and fix one input row $\mathbf{v}_{j,\ell} \in \mathcal{V}(f_j)$. Since predictor f_j itself is constructible from the observable information of the given instance, Definition 3.7 implies that $\mathbf{v}_{j,\ell} \in \mathcal{M} \setminus \{\mathbf{0}\}$. Thus, Algorithm 4 applies and returns a finite bag $\mathcal{B}(\mathbf{v}_{j,\ell})$ of vectors in $\mathcal{M} \setminus \{\mathbf{0}\}$ whose sum is $\mathbf{v}_{j,\ell}$, by Lemma F.3. Since $\lambda_i > 0$ for every i and every output vector is nonzero and nonnegative, we have $\lambda(\mathbf{v}) > 0$ for every $\mathbf{v} \in \widetilde{\mathcal{V}}_j$ and for every $\mathbf{v} \in \mathcal{V}_j^\star$. Thus every prediction $p(\mathbf{v})$ is well defined.

We next prove that $g_j^\star$ is a constructible predictor. Every vector in $\widetilde{\mathcal{V}}_j$ belongs to $\mathcal{M} \setminus \{\mathbf{0}\}$, hence to $\textsc{Row}(\mathbf{M}) \cap \mathbb{R}_+^n \setminus \{\mathbf{0}\}$. By construction, we have

$$\sum\nolimits_{\mathbf{v}\in\widetilde{\mathcal{V}}_j} \mathbf{v} = \sum\nolimits_{\ell\in[m_j]} \mathbf{v}_{j,\ell} = \mathbf{1}\,.$$

The ray-reduction $\mathcal{V}_j^\star$ is obtained by summing vectors inside each ray class, so every vector in $\mathcal{V}_j^\star$ still belongs to $\textsc{Row}(\mathbf{M}) \cap \mathbb{R}_+^n \setminus \{\mathbf{0}\}$, and the vectors in $\mathcal{V}_j^\star$ still sum coordinatewise to $\mathbf{1}$. Thus, Definition 3.7 implies that $g_j^\star \in \mathcal{G}$.

We now verify the prediction formula used in the algorithm. Fix $\mathbf{v} \in \mathcal{V}_j^\star$. Because $\mathbf{v} \in \textsc{Row}(\mathbf{M})$, the linear system $\mathbf{M}^\top \xi = \mathbf{v}$ has a solution. If ξ and ξ' are two such solutions, then $\mathbf{M}^\top(\xi - \xi') = \mathbf{0}$, so

$$(\xi - \xi')^\top \mathbf{b} = (\xi - \xi')^\top \mathbf{M}\mathbf{y} = \big(\mathbf{M}^\top(\xi - \xi')\big)^\top \mathbf{y} = 0\,.$$

Thus $\xi^\top \mathbf{b}$ depends only on $\mathbf{v}$ and not on the chosen solution ξ. Moreover,

$$\xi^\top \mathbf{b} = \xi^\top \mathbf{M}\mathbf{y} = (\mathbf{M}^\top \xi)^\top \mathbf{y} = \mathbf{v}^\top \mathbf{y} = Y(\mathbf{v})\,.$$

Thus, the algorithm indeed assigns the calibrated value

$$p(\mathbf{v}) = \frac{\xi^\top \mathbf{b}}{\lambda(\mathbf{v})} = \frac{Y(\mathbf{v})}{\lambda(\mathbf{v})} = \mu(\mathbf{v})\,.$$

We next show that every vector in $\mathcal{V}_j^\star$ spans an extreme ray of $\mathcal{M}$. Indeed, every vector in $\widetilde{\mathcal{V}}_j$ does so by Lemma F.3, and ray-reduction only merges positive multiples lying on the same ray.

Therefore Lemma F.4 gives

$$g_j^\star \in \mathcal{G}_\succ \,. \tag{11}$$

We now prove that $g_j^\star \succeq f_j$. Let $\widetilde{g}_j$ be the predictor represented by the unreduced bag $\widetilde{\mathcal{V}}_j$. For every piece $\mathbf{v} \in \mathcal{B}(\mathbf{v}_{j,\ell})$, define a deterministic garbling that maps the signal $\mathbf{v}$ of $\widetilde{g}_j$ to the parent signal $\mathbf{v}_{j,\ell}$ of f_j. Because $\sum_{\mathbf{v}\in\mathcal{B}(\mathbf{v}_{j,\ell})} \mathbf{v} = \mathbf{v}_{j,\ell}$, this is a valid feature-independent garbling from $\widetilde{g}_j$ to f_j. Thus, we have $\widetilde{g}_j \succeq f_j$. By the ray-reduction equivalence proved in Lemma F.5 in Section F, passing from the unreduced bag $\widetilde{\mathcal{V}}_j$ to its ray-reduction $\mathcal{V}_j^\star$ does not weaken informational-dominance content. Thus, we have $g_j^\star \succeq \widetilde{g}_j$. By transitivity of $\succeq$, this leads to

$$g_j^\star \succeq f_j \,. \tag{12}$$

We now analyze the two cases of the algorithm.

- **Case 1: $\mathcal{V}_j^\star \neq \mathcal{V}(f_j)$.** By Eqn. (12), we already know $g_j^\star \succeq f_j$. We claim that $f_j \not\succeq g_j^\star$. Otherwise, applying Lemma F.4 to $g_j^\star$ with $h = f_j$ would imply $\mathcal{V}(f_j) = \mathcal{V}_j^\star$, contrary to the assumption of this case. Thus, we have $g_j^\star \succ f_j$. Together with Eqn. (11), this shows that the algorithm outputs $g_j = g_j^\star \in \mathcal{G}_\succ$ and $g_j \succ f_j$ in this case.
- **Case 2: $\mathcal{V}_j^\star = \mathcal{V}(f_j)$.** Since every vector in $\mathcal{V}_j^\star$ spans an extreme ray of $\mathcal{M}$, the same is true for every vector in $\mathcal{V}(f_j)$. Applying Lemma F.4 directly to f_j gives $f_j \in \mathcal{G}_\succ$. Thus, there does not exist any predictor $h \in \mathcal{G}_\succ$ such that $h \succ f_j$. In this case the algorithm outputs $g_j = f_j$, exactly as required by the specification of PREDICTORREFINE.

Since the above argument holds for every $j \in [k]$, the algorithm is correct.

It remains to prove the running-time bound. Let L_{in} denote the total binary encoding length of the input: namely, of λ, of all rational entries of the input kernels, and of all rational input signal predictions. From the kernels, the reduced bags $\mathcal{V}(f_j)$ can be computed in polynomial time by first forming the raw vectors and then merging equal normalized ray representatives. For example, for every nonzero rational vector $\mathbf{v}$, define its normalized representative

$$\widehat{\mathbf{v}} \triangleq \frac{\mathbf{v}}{\sum_{i\in[n]} \mathbf{v}_i} \,.$$

Since $\mathbf{v} \in \mathbb{R}_+^n \setminus \{\mathbf{0}\}$, two such vectors lie on the same positive ray if and only if their normalized representatives are equal. Thus both ray-reduction and equality testing of reduced bags are polynomial-time operations over rational data.

The matrix $\mathbf{M}$ and the vector $\mathbf{b}$ of Definition 3.4 are computable in polynomial time from the reduced bags and the input signal predictions. Every LP instance FEASIBLE(S) has $N+n$ variables and a number of linear constraints polynomial in $N+n$, with rational coefficients of bit-length polynomial in L_{in}. Therefore every feasibility test and every feasible solution can be computed in time polynomial in L_{in} by standard polynomial-time algorithms for rational linear programming.

By Algorithm 3, one call to FINDEXTREMESIGNAL performs at most n LP-feasibility tests and one final LP solve. By Lemma F.3, one call to EXTREMERAYDECOMPOSE performs at most n iterations. Thus, one call to EXTREMERAYDECOMPOSE uses at most $O(n^2)$ LP solves. Since the total number of input reduced-bag rows is $N = \sum_{j\in[k]} m_j$, the whole algorithm performs at most $O(N\,n^2)$ LP solves.

Finally, each input row yields at most n extreme-ray pieces, so the unreduced bag $\widetilde{\mathcal{V}}_j$ has size at most $m_j n$, and the final reduced bag $\mathcal{V}_j^\star$ has size at most $m_j n$ as well. For each output row $\mathbf{v} \in \mathcal{V}_j^\star$, the algorithm solves the rational linear system $\mathbf{M}^\top \xi = \mathbf{v}$ and computes the rational number $p(\mathbf{v}) = \xi^\top \mathbf{b}/\lambda(\mathbf{v})$; this takes polynomial time. Therefore the total running time of Algorithm 5 is polynomial in the input size L_{in}.

Thus Algorithm 5 is a polynomial-time algorithm that solves PREDICTORREFINE under informational dominance. □

Lemma F.5 (Ray merging preserves dominance)**.** *Let $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_m\} \subseteq \mathbb{R}_+^n \setminus \{\mathbf{0}\}$ be a finite bag satisfying $\sum_{\ell\in[m]} \mathbf{v}_\ell = \mathbf{1}$. Let $\mathcal{V}^{\text{red}}$ be obtained from $\mathcal{V}$ by merging every $\sim_{\text{ray}}$-equivalence*

class into the sum of all vectors in that class. Let g^{raw} and g^{red} be the predictors represented by $\mathcal{V}$ and $\mathcal{V}^{\mathrm{red}}$, respectively. Then $g^{\mathrm{raw}} \succeq g^{\mathrm{red}}$ and $g^{\mathrm{red}} \succeq g^{\mathrm{raw}}$. Consequently, for every finitely supported predictor h,

$$h \succ g^{\mathrm{raw}} \iff h \succ g^{\mathrm{red}} \; ; \quad g^{\mathrm{raw}} \succ h \iff g^{\mathrm{red}} \succ h \; .$$

Proof. Let $\mathcal{R}_1, \ldots, \mathcal{R}_L$ be the distinct $\sim_{\mathrm{ray}}$-equivalence classes in $\mathcal{V}$, and for each $t \in [L]$ write

$$\bar{\mathbf{v}}_t \triangleq \sum\nolimits_{\mathbf{v}_\ell \in \mathcal{R}_t} \mathbf{v}_\ell \; .$$

Thus $\mathcal{V}^{\mathrm{red}} = \{\bar{\mathbf{v}}_t\}_{t \in [L]}$. To obtain g^{red} from g^{raw}, map every raw signal $\ell \in [m]$ deterministically to the unique class $t \in [L]$ with $\mathbf{v}_\ell \in \mathcal{R}_t$. This is a feature-independent garbling, so $g^{\mathrm{raw}} \succeq g^{\mathrm{red}}$.

For the reverse direction, fix a class $\mathcal{R}_t$. Choose any representative $\mathbf{v}_t^\star \in \mathcal{R}_t$. For every $\mathbf{v}_\ell \in \mathcal{R}_t$, there exists a unique scalar $\alpha_\ell > 0$ such that $\mathbf{v}_\ell = \alpha_\ell \mathbf{v}_t^\star$. Thus,

$$\bar{\mathbf{v}}_t = \sum\nolimits_{\mathbf{v}_\ell \in \mathcal{R}_t} \alpha_\ell \mathbf{v}_t^\star = \Big(\sum\nolimits_{\mathbf{v}_\ell \in \mathcal{R}_t} \alpha_\ell\Big) \mathbf{v}_t^\star \; .$$

Define $\gamma_\ell \triangleq \frac{\alpha_\ell}{\sum_{\mathbf{v}_r \in \mathcal{R}_t} \alpha_r}$ for each $\mathbf{v}_\ell \in \mathcal{R}_t$. Then $\gamma_\ell \geq 0$, $\sum_{\mathbf{v}_\ell \in \mathcal{R}_t} \gamma_\ell = 1$, and $\mathbf{v}_\ell = \gamma_\ell \bar{\mathbf{v}}_t$ for every $\mathbf{v}_\ell \in \mathcal{R}_t$. Therefore, conditioned on the merged signal t, one can recover a raw signal inside $\mathcal{R}_t$ by an independent random split with probabilities $(\gamma_\ell)_{\mathbf{v}_\ell \in \mathcal{R}_t}$. Doing this independently for every class yields a feature-independent garbling from g^{red} to g^{raw}. Thus $g^{\mathrm{red}} \succeq g^{\mathrm{raw}}$.

Finally, because $\succeq$ is transitive, mutual $\succeq$-equivalence preserves every strict comparison. For example, if $h \succ g^{\mathrm{raw}}$, then $h \succeq g^{\mathrm{raw}} \succeq g^{\mathrm{red}}$, so $h \succeq g^{\mathrm{red}}$. If also $g^{\mathrm{red}} \succeq h$, then $g^{\mathrm{raw}} \succeq g^{\mathrm{red}} \succeq h$, contradicting $h \succ g^{\mathrm{raw}}$. Thus $h \succ g^{\mathrm{red}}$. The other implications are identical. □

G Signal-Level Calibration and Prediction-Level Calibration

In this section, we discuss signal-level calibration, adopted throughout the paper, and conventional prediction-level calibration. We explain how prediction-level calibrated inputs can be represented within our framework and clarify the implications for informational refinement.

Signal-level versus prediction-level calibration. Recall that a predictor $f = (\Sigma_f, T_f, p_f)$ generates a signal S_f and assigns the numerical prediction $P_f = p_f(S_f)$. Signal-level calibration requires

$$\mathbb{E}[Y \mid S_f = s] = p_f(s) \quad \text{for every } s \in \Sigma_f \text{ with } \mathbb{P}[S_f = s] > 0 \; .$$

In contrast, prediction-level calibration requires only

$$\mathbb{E}[Y \mid P_f = a] = a \quad \text{for every } a \in [0,1] \text{ with } \mathbb{P}[P_f = a] > 0 \; .$$

As shown in Section 2, signal-level calibration implies prediction-level calibration, but the converse need not hold when distinct signals share the same numerical prediction. Prediction-level calibration constrains the average label within each group of signals having the same numerical prediction, rather than the conditional label mean of each signal individually. When the prediction map is injective on the positive-probability signals, the two notions coincide, since observing the numerical prediction then determines the signal. We do not impose injectivity: our framework deliberately allows distinct signals with identical numerical predictions.

Representing prediction-level calibrated inputs. Consider a predictor $f = (\Sigma_f, T_f, p_f)$ that satisfies prediction-level calibration. We define a new predictor $\bar{f}$ by merging signals that yield the same numerical prediction. Its signal space is $\Sigma_{\bar{f}} = p_f(\Sigma_f)$, the set of distinct numerical predictions generated by f. For every $a \in \Sigma_{\bar{f}}$ and $x \in \mathcal{X}$, define

$$T_{\bar{f}}(a \mid x) \triangleq \sum\nolimits_{s \in \Sigma_f : p_f(s) = a} T_f(s \mid x) \; , \quad p_{\bar{f}}(a) \triangleq a \; .$$

By construction, the event that $\bar{f}$ generates signal a coincides with the event that f outputs numerical prediction a. Thus, for every such signal of positive probability, prediction-level calibration of f implies

$$\mathbb{E}\big[Y \mid S_{\bar{f}} = a\big] = \mathbb{E}[Y \mid P_f = a] = a = p_{\bar{f}}(a) \; .$$

Consequently, $\bar{f}$ satisfies signal-level calibration and preserves the conditional distribution of numerical predictions given each feature.

This transformation incorporates prediction-level calibrated inputs into our signal-based framework, provided that the resulting signal kernels are available. The observable system is then built from the coarsened signal vectors, for which prediction-level calibration of the original inputs supplies the required label masses. The transformation may lose information carried by the original signals, but those distinctions are unavailable when only numerical predictions are observed. In particular, coarsening by numerical value differs from ray reduction: equal-valued signals need not lie on the same positive ray.

Implications for informational refinement. The input transformation does not make signal-based informational refinement and refinement based only on numerical predictions equivalent. Two output signals may carry different information about the feature while receiving the same calibrated numerical prediction. They remain distinguishable under our signal-based formulation, but become indistinguishable when only their numerical predictions are retained.

To illustrate this distinction, consider three equally likely features A, B, and C, with Bayes probabilities $q_{\mathrm{A}} = q_{\mathrm{B}} = 0.2$ and $q_{\mathrm{C}} = 0.8$. Suppose f_1 has signal partition $\{\{\mathrm{A}, \mathrm{B}\}, \{\mathrm{C}\}\}$, while f_2 has signal partition $\{\{\mathrm{A}\}, \{\mathrm{B}, \mathrm{C}\}\}$. Their respective calibrated numerical predictions are $(0.2, 0.8)$ and $(0.2, 0.5)$. The input signal vectors span $\mathbb{R}^3$: they include $(1, 0, 0)^\top$ and $(0, 0, 1)^\top$, and

$$(0, 1, 0)^\top = (1, 1, 0)^\top - (1, 0, 0)^\top .$$

Hence, the fully separating predictor g, whose signals distinguish A, B, and C, is constructible and strictly informationally dominates both inputs under Definition 2.1.

However, the numerical predictions of g are $(0.2, 0.2, 0.8)$. Merging its first two signals because they produce the same numerical prediction gives a predictor that cannot distinguish A from B. Any feature-independent post-processing of this merged predictor must have the same conditional output distribution on A and B, whereas f_2 predicts 0.2 on A and 0.5 on B. Therefore, the merged predictor cannot even simulate the numerical predictions of f_2.

Thus, merging equal-valued output signals does not necessarily preserve informational dominance over the original inputs. The maximality and refinement guarantees established in this paper concern signal-based informational dominance and do not automatically extend to a formulation in which only numerical predictions are observable outputs. In particular, input coarsening alone does not justify transferring these guarantees through numerical-output post-processing.

Our choice of signal-level calibration and signal-based informational dominance therefore defines a distinct information-refinement problem. It preserves informational distinctions among signals, supports the observable linear construction in Section 3.2, and is the formulation addressed by the algorithmic results in Sections 4 and 5.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The paper contains only theoretical results backed by formal proofs. The abstract and introduction summarize these theoretical results and discuss their potential applications.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We present both positive results (algorithms) and negative results (computational hardness). No major assumptions are needed, and all aspects of the model are clearly defined and discussed.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: All assumptions and proofs are provided.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [N/A]

Justification: N/A.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [N/A]

   Justification: N/A.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [N/A]

   Justification: N/A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [N/A]

   Justification: N/A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [N/A]

   Justification: N/A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: All items of the Code are honored.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: The paper is theoretically focused, and we do not see direct broader impacts in the sense concerned here.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: No data or models are released.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [N/A]

    Justification: No assets are used.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: No assets are released.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: No crowdsourcing or research with human subjects are involved.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No crowdsourcing or research with human subjects are involved.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not involved in substantial ways.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.