JMLR: Workshop and Conference Proceedings vol 35:1–16, 2014

Robust Multi-objective Learning with Mentor Feedback

Alekh Agarwal ALEKHA@MICROSOFT.COM
Microsoft Research, New York, NY USA

Ashwinkumar Badanidiyuru ASHWINKUMARBV@GMAIL.COM
Cornell University, Dept of Computer Science, Ithaca, NY USA

Miroslav Dudík MDUDIK@MICROSOFT.COM
Microsoft Research, New York, NY USA

Robert E. Schapire SCHAPIRE@CS.PRINCETON.EDU
Princeton University, Dept of Computer Science, Princeton, NJ USA

Aleksandrs Slivkins SLIVKINS@MICROSOFT.COM
Microsoft Research, New York, NY USA

Abstract

We study decision making when each action is described by a set of objectives, all of which are to be maximized. During the training phase, we have access to the actions of an outside agent ("mentor"). In the test phase, our goal is to maximally improve upon the mentor's (unobserved) actions across all objectives. We present an algorithm with a vanishing regret compared with the optimal possible improvement, and show that our regret bound is the best possible. The bound is independent of the number of actions, and scales only as the logarithm of the number of objectives.

Keywords: multi-objective learning, apprenticeship learning, random matrix games

1. Introduction

We study the problem of learning in the presence of multiple objectives. In this problem, the learner must develop the ability to select among a set of actions, given contextual information, so as to maximize payoff relative to some convex combination of objectives. The contextual information consists of the values of all objectives across all possible actions. The *correct* combination of objectives is never revealed to the learner, either because it is unknown or because it is genuinely undefined. (For example, there may be multiple stake-holders in the decision problem, and the learner may wish to perform well with respect to all of their subjective utilities.) Thus, the learner must learn in a robust way, so as to perform well with respect to *any* combination of the objectives.

The learner's goal is to compete with the decisions of an outside agent, called the *mentor*, whose actions are observable during training but not during testing. Furthermore, not only does the learner wish to imitate the mentor, but also attempts to improve on the choices of the mentor. In other words, when possible, the learner strives to achieve the payoff (with respect to the unknown correct combination of objectives) that is even better than that achieved by the mentor. In this sense, our work follows on and directly generalizes the approach of Syed and Schapire (2008).

Although it may sometimes be reasonable to suppose that the mentor makes optimal decisions with respect to some fixed (but unknown) combination of objectives, we do not make any such assumptions about the mentor's behavior. For example, the mentor may follow a relaxed rationality

model such as "quantal response," or optimize for several combinations of objectives simultaneously, or simply choose her actions without any specific combination of objectives in mind.

One possible application is a "digital travel assistant" helping to choose a travel itinerary for a user. During the training phase, the user acts as a mentor, selecting the most suitable itinerary from the available options. Each itinerary is described by objectives such as the duration of the flight, the number of transfers, presence of WiFi, legroom, price, etc. After the training phase, the learner (i.e., the digital assitant) will be able to select preferred itineraries for future trips, with the guarantee that its choices will be not much worse (and possibly even better) than those of the user.

Our contributions. Our learner can *outperform* the mentor in a very strong sense: its improvement in performance relative to the mentor, across all combinations of objectives, essentially matches that achieved by the *best possible* decision rule for choosing the action based on the contextual information. We bound the difference in performance between our learner and the best decision rule, as a function of the number of training examples and the number of objectives, and we prove a matching lower bound. Further, we extend this result to a slightly more general setting in which we observe the scores achieved by the mentor for the various objectives, but not the mentor's actions. As a special case in this setting, we recover Syed and Schapire's (2008) result on apprenticeship learning of MDPs. Our algorithm and analysis proceed by a reduction to random matrix games. Our guarantees for the random matrix games may be of independent interest.

While our main algorithmic result is for a batch setting, we also obtain a similar result for the online setting in which the training examples arrive over time. This result is stronger in that it extends to adversarial environments, and weaker in that the benchmark is the mentor feedback rather than the best possible decision rule.

Organization of the paper. In Section 2, we introduce our setting and main results more formally. In Section 3, we explain why some simple strategies do not work. We derive our algorithm and prove the performance guarantees in Sections 4 and 5. The lower bound is in Section 6.

Related work. The setting when the learner gets to learn from the mentor's actions, which are assumed to optimize an unknown combination of objectives describing the mentor's utility, is akin to a cluster of related problem areas known as inverse reinforcement learning, or "apprenticeship" or "imitation" learning (Abbeel and Ng, 2004; Ratliff et al., 2006; Ziebart et al., 2008; Syed and Schapire, 2008). Our treatment, which tries to outperform the mentor, is most similar to the work of Syed and Schapire (2008) and we describe the connection in Appendix A. There is also a large body of literature in marketing and behavioral economics, where the goal is to learn a utility function of an individual (as a linear combination of objectives), both from observational data as well as via active learning (experimental design) methods (see, for instance, Train, 2009; Green and Srinivasan, 1990). Our framework differs in that it does not assume a specific user model and attempts to perform well also in cases when there is no "true" objective combination.

2. Problem formulation and main results

Our model, called *robust multi-objective learning*, is defined as follows. The learner receives m training examples, followed by one test example. In each example, the contextual information is expressed as a *context matrix* $\mathbf{R}$. The columns of $\mathbf{R}$ correspond to the various objectives, while the rows are the possible actions for the given context. A given element of the matrix $R(a, o)$ then specifies the payoff under objective o if action a is taken. For simplicity of presentation, we assume

that all such entries are in the range $[-1, 1]$. The number of columns (objectives) is fixed to be n in all examples; however, the number of rows (actions) may vary from one example to the next.

A (convex) combination of objectives is defined by a probability vector $\mathbf{q} \in [0,1]^n$ over the objectives. Likewise, a (possibly randomized) action may be given by a probability vector $\mathbf{p}$ over the rows of $\mathbf{R}$. The resulting payoff on context matrix $\mathbf{R}$ is then given naturally by $\mathbf{p}^\top \mathbf{R}\mathbf{q}$.

Each training example is annotated with a (possibly randomized) action $\overline{\mathbf{p}}$ selected by the mentor for a given context matrix $\mathbf{R}$. The t-th training example is denoted $(\mathbf{R}_t, \overline{\mathbf{p}}_t)$; we assume it is drawn independently from some unknown but fixed distribution $\mathcal{D}$.

Given the training examples, the learner must output a *decision rule*: a function $\hat{\mathbf{p}} = \hat{\mathbf{p}}(\mathbf{R})$ from context matrices to randomized actions. The goal is to achieve expected payoff that is competitive with that attained by the mentor, with respect to any given combination of objectives $\mathbf{q}$. Formally, the performance of a given decision rule $\hat{\mathbf{p}}$ is summarized by the following performance criterion:

$$\texttt{Perf}(\hat{\mathbf{p}}) = \min_{\mathbf{q}} \mathop{\mathbb{E}}_{(\mathbf{R},\overline{\mathbf{p}})\sim\mathcal{D}} \left[\hat{\mathbf{p}}(\mathbf{R})^\top \mathbf{R}\mathbf{q} - \overline{\mathbf{p}}^\top \mathbf{R}\mathbf{q} \right] \ .$$

In words, we consider the difference between the learner's expected payoff and the mentor's expected payoff. This difference depends on the particular $\mathbf{q}$; we define $\texttt{Perf}(\hat{\mathbf{p}})$ as the smallest possible difference for any $\mathbf{q}$. This means that $\texttt{Perf}(\hat{\mathbf{p}})$ is the amount by which $\hat{\mathbf{p}}$ is guaranteed to outperform the mentor, in expectation, on every combination of objectives.

Main results. We design an algorithm which is comparable not only to the mentor, but also to the performance of the *best decision rule*: $\texttt{Perf}^* = \sup_{\mathbf{p}^*} \texttt{Perf}(\mathbf{p}^*)$, where the sup is over all measurable decision rules $\mathbf{p}^*$. Our main result is stated as follows:

Theorem 1 *Consider robust multi-objective learning with n objectives and m training examples. Our algorithm computes a decision rule $\hat{\mathbf{p}}$ such that for each $\delta > 0$,*

$$\Pr\left[\texttt{Perf}(\hat{\mathbf{p}}) \geq \texttt{Perf}^* - O\left(\sqrt{\tfrac{1}{m} \ln(n/\delta)} \right) \right] \geq 1 - \delta \ .$$

Consider the decision rule $\tilde{\mathbf{p}}(\mathbf{R}) := \mathbb{E}_{\mathcal{D}}[\overline{\mathbf{p}}|\mathbf{R}]$ defined by mentor feedback. Its performance is equivalent to first drawing $\overline{\mathbf{p}}$ from the conditional distribution $\mathcal{D}(\overline{\mathbf{p}}|\mathbf{R})$ and then acting according to $\overline{\mathbf{p}}$. Note that $\texttt{Perf}^* \geq \texttt{Perf}(\tilde{\mathbf{p}}) = 0$, which implies that our learner is not much worse than the mentor. Furthermore, our learner can be *better* than the mentor if $\texttt{Perf}^*$ is large. In fact, we believe that this situation is fairly common in a range of realistic settings (see Section 3).

Note that the regret bound scales only as the logarithm of the number of objectives. Interestingly, it is independent of the number of actions. While the regret bound is analogous to that for learning with n experts, we observe that the benchmark here is much stronger than in the standard experts settings—we compete with all mappings $\hat{\mathbf{p}}(\mathbf{R})$, as opposed to a single expert. Curiously, our bound behaves like the regret of competing against n experts, and not K experts even though we induce probabilities over K actions through the mappings $\hat{\mathbf{p}}(\mathbf{R})$. This is because our analysis uses duality to run a low-regret online algorithm in the space of $\mathbf{q}$ distributions rather than distributions over actions, and this duality is a key to our algorithm and analysis. In fact, the number of actions is not fixed and may vary from one example to the next.

The reader might wonder why we use an online algorithm, even though our main focus is on an i.i.d. batch setting. The reason is that applying standard analysis techniques for batch learning in our setup leads to suboptimal bounds ($O(\sqrt{n})$ as opposed to $\log n$). This is because we typically

require a uniform deviations argument in analyzing the batch algorithm, while the online-to-batch conversion here only relies on the Azuma-Hoeffding bound. We also point out that our algorithm in the next section will use `Hedge` as the online learning algorithm, in order to obtain concrete constants for the main theorem. However, we can replace `Hedge` with any other online learner having similar regret guarantees, affecting only the constants in our result.

We also prove a lower bound matching the result of Theorem 1:

Theorem 2 *In the setting of Theorem 1, for some absolute constant C, no algorithm can achieve*

$$\Pr\left[\texttt{Perf}(\hat{\mathbf{p}}) \geq \texttt{Perf}^* - C\sqrt{\tfrac{1}{m}\ln(n)}\right] \geq \tfrac{1}{2} \ .$$

A generalization: relaxed mentor feedback. The mentor's feedback can be summarized by $\mathbf{s}^\top = \overline{\mathbf{p}}^\top \mathbf{R}$, a vector of scores over the objectives; let us call it the *score vector*. We consider a relaxed model in which the algorithm is given score vectors, but not the actions $\overline{\mathbf{p}}$ themselves. Formally, each training example now consists of a pair $(\mathbf{R}, \mathbf{s})$, selected from some fixed but unknown distribution $\mathcal{D}$ (now modified to be over pairs of this form). The performance criterion is then

$$\texttt{Perf}(\hat{\mathbf{p}}) = \min_{\mathbf{q}} \mathop{\mathbb{E}}_{(\mathbf{R},\mathbf{s})\sim\mathcal{D}} \left[\hat{\mathbf{p}}(\mathbf{R})^\top \mathbf{R}\mathbf{q} - \mathbf{s}^\top \mathbf{q}\right] \ . \tag{1}$$

We prove that Theorem 1 extends to this relaxed setting. In particular, this extension subsumes MDP-based setting for apprenticeship learning of Abbeel and Ng (2004) and Syed and Schapire (2008), and essentially recovers the main result of Syed and Schapire (2008). This application is further discussed in Appendix A.

A special case: no mentor feedback. Our solution proceeds by first solving the case of no mentor feedback. Formally, this is a special case of relaxed mentor feedback where $\mathbf{s}$ is the zero vector. Then the performance criterion simplifies to

$$\texttt{Perf}(\hat{\mathbf{p}}) = \min_{\mathbf{q}} \mathop{\mathbb{E}}_{\mathbf{R}\sim\mathcal{D}} \left[\hat{\mathbf{p}}(\mathbf{R})^\top \mathbf{R}\mathbf{q}\right] \ . \tag{2}$$

This special case may be of independent interest. Here, the learner plays a *random matrix game* against an opponent who seeks to minimize the learner's expected payoff. In this setting, the learner first chooses a decision rule $\hat{\mathbf{p}}(\cdot)$, the opponent next chooses a convex combination $\mathbf{q}$ over objectives, and finally the context matrix $\mathbf{R}$ arrives, selected at random from some fixed but unknown distribution $\mathcal{D}$. For an optimal opponent (with complete knowledge of $\mathcal{D}$ and $\hat{\mathbf{p}}(\cdot)$), it can be seen that the resulting payoff will be exactly the quantity given in Eq. (2). Alternatively, this setting can be viewed as one in which there are multiple objectives with payoffs as given in the context matrix $\mathbf{R}$, but no mentor feedback; the goal then is to perform well in a certain conservative sense, doing as well as possible even for a worst-case combination of objectives.

3. Intuition: two algorithms that do not work, and why mentor can be bested

To build intuition, let us consider two seemingly reasonable algorithms whose performance is substantially inferior to that in Theorem 1.

Idea 1: maximize the minimum payoff. Consider an algorithm that simply maximizes the minimum payoff; that is, given a context matrix $\mathbf{R}$, this algorithm chooses the randomized action

$$\hat{\mathbf{p}}(\mathbf{R}) = \operatorname*{argmax}_{\mathbf{p}} \min_{\mathbf{q}} \mathbf{p}^\top \mathbf{R}\mathbf{q} \ . \tag{3}$$

This algorithm seems natural even though it does not learn from the training examples.

Consider the following problem instance. There are two actions and two objectives, and two context matrices $\mathbf{R}_1 = \left(\begin{smallmatrix} 2 & 0 \\ 0 & 1 \end{smallmatrix}\right)$ and $\mathbf{R}_2 = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 2 \end{smallmatrix}\right)$. Each of the two context matrices is selected with equal probability. Given a particular $\mathbf{R}_j$, $j \in \{1, 2\}$, the mentor chooses $\overline{\mathbf{p}}_j$, where $\overline{\mathbf{p}}_1^\top = (1/3,\ 2/3)$ and $\overline{\mathbf{p}}_2^\top = (2/3,\ 1/3)$, completing the description of the problem instance.

One can show that for this problem instance, the learner chooses exactly the same randomized action as the mentor: $\hat{\mathbf{p}}(\mathbf{R}_j) = \overline{\mathbf{p}}_j$, $j \in \{1, 2\}$. Thus, we trivially achieve $\texttt{Perf}(\hat{\mathbf{p}}) = 0$.

However, our goal is to *surpass* mentor's performance when possible, i.e., achieve $\texttt{Perf}(\hat{\mathbf{p}}) \geq c > 0$ for some absolute constant c. This is possible for this problem instance by selecting, for example, a simple decision rule $\hat{\mathbf{p}}(\mathbf{R})^\top = (1/2,\ 1/2)$, which achieves $\texttt{Perf}(\hat{\mathbf{p}}) = 1/12$.

(Indeed, let $\hat{\mathbf{p}}(\mathbf{R})^\top = (1/2,\ 1/2)$. Then $\mathbb{E}\left[\hat{\mathbf{p}}^\top \mathbf{R}\right] = (3/4,\ 3/4)$, and $\mathbb{E}\left[\hat{\mathbf{p}}^\top \mathbf{R}\mathbf{q}\right] = 3/4$ for any $\mathbf{q}$. Furthermore, a simple computation shows that $\overline{\mathbf{p}}_j^\top \mathbf{R}_j = (2/3,\ 2/3)$, for $j \in \{1, 2\}$. Therefore, for any $\mathbf{q}$, we have $\mathbb{E}\left[\hat{\mathbf{p}}^\top \mathbf{R}\mathbf{q}\right] - \mathbb{E}\left[\overline{\mathbf{p}}^\top \mathbf{R}\mathbf{q}\right] = 3/4 - 2/3 = 1/12$.)

Idea 2: a possible fix. Note that we can re-write our performance criterion as

$$\texttt{Perf}(\hat{\mathbf{p}}) = \min_{\mathbf{q}} \Big[\mathbb{E}\big[\hat{\mathbf{p}}^\top \mathbf{R}\mathbf{q}\big] - \mathbb{E}\big[\overline{\mathbf{p}}^\top \mathbf{R}\big]\mathbf{q}\Big] \ .$$

This suggests first estimating $\mathbb{E}\left[\overline{\mathbf{p}}^\top \mathbf{R}\right]$ from the training examples by some estimator $\mathbf{u}$, and then using this estimator to choose the randomized action in a max-min fashion:

$$\hat{\mathbf{p}}(\mathbf{R}) = \operatorname*{argmax}_{\mathbf{p}} \min_{\mathbf{q}} \left(\mathbf{p}^\top \mathbf{R}\mathbf{q} - \mathbf{u}^\top \mathbf{q}\right) \ . \tag{4}$$

However, consider the same problem instance as above, and assume for simplicity that the estimator $\mathbf{u}$ is 100% accurate, i.e., $\mathbf{u}^\top = \mathbb{E}\left[\overline{\mathbf{p}}^\top \mathbf{R}\right]$. Then $\mathbf{u}^\top = (2/3,\ 2/3)$, and $\mathbf{u}^\top \mathbf{q} = 2/3$, regardless of $\mathbf{q}$. It follows that the decision rule in Eq. (4) coincides with the one in Eq. (3), and therefore has exactly the same drawback.

Outperforming the mentor. We finish this section by showing that our goal to outperform the mentor is meaningful, i.e., it is possible to achieve $\texttt{Perf}^* > 0$ in realistic scenarios. As we briefly discussed in Section 1, in practice the mentor would not optimize exactly with respect to a specific combination of objectives, and instead would behave only approximately optimally, leaving some room for improvement. However, even in cases where the mentor actually attempts to choose optimal actions with respect to some fixed weighting $\mathbf{q}^*$, the mentor can be severely suboptimal if the number of actions is very large, because she would not typically explore all actions very carefully.

Consider the digital travel assistant example from Section 1, where the mentor selects among possible itineraries. Realistically, the system may allow to sort the itineraries on each objective individually, but not according to a given combination of objectives. The mentor accesses the itineraries through such sorted lists, and scans only the first few entries of each list. This restriction generically leads to many examples with $\texttt{Perf}^* > 0$. For instance, assume the mentor has a near-uniform weighting over the objectives. Accordingly, she picks an objective fairly uniformly each time t,

looks at the corresponding sorted list, and then picks the best action on that objective. For any single objective, there are many actions that are optimal (say, reward 1) on that objective, but obtain a very low reward on others, whereas there is a uniformly good action a_t^*, with reward $1-\varepsilon$ on each objective. This a_t^* would never be noticed, as it would not appear high enough in any sorted list. Yet, a_t^* would get expected reward $1-\varepsilon$ on each objective, whereas the top actions in the sorted lists only get $1/n$. Thus, the decision rule that picks a_t^* in each round achieves $\texttt{Perf}^* = 1-\varepsilon-1/n$.

While this is just a simple, stylized example, we believe it showcases a very general phenomenon. Additional examples from apprenticeship learning are described by Syed and Schapire (2008).

4. Robust multi-objective learning: algorithm and analysis

Tool: an online learning algorithm. Even though we consider the batch setting, we make use of an online learning algorithm Hedge (Freund and Schapire, 1997). As remarked in the previous section, we can replace Hedge with any other online learner that enjoys a similar regret guarantee. In this online setting, time proceeds in rounds $t = 1, \ldots, T$, where the time horizon T is known to the algorithm. In each round t, the learner selects a probability vector $\mathbf{q}_t \in [0,1]^n$ and reveals it to the adversary. Then the adversary responds with a loss vector $\boldsymbol{\ell}_t \in [-1,1]^n$ that is revealed to the algorithm. The resulting loss is $\mathbf{q}_t^\top \boldsymbol{\ell}_t = \sum_{i=1}^n q_t(i)\,\ell_t(i)$. (For readability, we sometimes use $x(i)$ to denote the i-th component of a vector $\mathbf{x}$.) The goal is to make the cumulative loss $\sum_{t=1}^T \mathbf{q}_t^\top \boldsymbol{\ell}_t$ small relative to the best fixed vector $\mathbf{q}$ selected in hindsight. We have the following guarantee:

Theorem 3 (Freund and Schapire, 1997) *Let $\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_T \in [-1,1]^n$ be the sequence of loss vectors in a given execution of* Hedge*, and let $\mathbf{q}_t$ be the probability vector selected by* Hedge *in each round t. Then*

$$\sum_{t=1}^T \mathbf{q}_t^\top \boldsymbol{\ell}_t \leq \min_{\mathbf{q}} \sum_{t=1}^T \mathbf{q}^\top \boldsymbol{\ell}_t + 2\sqrt{2T\ln n} + 2\ln n \ ,$$

where the minimum is taken over all probability vectors $\mathbf{q}$.

4.1. Special case: no mentor feedback

Let us consider robust multi-objective learning with no mentor feedback. Our approach reduces this special case to a carefully designed application of Hedge.

The general approach is as follows. We order the training examples in an arbitrary way, and treat the t-th example as the t-th round in the online setting. The corresponding loss vector $\boldsymbol{\ell}_t$ is defined as a function of the t-th training example, for each round t. We simulate an execution of Hedge on these loss vectors, record the probability vectors $\mathbf{q}_t$ returned by Hedge, and use these probability vectors to define our decision rule. Of course, the approach as described so far is fairly standard; the challenge is how to define the loss vectors $\boldsymbol{\ell}_t$, and how to interpret the probability vectors $\mathbf{q}_t$.

The details are provided in Algorithm 1. In what follows, we analyze Algorithm 1 and prove that it satisfies the regret bound of Theorem 1. Thus we prove the theorem for the special case of no mentor feedback.

Note that the loss suffered by Hedge in round t of Algorithm 1 is

$$\boldsymbol{\ell}_t^\top \mathbf{q}_t = \mathbf{p}_t^\top \mathbf{M}_t \mathbf{q}_t = \max_{\mathbf{p}} \mathbf{p}^\top \mathbf{M}_t \mathbf{q}_t \ . \tag{8}$$

Algorithm 1 (algorithm for robust multi-objective learning without mentor feedback)

Given: the training examples: the context matrices $\mathbf{M}_1, \ldots, \mathbf{M}_m$ with entries in $[-1, 1]$

In each round $t = 1, \ldots, m$

- **Observe** the probability vector $\mathbf{q}_t$ selected by `Hedge`.
- **Define** the loss vector $\boldsymbol{\ell}_t$ as follows:

$$\mathbf{f}_t(\mathbf{M}) = \operatorname{argmax}_{\mathbf{p}} \left[\mathbf{p}^\top \mathbf{M}\, \mathbf{q}_t\right] \qquad \text{for any context matrix } \mathbf{M} \ . \tag{5}$$

$$\mathbf{p}_t = \mathbf{f}_t(\mathbf{M}_t) \ . \tag{6}$$

$$\boldsymbol{\ell}_t^\top = \mathbf{p}_t^\top \mathbf{M}_t \ . \tag{7}$$

  (The argmax is over all probability vectors $\mathbf{p}$ of the appropriate number of dimensions.)

- **Submit** the loss vector $\boldsymbol{\ell}_t$ to `Hedge`.

Output the decision rule $\hat{\mathbf{p}}$ as the average of the $\mathbf{f}_t$'s across all rounds:

$$\hat{\mathbf{p}}(\mathbf{M}) = \tfrac{1}{m} \textstyle\sum_{t=1}^m \ \mathbf{f}_t(\mathbf{M}) \ .$$

We prove that w.h.p. $\hat{\mathbf{p}}$ selects actions that are nearly optimal w.r.t. our performance criterion

$$\texttt{Perf}(\hat{\mathbf{p}}) = \min_{\mathbf{q}} \mathop{\mathbb{E}}_{\mathbf{M}\sim\mathcal{D}} \left[\hat{\mathbf{p}}(\mathbf{M})^\top \mathbf{M}\mathbf{q}\right] \ .$$

We will show that this quantity is almost as large as

$$\texttt{Perf}^* = \sup_{\mathbf{p}^*} \min_{\mathbf{q}} \mathop{\mathbb{E}}_{\mathbf{M}\sim\mathcal{D}} \left[\mathbf{p}^*(\mathbf{M})^\top \mathbf{M}\mathbf{q}\right] \ , \tag{9}$$

where $\mathbf{p}^*$ is a mapping from matrices $\mathbf{M}$ to mixed strategies over the rows of $\mathbf{M}$. It is understood that this supremum is taken over all such (measurable) mappings.

For fixed $\mathbf{q}$, let us define random variables

$$Z_t = \mathbf{f}_t(\mathbf{M}_t)^\top \mathbf{M}_t \mathbf{q} - \mathbb{E}_t \left[\mathbf{f}_t(\mathbf{M}_t)^\top \mathbf{M}_t \mathbf{q}\right] = \mathbf{f}_t(\mathbf{M}_t)^\top \mathbf{M}_t \mathbf{q} - \mathop{\mathbb{E}}_{\mathbf{M}\sim\mathcal{D}} \left[\mathbf{f}_t(\mathbf{M})^\top \mathbf{M}\mathbf{q}\right] \ ,$$

where we use the notation $\mathbb{E}_t[\cdot]$ to denote the conditional expectation at time t, conditioning on all the randomness at rounds 1 through $t-1$. The second equality follows since $\mathbf{f}_t$ is deterministically defined in terms of $\mathbf{M}_1, \ldots, \mathbf{M}_{t-1}$, and $\mathbf{M}_t$ is an i.i.d. draw from the distribution $\mathcal{D}$. Thus, the Z_t's form a sequence of martingale differences with respect to the $\mathbf{M}_t$'s since $\mathbb{E}_t[Z_t] = 0$. Moreover, $|Z_t|$ is at most 2 since the entries of the matrices are all in $[-1, 1]$. Therefore, by Azuma's inequality

$$\Pr\left[\tfrac{1}{m} \textstyle\sum_{t=1}^m Z_t \le C_\delta\right] \ge 1 - \delta \ ,$$

where the confidence term is $C_\delta = 2\sqrt{2\ln(1/\delta)\,/\,m}$.

We can apply this argument separately to each of the n basis vectors $\mathbf{q}$. Replacing δ by $\delta' = \delta/2n$ and applying union bound, we get that with probability at least $1 - \delta/2$, for every basis vector $\mathbf{q}$,

$$\frac{1}{m}\sum_{t=1}^m \mathop{\mathbb{E}}_{\mathbf{M}\sim\mathcal{D}} \left[\mathbf{f}_t(\mathbf{M})^\top \mathbf{M}\mathbf{q}\right] \ge \frac{1}{m}\sum_{t=1}^m \mathbf{f}_t(\mathbf{M}_t)^\top \mathbf{M}_t \mathbf{q} - C_{\delta'} \ . \tag{10}$$

Next, if Eq. (10) holds for every basis vector, then it must also hold for every probability vector $\mathbf{q}$ simply by taking a weighted average, according to $\mathbf{q}$, of the corresponding basis-vector inequalities.

Let $\mathbf{p}^*$ attain the supremum in Eq. (9) up to a suboptimality of at most $1/m$. By a similar argument to the one above, with probability at least $1-\delta/2$,

$$\frac{1}{m}\sum_{t=1}^{m}\mathbf{p}^*(\mathbf{M}_t)^\top\mathbf{M}_t\mathbf{q}_t \geq \frac{1}{m}\sum_{t=1}^{m}\mathop{\mathbb{E}}_{\mathbf{M}\sim\mathcal{D}}\left[\mathbf{p}^*(\mathbf{M})^\top\mathbf{M}\mathbf{q}_t\right] - C_{\delta'} \ . \tag{11}$$

This is because we can define

$$Z'_t = \mathbb{E}_t\left[\mathbf{p}^*(\mathbf{M}_t)^\top\mathbf{M}_t\mathbf{q}_t\right] - \mathbf{p}^*(\mathbf{M}_t)^\top\mathbf{M}_t\mathbf{q}_t = \mathop{\mathbb{E}}_{\mathbf{M}\sim\mathcal{D}}\left[\mathbf{p}^*(\mathbf{M})^\top\mathbf{M}\mathbf{q}_t\right] - \mathbf{p}^*(\mathbf{M}_t)^\top\mathbf{M}_t\mathbf{q}_t$$

which again forms a sequence of martingale differences with respect to the $\mathbf{M}_t$'s. (Note that $\mathbf{q}_t$ is determined by $\mathbf{M}_1,\ldots,\mathbf{M}_{t-1}$.) Eq. (11) then follows again by Azuma's inequality.

Henceforth, we assume that Eq. (10) holds for all probability vectors $\mathbf{q}$, and that Eq. (11) also holds. By the arguments above, this will be the case with probability at least $1-\delta$.

Applying the regret bound for `Hedge` (Theorem 3), as the algorithm is being applied in our setting (i.e., taking into account Eq. 7 in Algorithm 1), yields

$$\frac{1}{m}\sum_{t=1}^{m}\mathbf{p}_t^\top\mathbf{M}_t\mathbf{q}_t \leq \min_{\mathbf{q}}\frac{1}{m}\sum_{t=1}^{m}\mathbf{p}_t^\top\mathbf{M}_t\mathbf{q} + \Delta_0 \tag{12}$$

where $\Delta_0 = 2\sqrt{2\ln n/m} + 2\ln n/m$.

Putting everything together, we have:

$$\begin{aligned}
\min_{\mathbf{q}}\mathop{\mathbb{E}}_{\mathbf{M}\sim\mathcal{D}}\left[\hat{\mathbf{p}}(\mathbf{M})^\top\mathbf{M}\mathbf{q}\right] &= \min_{\mathbf{q}}\mathop{\mathbb{E}}_{\mathbf{M}\sim\mathcal{D}}\left[\tfrac{1}{m}\textstyle\sum_{t=1}^{m}\ \mathbf{f}_t(\mathbf{M})^\top\mathbf{M}\mathbf{q}\right] && (13)\\
&= \min_{\mathbf{q}}\ \tfrac{1}{m}\textstyle\sum_{t=1}^{m}\ \mathbb{E}_{\mathbf{M}\sim\mathcal{D}}\left[\mathbf{f}_t(\mathbf{M})^\top\mathbf{M}\mathbf{q}\right]\\
&\geq \min_{\mathbf{q}}\ \tfrac{1}{m}\textstyle\sum_{t=1}^{m}\ \mathbf{f}_t(\mathbf{M}_t)^\top\mathbf{M}_t\mathbf{q} - C_{\delta'} && (14)\\
&= \min_{\mathbf{q}}\ \tfrac{1}{m}\textstyle\sum_{t=1}^{m}\ \mathbf{p}_t^\top\mathbf{M}_t\mathbf{q} - C_{\delta'} && (15)\\
&\geq \tfrac{1}{m}\textstyle\sum_{t=1}^{m}\ \mathbf{p}_t^\top\mathbf{M}_t\mathbf{q}_t - \Delta_0 - C_{\delta'} && (16)\\
&= \tfrac{1}{m}\textstyle\sum_{t=1}^{m}\ \max_{\mathbf{p}}\mathbf{p}^\top\mathbf{M}_t\mathbf{q}_t - \Delta_0 - C_{\delta'} && (17)\\
&\geq \tfrac{1}{m}\textstyle\sum_{t=1}^{m}\ \mathbf{p}^*(\mathbf{M}_t)^\top\mathbf{M}_t\mathbf{q}_t - \Delta_0 - C_{\delta'} && (18)\\
&\geq \tfrac{1}{m}\textstyle\sum_{t=1}^{m}\ \mathbb{E}_{\mathbf{M}\sim\mathcal{D}}\left[\mathbf{p}^*(\mathbf{M})^\top\mathbf{M}\mathbf{q}_t\right] - \Delta_0 - C_{\delta'} - C_{\delta'} && (19)\\
&\geq \tfrac{1}{m}\textstyle\sum_{t=1}^{m}\ \min_{\mathbf{q}}\mathbb{E}_{\mathbf{M}\sim\mathcal{D}}\left[\mathbf{p}^*(\mathbf{M})^\top\mathbf{M}\mathbf{q}\right] - \Delta_0 - C_{\delta'} - C_{\delta'} && (20)\\
&= \min_{\mathbf{q}}\mathop{\mathbb{E}}_{\mathbf{M}\sim\mathcal{D}}\left[\mathbf{p}^*(\mathbf{M})^\top\mathbf{M}\mathbf{q}\right] - \Delta_0 - C_{\delta'} - C_{\delta'}\\
&= \sup_{\mathbf{p}^*}\min_{\mathbf{q}}\mathop{\mathbb{E}}_{\mathbf{M}\sim\mathcal{D}}\left[\mathbf{p}^*(\mathbf{M})^\top\mathbf{M}\mathbf{q}\right] - \tfrac{1}{m} - \Delta_0 - C_{\delta'} - C_{\delta'} \ . && (21)
\end{aligned}$$

Here is the reasoning for each line: Eq. (13) uses definition of $\hat{\mathbf{p}}$. Eq. (14) uses Eq. (10). Eq. (15) uses Eq. (6). Eq. (16) follows from Eq. (12). Eq. (17) uses Eq. (8). Eq. (18) uses the fact that a maximum (over $\mathbf{p}$) is always at least the value for any specific choice (in this case, $\mathbf{p}^*(\mathbf{M}_t)$). Eq. (19) uses Eq. (11). Eq. (20) uses the fact that a minimum (over $\mathbf{q}$) is at most the value for any specific choice (here, $\mathbf{q}_t$), and Eq. (21) follows from our specific choice of $\mathbf{p}^*$.

This completes the proof that Algorithm 1 satisfies the regret bound in Theorem 1.

4.2. The general case: relaxed mentor feedback

Now we consider the general case: robust multi-objective learning with relaxed feedback. We apply the result of Section 4.1, by calling Algorithm 1 with context matrices $\mathbf{M}_t$ defined in terms of the training examples $(\mathbf{R}_t, \mathbf{s}_t)$. The challenge here is to define an appropriate reduction and prove that it works. Our algorithm is as follows.

Algorithm 2 (algorithm for robust multi-objective learning with relaxed mentor feedback)

Given: the training examples: pairs $(\mathbf{R}_1, \mathbf{s}_1), \ldots, (\mathbf{R}_m, \mathbf{s}_m)$.

Define for each example $t = 1, \ldots, m$

$$\mathbf{M}_t = \mathbf{R}_t - \mathbf{1}\,\mathbf{s}_t^\top \ , \tag{22}$$

where $\mathbf{1}$ is a column vector of all 1's, of dimension equal to the number of rows of $\mathbf{R}_t$.

Call Algorithm 1 with matrices $\mathbf{M}_1, \ldots, \mathbf{M}_m$, to obtain the decision rule $\hat{\mathbf{p}}(\cdot)$.

In testing phase: given context matrix $\mathbf{R}$, return randomized action $\hat{\mathbf{p}}(\mathbf{R})$.

In what follows, let $\hat{\mathbf{p}}(\cdot)$ be the decision rule computed by Algorithm 2. Given a context matrix $\mathbf{R}$ and a score vector $\mathbf{s}$, define $\mathbf{M} = \mathbf{M}(\mathbf{R}, \mathbf{s}) = \mathbf{R} - \mathbf{1}\,\mathbf{s}^\top$ as in Eq. (22).

Consider an instance of robust multi-objective learning with relaxed mentor feedback; call it the *original problem*. Let $\mathcal{D}$ be the distribution from which the examples $(\mathbf{R}, \mathbf{s})$ are sampled. We reduce this problem instance to the no-feedback problem in which the examples are sampled from distribution $\mathcal{D}_{\texttt{red}}$ on matrices $\mathbf{M} = \mathbf{M}(\mathbf{R}, \mathbf{s})$ induced by the random choice $(\mathbf{R}, \mathbf{s}) \sim \mathcal{D}$; call this the *reduced problem*.

When run on the reduced problem, Algorithm 1 chooses an action $\hat{\mathbf{p}}(\mathbf{M})$ in the testing phase, whereas Algorithm 2 chooses an action $\hat{\mathbf{p}}(\mathbf{R})$. In order to apply the regret bound for Algorithm 1, we first prove that $\hat{\mathbf{p}}(\mathbf{R}) = \hat{\mathbf{p}}(\mathbf{M})$.

Lemma 4 $\hat{\mathbf{p}}(\mathbf{R}) = \hat{\mathbf{p}}(\mathbf{M})$, $\mathbf{M} = \mathbf{M}(\mathbf{R}, \mathbf{s})$ *for any context matrix* $\mathbf{R}$ *and a score vector* $\mathbf{s}$.

Proof Note that for any probability vector $\mathbf{p}$ (of appropriate dimension) we have

$$\begin{aligned} \mathbf{p}^\top \mathbf{M}\mathbf{q}_t &= \mathbf{p}^\top(\mathbf{R} - \mathbf{1}\mathbf{s}^\top)\mathbf{q}_t \\ &= \mathbf{p}^\top \mathbf{R}\mathbf{q}_t - (\mathbf{p}^\top \mathbf{1})(\mathbf{s}^\top \mathbf{q}_t) \\ &= \mathbf{p}^\top \mathbf{R}\mathbf{q}_t - \mathbf{s}^\top \mathbf{q}_t \ . \end{aligned}$$

Therefore $\mathbf{p}^\top \mathbf{M}\mathbf{q}_t$ and $\mathbf{p}^\top \mathbf{R}\mathbf{q}_t$ only differ by an additive scalar, $\mathbf{s}^\top \mathbf{q}_t$, which is irrelevant when computing the maximizing probability vector $\mathbf{p}$. It follows that $\mathbf{f}_t(\mathbf{M}) = \mathbf{f}_t(\mathbf{R})$ for each training example t, where $\mathbf{f}_t$ is defined by Eq. (5) in Algorithm 1, so $\hat{\mathbf{p}}(\mathbf{R}) = \hat{\mathbf{p}}(\mathbf{M})$. ■

Recall from Eq. (2) that the performance criterion for the reduced problem is

$$\texttt{Perf}_{\texttt{red}}(\hat{\mathbf{p}}) = \min_{\mathbf{q}} \ \mathop{\mathbb{E}}_{\mathbf{M} \sim \mathcal{D}_{\texttt{red}}} \left[\hat{\mathbf{p}}(\mathbf{M})^\top \mathbf{M}\mathbf{q} \right] \ .$$

By inspecting the performance criterion for the original problem (see Eq. 1), it is easy to see that $\texttt{Perf}_{\texttt{red}}(\hat{\mathbf{p}}) = \texttt{Perf}(\hat{\mathbf{p}})$, i.e., the performance of our returned rule in the reduced problem is the same as its performance in the original problem. However, the optimal values $\texttt{Perf}^*$ and $\texttt{Perf}^*_{\texttt{red}}$

do not necessarily align, because they are formally based on slightly different classes of decision rules: the former uses contexts $\mathbf{R}$, the latter uses contexts $\mathbf{M} = \mathbf{M}(\mathbf{R}, \mathbf{s})$. To address this issue, we can simply strengthen the notion of our reference class for the reduced matrix-game problem, and define

$$\texttt{Perf}^{**}_{\texttt{red}} = \sup_{\mathbf{p}^{**}} \min_{\mathbf{q}} \mathop{\mathbb{E}}_{(\mathbf{M},\mathbf{R})\sim\mathcal{D}_{\texttt{red}}} \left[\mathbf{p}^{**}(\mathbf{M},\mathbf{R})^\top \mathbf{R}\mathbf{q}\right] \ ,$$

i.e., we allow rules $\mathbf{p}^{**}$ to depend not only on contexts $\mathbf{M}$, but also on any additional information sampled by $\mathcal{D}_{\texttt{red}}$ (e.g., the original matrix $\mathbf{R}$). [1] The regret bound for Algorithm 1 still holds with $\texttt{Perf}^{*}_{\texttt{red}}$ replaced by $\texttt{Perf}^{**}_{\texttt{red}}$, because Eq. (18) in its proof remains valid. Since $\texttt{Perf}^{*} \leq \texttt{Perf}^{**}_{\texttt{red}}$, we conclude that Algorithm 2 satisfies the regret bound of Theorem 1. [2]

5. An online algorithm

Here we consider an online version of robust multi-objective learning. Here, on each round t, the learner is presented with a context matrix $\mathbf{R}_t$, must then choose a strategy $\mathbf{p}_t$, and finally is allowed to observe the mentor's choice $\overline{\mathbf{p}}_t$. The goal is for the learner to choose actions that result in payoff not much worse than that of the mentor, for any combination of the objectives.

We adapt Algorithm 2 to the online setting in a natural way.

Algorithm 3 (algorithm for the online version of robust multi-objective learning)

Given: time horizon T.

In each round $t = 1, \ldots, T$

- **Observe** the context matrix $\mathbf{R}_t$ and the probability vector $\mathbf{q}_t$ selected by `Hedge`.
- **Choose** randomized action $\mathbf{p}_t = \mathbf{f}_t(\mathbf{R}_t)$, where $\mathbf{f}_t(\cdot)$ is defined by Eq. (5).
- **Observe** mentor's randomized action $\overline{\mathbf{p}}_t$.
- **Define** the loss vector $\boldsymbol{\ell}_t^\top = \mathbf{p}_t^\top \mathbf{M}_t$, where $\mathbf{M}_t = \mathbf{R}_t - \mathbf{1}\,\overline{\mathbf{p}}_t^\top \mathbf{R}_t$.
- **Submit** the loss vector $\boldsymbol{\ell}_t$ to `Hedge`.

From the regret bound for `Hedge` (Theorem 3), it follows that for all weightings $\mathbf{q}$,

$$\textstyle\sum_{t=1}^T \mathbf{p}_t^\top \mathbf{M}_t\,\mathbf{q} \ + \ \Delta \ \geq \ \sum_{t=1}^T \mathbf{p}_t^\top \mathbf{M}_t\,\mathbf{q}_t \ = \ \sum_{t=1}^T \max_{\mathbf{p}} \mathbf{p}^\top \mathbf{M}_t\,\mathbf{q}_t \ , \tag{23}$$

where $\Delta = O\left(\sqrt{T \ln n}\right)$. Note that

$$\begin{aligned} \max_{\mathbf{p}} \mathbf{p}^\top \mathbf{M}_t \mathbf{q}_t \ &= \ \max_{\mathbf{p}} \mathbf{p}^\top \left(\mathbf{R}_t - \mathbf{1}\overline{\mathbf{p}}_t^\top \mathbf{R}_t\right)\mathbf{q}_t \\ &= \ \max_{\mathbf{p}} \mathbf{p}^\top \mathbf{R}_t \mathbf{q}_t - \overline{\mathbf{p}}_t^\top \mathbf{R}_t \mathbf{q}_t \ \geq \ 0 \end{aligned}$$

since this maximum must be at least what is obtained by choosing $\mathbf{p} = \overline{\mathbf{p}}_t$. Thus, the right-hand side of Eq. (23) is non-negative as well. Expanding the $\mathbf{M}_t$'s on the left-hand side of Eq. (23), it follows that the algorithm's total payoff is not much worse than the mentor's, for any given $\mathbf{q}$, and moreover this holds even if the pairs $(\mathbf{R}_t, \overline{\mathbf{p}}_t)$ are chosen by an adaptive adversary.

1. Note that the comparator can now construct the score vector $\mathbf{s}$ on each test example, but this is not a problem for the analysis which holds even against this stronger benchmark.
2. Strictly speaking, the coefficients of $\mathbf{M}$ are in the range $[-2, 2]$, whereas our analysis of Algorithm 1 assumes the range $[-1, 1]$. However, it is easy to extended this to any given range $[-b, b]$ by rescaling. Then Algorithm 1 satisfies the guarantee of Theorem 1 where the regret $\texttt{Perf}(\hat{\mathbf{p}}) - \texttt{Perf}^*$ is increased by a factor of b.

Theorem 5 *Consider the online version of robust multi-objective learning. Let T be the time horizon, and let $\mathbf{R}_t$ and $\overline{\mathbf{p}}_t$ be the context matrix and mentor's action for round t, and let $\mathbf{p}_t$ be the action chosen by Algorithm 3. Then for all convex combinations $\mathbf{q}$ of objectives,*

$$\sum_{t=1}^{T} \mathbf{p}_t^\top \mathbf{R}_t \mathbf{q} \geq \sum_{t=1}^{T} \overline{\mathbf{p}}_t^\top \mathbf{R}_t \mathbf{q} - O\left(\sqrt{T \ln n}\right) \ .$$

Moreover, this holds even if each pair $(\mathbf{R}_t, \overline{\mathbf{p}}_t)$ is chosen by an adaptive adversary, i.e., arbitrarily and possibly depending on the history $(\mathbf{p}_1, \ \ldots \ , \mathbf{p}_{t-1})$.

6. Lower bound

In this section, we prove Theorem 2, i.e., we show that the generalization bound for our algorithm is tight. Let ALG be a learner for the setting with the mentor feedback for n objectives. Recall, that its input is an i.i.d. sample $(\mathbf{R}_1, \overline{\mathbf{p}}_1), \ldots, (\mathbf{R}_m, \overline{\mathbf{p}}_m)$ from $\mathcal{D}$, and it returns a decision rule $\hat{\mathbf{p}}$ that maps matrices $\mathbf{R}$ into (randomized) actions. Assume that ALG guarantees for all distributions $\mathcal{D}$, with probability at least 1/2 over the draw of the sample, that the resulting $\hat{\mathbf{p}}$ achieves the performance bounded by

$$\min_{\mathbf{q}} \mathbb{E}\left[\hat{\mathbf{p}}(\mathbf{R})^\top \mathbf{R}\mathbf{q} - \overline{\mathbf{p}}^\top \mathbf{R}\mathbf{q}\right] \geq \sup_{\mathbf{p}^*} \min_{\mathbf{q}} \mathbb{E}\left[\mathbf{p}^*(\mathbf{R})^\top \mathbf{R}\mathbf{q} - \overline{\mathbf{p}}^\top \mathbf{R}\mathbf{q}\right] - \varepsilon(m) \ , \tag{24}$$

where the expectation is with respect to $(\mathbf{R}, \overline{\mathbf{p}}) \sim \mathcal{D}$, and we have expanded the definitions of $\texttt{Perf}(\hat{\mathbf{p}})$ and $\texttt{Perf}^*$.

In the remainder of the section we prove the following lemma, which immediately yields Theorem 2 (by taking any $C > c'$):

Lemma 6 *Let $n = 2K$ be even, where $K \geq 5$. Then there is an absolute constant c' such that*

$$\varepsilon(m) \geq \min\left\{\tfrac{1}{4}, \ c'\sqrt{\tfrac{\ln K}{m}}\right\} \ .$$

Proof If $\varepsilon(m) \geq \frac{1}{4}$, the lemma holds. In the remainder only consider cases where $\varepsilon(m) < \frac{1}{4}$. We begin by constructing K problem instances indexed by $I \in [K] \coloneqq \{1, \ldots, K\}$, and parameterized by $0 < \gamma \leq \frac{1}{4}$. We will show that our algorithm can be used to distinguish among these K instances, which will form the basis of our lower bound via Generalized Fano Method (Lemma 7 in Appendix B).

Our problem instances are inspired by noisy classification. For $I \in [K]$, the I-th problem instance is the distribution $\mathcal{D}_I$ over pairs $(\mathbf{x}, y)$, where $\mathbf{x}$ is sampled uniformly at random from $\{-1, 1\}^K$, and

$$y = \begin{cases} x_I & \text{with probability } \frac{1}{2} + \gamma \\ -x_I & \text{with probability } \frac{1}{2} - \gamma \ . \end{cases}$$

Given $\mathbf{x}$ and y, we define $\mathbf{R}$ and $\overline{\mathbf{p}}$ deterministically as

$$\mathbf{R} = \mathbf{R}(\mathbf{x}) \coloneqq \begin{pmatrix} \mathbf{x}^\top & -\mathbf{x}^\top \\ -\mathbf{x}^\top & \mathbf{x}^\top \end{pmatrix} \ , \qquad \overline{\mathbf{p}}^\top = \begin{cases} (1,0) & \text{if } y = 1, \\ (0,1) & \text{if } y = -1. \end{cases}$$

Note that $y = \overline{\mathbf{p}}^\top \left(\begin{smallmatrix} 1 \\ -1 \end{smallmatrix}\right)$, and we analogously define $\hat{y} = \hat{\mathbf{p}}(\mathbf{R})^\top \left(\begin{smallmatrix} 1 \\ -1 \end{smallmatrix}\right)$. Finally, let $\mathbf{R}_{1/2}$ denote the first K columns of $\mathbf{R}$, i.e., $\mathbf{R} = (\mathbf{R}_{1/2},\ -\mathbf{R}_{1/2})$, and define vectors corresponding to various sub-expressions in Eq. (24):

$$\hat{\boldsymbol{\mu}}^\top := \mathbb{E}\left[\hat{y}\mathbf{x}^\top\right] = \mathbb{E}\left[\hat{\mathbf{p}}(\mathbf{R})^\top \left(\begin{smallmatrix} 1 \\ -1 \end{smallmatrix}\right)\mathbf{x}^\top\right] = \mathbb{E}\left[\hat{\mathbf{p}}(\mathbf{R})^\top \mathbf{R}_{1/2}\right]$$
$$\boldsymbol{\mu}^\top := \mathbb{E}\left[y\mathbf{x}^\top\right] = \mathbb{E}\left[\overline{\mathbf{p}}(\mathbf{R})^\top \left(\begin{smallmatrix} 1 \\ -1 \end{smallmatrix}\right)\mathbf{x}^\top\right] = \mathbb{E}\left[\overline{\mathbf{p}}(\mathbf{R})^\top \mathbf{R}_{1/2}\right] = 2\gamma \mathbf{e}_I^\top \ ,$$

where $\mathbf{e}_I$ is the I-th vector of the standard basis.

Step 1 *For every I, with probability at least 1/2 over the draw of the sample,*

$$\min_{\mathbf{q}} \begin{pmatrix} \hat{\boldsymbol{\mu}} - \boldsymbol{\mu} \\ \boldsymbol{\mu} - \hat{\boldsymbol{\mu}} \end{pmatrix}^\top \mathbf{q} \geq -\varepsilon(m) \ . \tag{25}$$

The left-hand side of Eq. (25) just follows by rewriting the left-hand side of Eq. (24) using definitions of $\hat{\boldsymbol{\mu}}$ and $\boldsymbol{\mu}$. Therefore, we just need to show that the sup-min expression on the right-hand side of Eq. (24) is zero. To begin, note that since $\mathbf{R}$ is of the form $(\mathbf{R}_{1/2},\ -\mathbf{R}_{1/2})$, we obtain that for any $\mathbf{p}^*$, $\mathbb{E}\left[\mathbf{p}^*(\mathbf{R})^\top \mathbf{R} - \overline{\mathbf{p}}^\top \mathbf{R}\right] = (\mathbf{z}^\top, -\mathbf{z}^\top)$ for some $\mathbf{z} \in \mathbb{R}^K$. Let $i = \operatorname{argmax}_{i \in [K]} |z_i|$. By choosing $\mathbf{q} = \mathbf{e}_i$ or $\mathbf{q} = \mathbf{e}_{K+i}$, we can guarantee that

$$\min_{\mathbf{q}} \mathbb{E}\left[\mathbf{p}^*(\mathbf{R})^\top \mathbf{R}\mathbf{q} - \overline{\mathbf{p}}^\top \mathbf{R}\mathbf{q}\right] \leq -\|\mathbf{z}\|_\infty \ ,$$

i.e.,

$$\max_{\mathbf{p}^*} \min_{\mathbf{q}} \mathbb{E}\left[\mathbf{p}^*(\mathbf{R})^\top \mathbf{R}\mathbf{q} - \overline{\mathbf{p}}^\top \mathbf{R}\mathbf{q}\right] \leq 0 \ .$$

The value of zero is attained for $\mathbf{p}^*(\mathbf{R})^\top = \left(\frac{1}{2} + \gamma x_I,\ \frac{1}{2} - \gamma x_I\right)$.

Step 2 *If $\gamma > \varepsilon(m)$ then the algorithm* ALG *can be used to obtain an estimator $\hat{I}$ which takes as an input a sample of size m from $\mathcal{D}_I$ (for any $I \in [K]$), and has an error bounded by $\Pr[\hat{I} \neq I] \leq 1/2$, where the probability is over the draw of the sample.*

For any $i \in [K]$, consider plugging $\mathbf{e}_i$ or $\mathbf{e}_{K+i}$ for $\mathbf{q}$ in the statement of Step 1, to obtain $-|\hat{\mu}_i - \mu_i| \geq -\varepsilon(m)$ or equivalently $|\hat{\mu}_i - \mu_i| \leq \varepsilon(m)$. Since this holds for all $i \in [K]$, we obtain $\|\hat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_\infty \leq \varepsilon(m)$.

If $\gamma > \varepsilon(m)$, then $\|\hat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_\infty < \gamma$. Since $\boldsymbol{\mu} = 2\gamma \mathbf{e}_I$, we obtain that $\hat{\mu}_I > \gamma$ and $\hat{\mu}_i < \gamma$ for $i \neq I$, i.e., we will have our estimator $\hat{I} = \operatorname{argmax}_i \hat{\mu}_i$ provided that we can calculate $\hat{\boldsymbol{\mu}}$. That can be done explicitly as

$$\hat{\boldsymbol{\mu}}^\top = \frac{1}{2^K} \sum_{\mathbf{x} \in \{-1,1\}^K} \hat{\mathbf{p}}(\mathbf{R}(\mathbf{x}))^\top \begin{pmatrix} 1 \\ -1 \end{pmatrix} \mathbf{x}^\top \ .$$

Step 3 *Assume that I is chosen uniformly at random from $[K]$. There is an absolute constant c, such that any estimator $\hat{I}$ which takes as an input a sample of size m from $\mathcal{D}_I$ (for any $I \in [K]$), has the average error at least*

$$\Pr[\hat{I} \neq I] \geq 1 - (mc\gamma^2/4 + \ln 2)/\ln K$$

where the probability is over the draw of I followed by the draw of the sample.

We prove the statement by Generalized Fano Method (Lemma 7 in Appendix B). In our case, the estimator $\hat{I}$ takes a sample from $\mathcal{D}_I^m$, so we need to bound $\mathrm{KL}(\mathcal{D}_i^m \| \mathcal{D}_j^m)$ for $i \neq j$. We begin by bounding $\mathrm{KL}(\mathcal{D}_i \| \mathcal{D}_j)$ for $i \neq j$. Let $p = \frac{1}{2} + \gamma$ be the probability that $y = x_i$ for $(\mathbf{x}, y) \sim \mathcal{D}_i$. Then:

$$\mathrm{KL}(\mathcal{D}_i \| \mathcal{D}_j) = \Pr_{\mathbf{x} \sim \mathcal{D}_i}[x_i \neq x_j] \left(p \ln \left(\frac{p}{1-p} \right) + (1-p) \ln \left(\frac{1-p}{p} \right) \right) = \frac{1}{2} f(p)$$

where $f(p)$ is the KL divergence between the distribution $(p, 1-p)$ and $(1-p, p)$. Note that $f\left(\frac{1}{2}\right) = f'\left(\frac{1}{2}\right) = 0$, and there is a constant c such that $f''(p) \leq c$ for $\left|p - \frac{1}{2}\right| = \gamma \leq \frac{1}{4}$. From the Taylor expansion of f at $\frac{1}{2}$ we therefore obtain $\mathrm{KL}(\mathcal{D}_i \| \mathcal{D}_j) \leq c\gamma^2/4$. Combining with the identity $\mathrm{KL}(\mathcal{D}_i^m \| \mathcal{D}_j^m) = m\mathrm{KL}(\mathcal{D}_i \| \mathcal{D}_j)$, we then have the proof of Step 3 by Generalized Fano Method.

Finishing the proof Taking $\gamma > \varepsilon(m)$ and combining Steps 2 and 3, we obtain

$$mc\gamma^2/4 + \ln 2 \geq (\ln K)/2$$

and hence

$$\gamma \geq \sqrt{\frac{4}{mc}\left(\frac{\ln K}{2} - \ln 2\right)} \geq c' \sqrt{\frac{\ln K}{m}}$$

for a suitable absolute constant c' since $K \geq 5$. Since the above holds for all $\gamma > \varepsilon(m)$, taking the limit $\gamma \to \varepsilon(m)$, we obtain the statement of the lemma. ■

7. Conclusions and open questions

In this paper, we have formalized the problem of learning with multiple objectives as *robust multi-objective learning*. We have proposed an algorithm that solves this problem by combining `Hedge` with a carefully crafted online-to-batch reduction. Our algorithm is able to compete with the set of *all (measurable) decision rules* with regret that is independent of the number of actions, and only scales as a logarithm of the number of objectives. We also proved a matching lower bound.

We have analyzed a stochastic batch setting, where our algorithm can compete with all (measurable) decision rules, and an online adversarial setting, where our algorithm can compete with the mentor. A natural question open for future research is whether the stronger regret guarantees can be generalized to the online stochastic setting, or even to the adversarial setting (batch or online).

Another open question is whether our regret bounds can be improved if we assume that the mentor is rational, i.e., that it optimizes a specific (albeit unknown) combination of objectives.

In this paper, the aggregate utility from a given tuple of objectives is a convex combination of the objectives. It is interesting to consider other well-motivated classes of aggregate utility functions, for example convex combinations under constraints.

Acknowledgments. Much of this research was conducted while A. Badanidiyuru and R. Schapire were visiting Microsoft Research. Support was also provided by NSF grant #1016029.

References

Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *ICML*, 2004.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, August 1997.

Paul E. Green and V. Srinivasan. Conjoint analysis in marketing: New developments with implications for research and practice. *Journal of Marketing*, 54(4):3–19, October 1990.

Nathan D. Ratliff, J. Andrew Bagnell, and Martin Zinkevich. Maximum margin planning. In *ICML*, pages 729–736, 2006.

Umar Syed and Robert Schapire. A game-theoretic approach to apprenticeship learning. In *Advances in Neural Information Processing Systems 20*, NIPS 2007, pages 1449–1456, 2008.

Kenneth Train. *Discrete Choice Methods with Simulation*. Cambridge University Press, New York, NY, 2nd edition, 2009.

Bin Yu. Assouad, Fano and Le Cam. *Festschrift in Honor of L. Le Cam on his 70th Birthday*, 1993.

Brian D. Ziebart, Andrew L. Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, pages 1433–1438, 2008.

Appendix A. Application to apprenticeship learning

As noted earlier, our model is a generalization of the apprenticeship learning framework of Abbeel and Ng (2004) as subsequently extended by Syed and Schapire (2008). Here, we briefly describe how their framework and algorithm are essentially a special case of ours.

In this setting, an agent called an *apprentice* is allowed to observe the actions of a so-called *expert* in a stochastic environment modeled as a Markov Decision Process (MDP). We assume that both the apprentice and the expert have access to the states, actions, and transition function of the MDP, as well as an assumed initial distribution over start states. However, the MDP has no explicit reward function. Rather, each state σ is associated with a vector of *reward features*, denoted $\boldsymbol{\phi}(\sigma)$. We suppose that the "true" reward function is some convex combination $\mathbf{q}^*$ of the reward features so that the (never directly observed) reward at state σ is $\mathbf{q}^* \cdot \boldsymbol{\phi}(\sigma)$. Although $\mathbf{q}^*$ is unknown, we assume that each of the reward features has known direction, that is, that larger values are always at least as desirable as lower values.

We suppose that the apprentice observes the expert following trajectories through the MDP. On each one of these, an initial state σ_0 is chosen from the initial state distribution D_0, then actions are selected by the expert causing the MDP to progress along a trajectory of states $\boldsymbol{\sigma} = (\sigma_0, \sigma_1, \ldots)$ in the usual fashion. For simplicity, we assume these trajectories are infinite, although, as with most of our assumptions, this certainly can be relaxed. Further, we assume that the expert's behavior induces a distribution $\mathcal{D}$ over such state sequences, and that these trajectories are sampled independently from $\mathcal{D}$.

Given m observed expert trajectories through the MDP, the apprentice's goal is to find a policy π that achieves expected reward at least as great as that achieved by the expert, even without knowing $\mathbf{q}^*$. In particular, the discounted reward attained along a trajectory $\boldsymbol{\sigma}$ with respect to a weighting $\mathbf{q}$ over reward features is

$$v(\boldsymbol{\sigma}, \mathbf{q}) = \sum_{t=0}^{\infty} \gamma^t \mathbf{q} \cdot \boldsymbol{\phi}(\sigma_t) \ .$$

The expert's reward (with respect to the true weighting $\mathbf{q}^*$) is then

$$\mathbb{E}_{\mathcal{D}}\big[v(\boldsymbol{\sigma}, \mathbf{q}^*)\big] \ ,$$

meaning that $\boldsymbol{\sigma}$ is distributed according to $\mathcal{D}$, while the apprentice's is

$$\mathbb{E}_{\pi}[v(\boldsymbol{\sigma}, \mathbf{q}^*)] \ ,$$

meaning that σ_0 is chosen according to D_0, and thereafter, when in state σ_t, action $a_t = \pi(\sigma_t)$ is selected. The apprentice's goal then is to find a policy for which its expected discounted reward is close to, or even exceeds, that of the expert.

We also allow the apprentice to select a mixed policy $\boldsymbol{\psi}$ in which, before taking any action, a (deterministic) policy π is chosen randomly according to the distribution $\boldsymbol{\psi}$, and that one deterministic policy is then followed henceforth. (In fact, such mixed policies can be converted into ordinary randomized policies.)

This problem can be reduced to our relaxed mentor-feedback setting. To do so, we take random expert trajectories $\boldsymbol{\sigma}$ from $\mathcal{D}$ and reduce them to pairs $(\mathbf{R}, \mathbf{s})$. First, we define

$$\mathbf{s} = \sum_{t=0}^{\infty} \gamma^t \boldsymbol{\phi}(\sigma_t) \ ,$$

the cumulative discounted sum of the reward-feature vectors along this particular trajectory. Next, we define the $\mathbf{R}$ whose rows are indexed by deterministic policies π, and whose columns are indexed by the reward features. In particular, we define individual entries of $\mathbf{R}$ as follows:

$$R(\pi, j) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t \phi_j(\sigma_t)\right] \ .$$

Again, expectation here is with respect to a trajectory of states that follow the policy π. Thus, in fact, the same $\mathbf{R}$ is always used. Note that this matrix will have an enormous, but nevertheless finite, number of rows—one for every possible deterministic policy. Even so, our approach will be efficient.

A distribution $\mathbf{q}$ now corresponds directly to a convex combination of reward features. Then, for $\mathbf{s}$ constructed as above, we have

$$\mathbb{E}\left[\mathbf{s}^\top \mathbf{q}\right] = \mathbb{E}_{\mathcal{D}}[v(\boldsymbol{\sigma}, \mathbf{q})] \ .$$

Also, a probability vector $\mathbf{p}$ over rows of $\mathbf{R}$ corresponds to a mixed policy $\boldsymbol{\psi}$, and we have

$$\mathbb{E}\left[\mathbf{p}^\top \mathbf{R}\mathbf{q}\right] = \mathbf{p}^\top \mathbf{R}\mathbf{q} = \mathbb{E}_{\pi \sim \boldsymbol{\psi}}\big[\mathbb{E}_{\pi}[v(\boldsymbol{\sigma}, \mathbf{q})]\big] \ .$$

In the relaxed mentor-feedback setting, we find a vector $\hat{\mathbf{p}} = \hat{\mathbf{p}}(\mathbf{R})$ (which is fixed since $\mathbf{R}$ is fixed) for which $\mathbb{E}\left[\hat{\mathbf{p}}^\top \mathbf{R}\mathbf{q}\right]$ is large relative to $\mathbb{E}\left[\mathbf{s}^\top \mathbf{q}\right]$, for any $\mathbf{q}$. But by the arguments above, we are

actually finding a corresponding mixed policy ψ whose expected discounted reward is large relative to that of the expert, for any weighting $\mathbf{q}$ of the reward features (including the true weighting $\mathbf{q}^*$).

To apply Algorithm 2 of Section 4.2, we need to be able to solve two key computational problems. First, we need to be able to compute $\boldsymbol{\ell}_t$ as in Eq. (7). In our case, $\mathbf{p}_t$ will be concentrated on a single policy π_t, and it can be shown by tracing through definitions that the j-th element of $\mathbf{p}_t^\top \mathbf{R}$ is

$$\mathbb{E}_\pi[v(\boldsymbol{\sigma}, \mathbf{e}_j)] \ .$$

This can be approximated through sampling, or alternatively, can be computed exactly using policy evaluation.

The other problem we need to solve is the computation of $\mathbf{p}_t$ as at Eq. (6). (Note that, since there is only a single context matrix $\mathbf{R}$, $\mathbf{f}_t$ is a constant function that always returns $\mathbf{p}_t$.) Thus, given $\mathbf{q}_t$, we need to find $\mathbf{p}$ which maximizes $\mathbf{p}^\top \mathbf{R}\mathbf{q}_t$. In fact, $\mathbf{q}_t$ defines a weighting over reward features, and $\mathbf{R}\mathbf{q}_t$ then encodes the value of each policy for the single reward induced by the resulting convex combination over reward features. The problem we need to solve, therefore, is that of finding the policy with maximum value when the reward of the MDP is known (as well as all of the other ingredients). But this problem can be solved using standard techniques like value iteration or policy iteration. Thus, although we are conceptually working over an extremely large matrix, all computations can in fact be completed very efficiently.

Appendix B. Generalized Fano Method

The lemma below is a corollary of Lemma 3 of Yu (1993).

Lemma 7 (Generalized Fano Method) *Let I be a random variable uniformly distributed on $[K]$ and let ξ be a random variable with the conditional distribution P_i given $I = i$. Assume that $\mathrm{KL}(P_i \| P_j) \leq \beta$ for all $i, j \in [K]$. Then any estimator $\hat{I}(\xi)$ satisfies*

$$\Pr_{I,\xi}\big[\hat{I}(\xi) \neq I\big] \geq 1 - \frac{\beta + \ln 2}{\ln K} \ .$$