

Imitation Learning via Focused Satisficing

Rushit N. Shah¹, Nikolaos Agadakos¹, Synthia Sasulski¹, Ali Farajzadeh¹,
Sanjiban Choudhury² and Brian Ziebart¹

¹Department of Computer Science, University of Illinois Chicago

²Department of Computer Science, Cornell University

{rshah231, nagada2, afaraj5, bziebart}@uic.edu, synthiasasulski@gmail.com, sanjibanc@cornell.edu

Abstract

Imitation learning often assumes that demonstrations are close to optimal according to some fixed, but unknown, cost function. However, according to *satisficing theory*, humans often choose *acceptable* behavior based on their personal (and potentially dynamic) levels of *aspiration*, rather than achieving (near-) optimality. For example, a lunar lander demonstration that successfully lands without crashing might be acceptable to a novice despite being slow or jerky. Using a margin-based objective to guide deep reinforcement learning, our **focused satisficing** approach to imitation learning seeks a policy that surpasses the demonstrator’s aspiration levels—defined over trajectories or portions of trajectories—on unseen demonstrations *without explicitly learning those aspirations*. We show experimentally that this focuses the policy to imitate the highest quality (portions of) demonstrations better than existing imitation learning methods, providing much higher rates of guaranteed acceptability to the demonstrator, and competitive true returns on a range of environments.

1 Introduction

Hand-engineered policies and reinforcement-learned policies from hand-specified cost functions often fail to perform adequately in complicated tasks of interest (e.g., self-driving). Prevalent imitation learning approaches [Osa *et al.*, 2018] address this issue either by directly mimicking human demonstrations via behavioral cloning [Pomerleau, 1991] or by estimating reward functions that rationalize demonstrator behavior [Ng and Russell, 2000; Abbeel and Ng, 2004]—both under the assumption that the demonstrator is (near) optimal. The many advantages autonomous systems have over human actors, including faster reaction time [Whelan, 2008], more precise control [Ladha *et al.*, 2023], increased rationality, and lossless memory [Miller, 1956], can violate this assumption and lead to potential value misalignment [Amodei

Proofs of theorems and additional implementation details are available in the extended version of this paper at <https://arxiv.org/abs/2505.14820>.

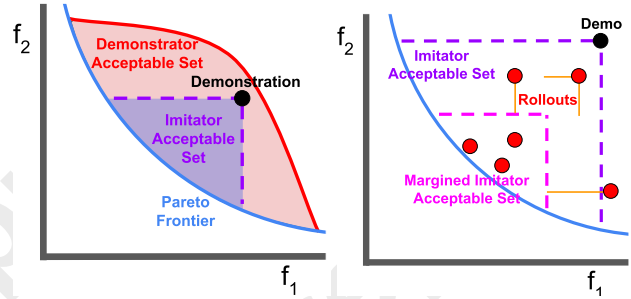


Figure 1: Left: Pareto-dominating in the cost function bases (f_1, f_2) of acceptable behavior (purple: *imitator acceptable set*) guarantees the imitator is acceptable to the demonstrator (red: *demonstrator acceptable set*). Right: The subdominance (orange lines) measures how far imitator trajectory rollouts are from guaranteed acceptance (by a margin).

et al., 2016] between demonstrator and imitator. New perspectives are needed to train more capable imitation learners from less capable demonstrators without supplemental annotations [Christiano *et al.*, 2017; Brown *et al.*, 2019; Rafailov *et al.*, 2024] or assuming some expert-level demonstrations being available [Tangkaratt *et al.*, 2021].

When faced with challenging decision tasks, *satisficing theory* [Simon, 1956] suggests that demonstrators produce behavior that is *acceptable* rather than (near) optimal. By viewing imitation learning through this lens, we aim for imitator behavior that is similarly *acceptable* to the demonstrator, despite never knowing the demonstrator’s precise acceptability criteria (Figure 1, left)—working instead with an assumed class of cost functions that defines it. To pursue this aim, we develop **Minimally Subdominant Focused Imitation (MinSubFI)**, which employs the subdominance [Ziebart *et al.*, 2022], a margin-based measure of *insufficiency* (i.e., the distance from guaranteeing imitator-acceptability by a margin), as a training objective for policy gradient optimization (Figure 1, right). This produces policies that are maximally acceptable rather than reward-maximizing. Compared to existing inverse reward learning methods [Brown *et al.*, 2019; Burchfiel *et al.*, 2016; Wirth *et al.*, 2017; Wu *et al.*, 2019; Chen *et al.*, 2020; Zhang *et al.*, 2021], which are highly reliant on an estimated scalar reward function to guide re-

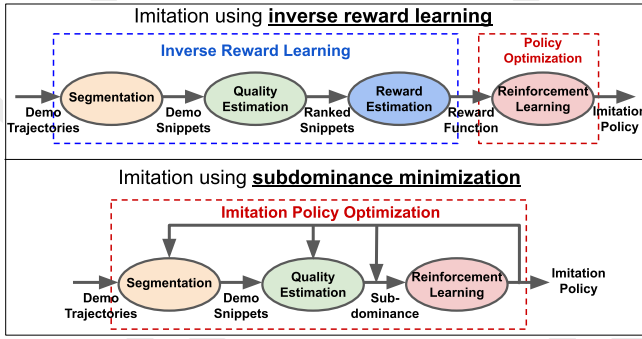


Figure 2: Existing reward-based imitation methods, e.g., TREX [Brown *et al.*, 2019], seek to outperform demonstrations using a pipeline of engineered components (top) to first segment trajectories into “snippets,” and to ultimately estimate a reward function that is then optimized using reinforcement learning. Our approach (bottom) uses the **subdominance** as the reinforcement learning objective, which is defined by the relative performance of the imitator compared to the demonstrations in each cost feature. This effectively uses feedback from the learned imitator policy to guide additional reinforcement learning without an explicit reward function.

inforcement learning (e.g., using the pipeline of engineered components in Figure 2, top), our approach more directly optimizes the imitator’s policy, enabling it to:

- Learn context-sensitive policies without learning context-sensitive cost functions;
- Ignore less optimal demonstrations without requiring explicit noise modeling;
- Automatically select and learn from portions of trajectories (i.e., snippets) of high quality; and
- Provide generalization guarantees for changing acceptability (e.g., due to skill improvement or fatigue).

Under the MinSubFI objective, many of the same engineered components of existing approaches (Figure 2, top) are jointly optimized in a unified manner (Figure 2, bottom). We evaluate the benefits of MinSubFI on imitation learning tasks using human and synthetic demonstrations with both engineered and learned cost features.

2 Satisficing Demonstrations & Policy Gradient Subdominance Minimization

We now formally recast imitation learning through the lens of *satisficing theory*. Under this perspective, policies are learned from demonstrations that are *acceptable*, according to an unknown acceptability set, rather than *near-optimal*. We broadly define this notion of acceptability over trajectories and trajectory “snippets” (i.e., portions of trajectories), and develop new imitation learning methods that are designed to be performant with respect to the demonstrators’ unknown acceptability sets in both theory and practice.

2.1 Imitation Learning Problem Setting

We consider the imitation learning [Osa *et al.*, 2018] task of producing a policy $\hat{\pi}$ based on demonstrated trajectories of states and actions, $\tilde{\xi} = (\tilde{s}_1, \tilde{a}_1, \tilde{s}_2, \dots, \tilde{s}_T)$. Demonstrations are produced from a task-indexed Markov decision process

(MDP), $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \{\tau_i\}, C)$, characterized by states \mathcal{S} , actions \mathcal{A} , state transition probability distributions $\tau_i : \mathcal{S} \times \mathcal{A} \rightarrow \Delta_{\mathcal{S}}$ (with Δ representing a probability simplex), and a cost function $C : \mathcal{S} \rightarrow \mathbb{R}_{\geq 0}$. The state transition probability distribution is defined for $s = a = \emptyset$ to provide an initial state distribution. Each state transition probability distribution, τ_i , corresponds to a different task i that shares the same state-action space, but may have different initial states, different absorbing (goal) states, or different dynamics more generally. We use $\tilde{\xi}_{i,j}$ to denote the j^{th} demonstration for the i^{th} task, $\tilde{\Xi}$ to denote the set of all demonstrations, and $\tilde{\Xi}_i$ to denote the set of demonstrations corresponding to task i . The cost/reward function is unavailable to the imitator (providing at most $\mathcal{M} \setminus C$), distinguishing imitation learning from (offline) reinforcement learning [Levine *et al.*, 2020].

2.2 Satisficing Perspective of Demonstrations

According to satisficing theory [Simon, 1956], when faced with challenging decision tasks, humans tend to prioritize behaviors that are acceptable to them rather than striving for optimality. This implies that demonstrated behavior is selected to be *acceptable*, according to some aspirational criteria of the demonstrator, rather than being (near) *optimal*.

Definition 1. Trajectory ξ **satisfices** (or is **acceptable**) for a particular **aspiration**, defined by (\mathbf{w}, ν, t, t') if and only if it is less costly than the aspirational threshold ν evaluated using the cost function parameterized by \mathbf{w} : $\text{cost}_{\mathbf{w}}(\xi_{t:t'}) < \nu$. It **satisfices** the **aspiration/acceptability set** $\Omega = \{(\mathbf{w}, \nu, t, t')\}$, i.e., $\xi \in \text{Satisf}_{\Omega}$, if and only if ξ satisfices each aspiration in Ω .

Note that the aspiration set can be context-dependent and vary for each demonstration. For example, it may change with the growing experience (or fatigue) of the demonstrator, or based on available side information (e.g., the weather conditions when controlling a vehicle). Additionally, each aspiration criteria can be defined over a portion (i.e., a “snippet”) $\xi_{t:t'}$ of the full trajectory $\xi_{1:T}$.

Aspiration sets—and their relationships to available contextual information—are generally unknown. Our aim is not to learn them explicitly. Instead, we seek a policy that produces trajectories $\xi \sim \pi \times \tau$, with **maximal probability of acceptance**, $P(\xi \in \text{Satisf}_{\tilde{\xi}})$, for $\tilde{\xi}$ ’s implicit satisfaction set.

A **key question** from this satisficing perspective is: *do existing imitation learners provide acceptability guarantees with respect to (unknown) demonstrator acceptability sets?*

Behavioral cloning approaches [Pomerleau, 1991] directly estimate a (stochastic) policy $\pi_{\theta} : \mathcal{S} \rightarrow \Delta_{\mathcal{A}}$ from demonstrated state-action pairs, (s_t, a_t) . The simplicity of this approach allows the full range of supervised machine learning techniques to be employed to estimate the policy. For example, generative adversarial imitation learning (GAIL) [Ho and Ermon, 2016a] employs a discriminator to distinguish between human and automated action choices, and guide policy learning to minimize any differences. Unfortunately, behavioral cloning methods cannot outperform the demonstration policy beyond being Bayes optimal for a predictive loss that may not align with the acceptability set

cost function(s). This prevents behavioral cloning methods from providing satisficing guarantees.

Inverse reinforcement learning [Kalman, 1964] estimates the cost function $C(s)$ that explains or rationalizes demonstrations (making them near optimal). A cost function linear in a set of state features, $\mathbf{f} : \mathcal{S} \rightarrow \mathbb{R}^K$, or state-action features, $\mathbf{f} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}^K$ is commonly assumed [Ng and Russell, 2000]. Under this assumption, **feature matching** [Abbeel and Ng, 2004] guarantees the estimated policy $\hat{\pi}$ has expected cost under the demonstrator’s unknown fixed cost function weights $\tilde{\mathbf{w}} \in \mathbb{R}^K$ equal to the average of the demonstration policies π if the expected feature counts match:

$$\begin{aligned} \mathbb{E}_{\substack{\tau_i \sim \tilde{\pi}, \\ \xi \sim \pi \times \tau_i}} [f_k(\xi)] &= \frac{1}{|\tilde{\Xi}|} \sum_{\tilde{\xi}_{i,j} \in \tilde{\Xi}} f_k(\tilde{\xi}_{i,j}), \forall k \\ \implies \mathbb{E}_{\substack{\tau_i \sim \tilde{\pi}, \\ \xi \sim \pi_{\theta} \times \tau_i}} [C_{\tilde{\mathbf{w}}}(\xi)] &= \frac{1}{|\tilde{\Xi}|} \sum_{\tilde{\xi}_{i,j} \in \tilde{\Xi}} C_{\tilde{\mathbf{w}}}(\tilde{\xi}_{i,j}), \end{aligned} \quad (1)$$

where $f_k(\xi) \triangleq \sum_{s_t, a_t \in \xi} f_k(s_t, a_t)$ and $C_{\tilde{\mathbf{w}}}(\xi) \triangleq \sum_{s_t, a_t \in \xi} C_{\tilde{\mathbf{w}}}(s_t, a_t)$. This feature-matching constraint (1) can be enforced using a potential term measuring the demonstration $\tilde{\xi}$ ’s suboptimality relative to induced behavior ξ . Closer to our approach, game-theoretic apprenticeship learning [Syed and Schapire, 2007] assumes the sign of the linear cost function’s weights are known and produces a policy that is guaranteed to be better in expectation than the demonstration average under worst-case weights.

Unfortunately, matching the demonstrator’s unknown expected rewards (or outperforming on average) only guarantees that the imitator achieves the aspiration level in expectation. If the demonstrators’ aspirations depend on context that is not incorporated in the learned cost function, better levels of aspiration will not be guaranteed. Thus, inverse reinforcement learning does not provide useful guarantees for per-demonstration satisficing; it is not a discriminative enough policy optimization method.

2.3 Subdominance Minimization and Satisficing

The subdominance measures how far trajectory ξ is from Pareto-dominating (i.e., smaller in each cost feature dimension than) a demonstrated trajectory $\tilde{\xi}$ by a margin (Figure 1, right). It has been previously employed for inverse optimal control to make the optimal trajectory induced by learned linear cost function weights $\mathbf{w} \in \mathbb{R}_{\geq 0}^K$, outperform sets of task-specific demonstrations $\{\tilde{\xi}_i\}$ [Ziebart *et al.*, 2022]:

$$\begin{aligned} \min_{\mathbf{w} \geq 0} \min_{\alpha \geq 0} \sum_{i=1}^N \frac{|\tilde{\Xi}_i|}{|\tilde{\Xi}|} \text{subdom}_{\alpha}(\xi_i^*(\mathbf{w}), \tilde{\Xi}_i) + \frac{\lambda}{2} \|\alpha\|, \text{ where:} \\ \text{subdom}_{\alpha}(\xi, \tilde{\Xi}) = \frac{1}{|\tilde{\Xi}|} \sum_{\tilde{\xi} \in \tilde{\Xi}} \sum_k \underbrace{\left[\alpha_k (f_k(\xi) - f_k(\tilde{\xi})) + 1 \right]_+}_{\text{(feature } k \text{) subdom}_{\alpha_k}^k(\xi, \tilde{\xi})}, \quad (2) \\ \underbrace{\hspace{10em}}_{\text{(aggregated) subdom}_{\alpha}(\xi, \tilde{\Xi})} \end{aligned}$$

with $[x]_+ \triangleq \max(x, 0)$ as the hinge function, and trajectory cost features $\mathbf{f} : \Xi \rightarrow \mathbb{R}_{\geq 0}^K$. Other variants include defining the subdominance using relative cost features, $\text{relsubdom}_{\alpha_k}^k(\xi, \tilde{\xi}) \triangleq \left[\alpha_k \left(\frac{f_k(\xi)}{f_k(\tilde{\xi})} - 1 \right) + 1 \right]_+$, and/or

aggregating over feature dimensions using maximization, $\text{subdom}_{\alpha}(\xi, \tilde{\xi}) \triangleq \max_k \text{subdom}_{\alpha_k}^k(\xi, \tilde{\xi})$ [Ziebart *et al.*, 2022]. Like support vector machines [Vapnik and Chapelle, 2000], only a subset of *support demonstrations*, $\tilde{\Xi}_i^{\text{SV}_k}(\xi) \subseteq \tilde{\Xi}_i$, for each task i and feature k , actively influence θ :

$$\tilde{\xi} \in \tilde{\Xi}_i^{\text{SV}_k}(\xi) \iff f_k(\xi) + \frac{1}{\alpha_k} \geq f_k(\tilde{\xi}). \quad (3)$$

For notational convenience, when ξ is indexed (e.g., by (i, j) as $\xi_{i,j}$), we denote this resulting support vector set for all demonstrations of task i as $\tilde{\Xi}_i^{\text{SV}_k}$ for feature k . Unfortunately, optimal control is impractical for many realistic imitation learning problems of interest. Additionally, it makes the learned cost/reward function (Fig. 2) a bottleneck that can prevent the imitation policy from better fitting to (or outperforming) demonstrations.

However, subdominance has an important relationship to satisficing (Theorem 2): if it can be lowered to zero, acceptability of the imitator’s behavior is guaranteed under mild cost function assumptions (positive linear functions of monotonic transformations of cost features).

Theorem 2. *A trajectory ξ with zero subdominance with respect to demonstration $\tilde{\xi}$ implies that the demonstration’s corresponding aspiration set (for full trajectory aspiration functions/thresholds) is satisfied by ξ : $(\exists \alpha \succ \mathbf{0}, \text{subdom}_{\alpha}(\xi, \tilde{\xi}) = 0) \implies \xi \in \text{Satisf}_{\tilde{\xi}}$.*

Proof of Theorem 2. Zero subdominance implies Pareto dominance of the imitator cost feature over the demonstrator cost features, which implies that the imitator is acceptable under any cost functions defining the demonstrator’s acceptable set.

$$\forall \alpha \succ \mathbf{0}, \text{subdom}(\xi, \tilde{\xi}) = 0 \implies \mathbf{f}(\xi) \preceq \mathbf{f}(\tilde{\xi}) \quad (4)$$

$$\implies \forall \theta \succeq \mathbf{0}, \text{cost}_{\theta}(\xi) \leq \text{cost}_{\theta}(\tilde{\xi}) \quad (5)$$

$$\implies \xi \in \text{satisf}_{\tilde{\xi}} \quad (6)$$

□

Note that the additional margin incorporated in the subdominance plays an important role in providing generalization guarantees for the imitator (Theorem 8) that do not exist if the imitator simply matches the features of the demonstrator on training examples.

As an illustrative example, consider two cost features for lunar lander depicted in Figure 3: its x offset from the landing pad and its angular velocity ω . An imitator trajectory ξ which lands more precisely (i.e., smaller x offset) and more smoothly (i.e., smaller angular velocity ω) than a demonstration $\tilde{\xi}$, by definition has zero subdominance. Such a trajectory would also be part of the margined imitator acceptable set (Figure 1, right) and hence satisfies demonstration $\tilde{\xi}$.

Thus, our objective is to better minimize the subdominance by finely optimizing over a more flexible class of policies. To generalize to unseen data, we additionally seek a margin of improvement over the demonstrator, i.e., subdom_{α} , throughout our formulation. With this added margin, the subdominance is a convex function (in trajectory features) that upper

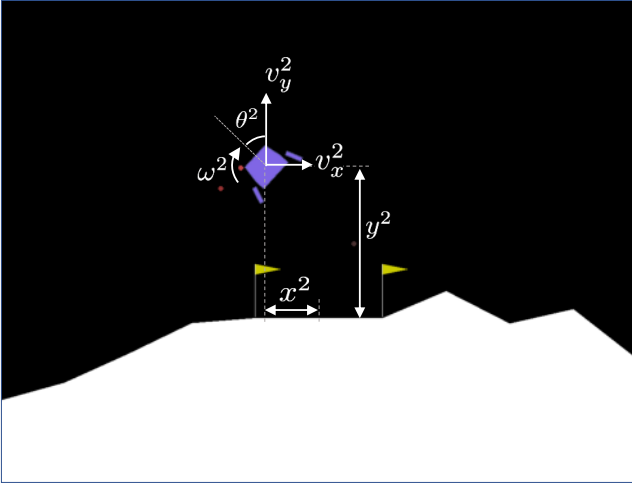


Figure 3: Examples of lunarlander cost features, which are computed easily from the environment’s observation vector.

bounds the Satisf_ξ non-membership, measuring how far the trajectory is from being guaranteed to satisfy the demonstrator’s aspirations by a margin.

2.4 Snippet-focused Subdominance

To enable snippet-level satisficing (in addition to the trajectory-level satisficing guaranteed by Theorem 2), we define a snippet-focused variant of the subdominance. We select snippet pairs that maximize subdominance:

$$\text{subdom}_\alpha^{\text{snip}}(\xi, \tilde{\xi}) = \max_{(\xi_{\text{sn}}, \tilde{\xi}_{\text{sn}}) \in \mathbb{S}(\xi, \tilde{\xi})} \text{subdom}_\alpha(\xi_{\text{sn}}, \tilde{\xi}_{\text{sn}}), \quad (7)$$

where \mathbb{S} extracts snippet pairs from the full trajectories. This focuses imitation on high-quality snippets (with high subdominance) even if the larger trajectory they come from is of lower quality (and low or zero subdominance because it is easy for the imitator to outperform as a whole). The design of \mathbb{S} provides a great deal of flexibility for defining snippets based on states and/or time steps.

2.5 Subdominance for Stochastic Policies

We next expand the subdominance definition to incorporate stochastic action selection from policy π :

$$\text{subdom}_\alpha(\pi, \tilde{\Xi}) = \mathbb{E}_{\xi \sim \pi \times \tau} [\text{subdom}_\alpha(\xi, \tilde{\Xi})]. \quad (8)$$

Definition 3. The minimally subdominant stochastic policy $\pi_\theta : \mathcal{S} \rightarrow \Delta_{\mathcal{A}}$ minimizes the expected subdominance of the minimum cost trajectory, $\xi^*(\pi_\theta)$ induced by the weights θ of policy π , with respect to the set of demonstration trajectories ξ_i using hinge slopes α :

$$\min_{\theta} \min_{\alpha \geq 0} \sum_{\text{task } i} \frac{|\tilde{\Xi}_i|}{|\tilde{\Xi}|} \text{subdom}_\alpha(\pi_\theta, \tilde{\Xi}_i) + \frac{\lambda_\alpha}{2} \|\alpha\| + \frac{\lambda_\theta}{2} \|\theta\|. \quad (9)$$

This optimization seeks hinge loss slopes α and a policy π_θ that both minimize the subdominance. Naively approaching this optimization can be problematic, since $\alpha = 0$ corresponds to a degenerate local optimum. However, the optimal

α values for a policy achieving at least the average feature counts of the demonstrations are not degenerate. This suggests bootstrapping from an initial policy estimate when minimizing α values or restricting α values above zero.

2.6 Subdominance Policy Gradient Optimization

To efficiently optimize the objective outlined in Definition 3, we consider policy gradient algorithms. We leverage Theorem 4 for the computation of the policy gradient using the trajectory-based subdominance as a reinforcement signal. Corollary 5 provides a per-state decomposition of the subdominance, making a wide range of existing policy gradient methods applicable that assign credit in a temporally consistent manner.

Theorem 4. Policy π_θ ’s subdominance with respect to demonstration set $\{\tilde{\Xi}_i\}$ has policy gradient:

$$\begin{aligned} \nabla_\theta \sum_i \frac{|\tilde{\Xi}_i|}{|\tilde{\Xi}|} \mathbb{E}_{\xi_i \sim \pi_\theta \times \tau_i} [\text{subdom}_\alpha(\xi_i, \tilde{\Xi}_i)] \\ = \sum_i \frac{|\tilde{\Xi}_i|}{|\tilde{\Xi}|} \mathbb{E}_{\xi_i \sim \pi_\theta \times \tau_i} \left[\text{subdom}_\alpha(\xi_i, \tilde{\Xi}_i) \sum_{(s,a) \in \xi_i} \nabla_\theta \log \pi_\theta(a|s) \right], \end{aligned}$$

For a set of single trajectory samples, $\xi_i \sim \pi_\theta \times \tau_i$, for each task i , the policy parameters θ can be (stochastically) updated via gradient descent: $\theta \leftarrow \theta + \eta \sum_i \sum_{(a_t, s_t) \in \xi_i} G_t \nabla_\theta \log \pi_\theta(a_t|s_t)$, where G_t is any function of the full or future expected subdominance, $\text{subdom}_\alpha(\xi_i, \tilde{\Xi}_i)$, such as the Q -value, the advantage estimate, or the trajectory return [Sutton et al., 1999].

The proof for Theorem 4 is provided in our supplementary material. We now present a state-based decomposition of trajectory level subdominance. Subdominance is computed at the final state to determine which features contribute to it, and the contribution of each state-action pair to the total trajectory subdominance is the calculated.

Corollary 5. The absolute and relative subdominances for a trajectory ξ , with respect to a set of demonstrations Ξ can be further expanded as:

$$\begin{aligned} \text{subdom}_\alpha(\xi, \tilde{\Xi}) &= \sum_{s_t \in \xi, k} \left(\frac{\tilde{C}_{\xi, \tilde{\Xi}}^k}{|\xi|} + \tilde{C}_{\xi, \tilde{\Xi}}^k \alpha_k f_k(s_t) - \frac{\alpha_k \tilde{f}_{k, \xi, \tilde{\Xi}}^{\text{abs}}}{|\xi| |\tilde{\Xi}|} \right); \\ \text{relsubdom}_\alpha(\xi, \tilde{\Xi}) &= \sum_{s_t \in \xi, k} \left(\frac{\tilde{C}_{\xi, \tilde{\Xi}}^k (1 - \alpha_k)}{|\xi|} + \frac{\alpha_k f_k(s_t) \tilde{f}_{k, \xi, \tilde{\Xi}}^{\text{rel}}}{|\tilde{\Xi}|} \right), \end{aligned}$$

where $\tilde{C}_{\xi, \tilde{\Xi}}^k = \frac{|\tilde{\Xi}^{\text{sv}_k}(\xi)|}{|\tilde{\Xi}|}$, $\tilde{f}_{k, \xi, \tilde{\Xi}}^{\text{abs}} = \sum_{\tilde{\xi} \in \tilde{\Xi}^{\text{sv}_k}(\xi)} \sum_{s'_t \in \tilde{\xi}} f_k(s'_t)$, and $\tilde{f}_{k, \xi, \tilde{\Xi}}^{\text{rel}} = \sum_{\tilde{\xi} \in \tilde{\Xi}^{\text{sv}_k}(\xi)} \left(\sum_{s'_t \in \tilde{\xi}} f_k(s'_t) \right)^{-1}$.

This decomposition enables state-of-the-art reinforcement learning algorithms [Schulman et al., 2017] that assign credit to actions in a causally consistent manner (i.e., only future returns influence an action’s updates) to be employed. Further flexibility is gained via the choice of policy representation.

2.7 Subdominance Policy Gradient Algorithms

Algorithm 1 outlines our approach for optimization. For each task (i), a trajectory is rolled out by sampling from the current learned policy (Line 2). The cost features of the sampled trajectory and the demonstrated trajectory are compared to determine which dimensions the sampled trajectory does not sufficiently outperform the demonstration, and are thus support vectors (Line 4). Here, the α values defining margin slopes (Eq. (3)) can either be optimized numerically (e.g., using stochastic gradient descent) or analytically [Memarrast *et al.*, 2023]. A policy update is then employed to reduce the subdominance (Line 7).

Algorithm 1 Online subdominance policy gradient

```

1: while  $\theta$  not converged do
2:   Sample a set of  $M$  trajectories  $\Xi_i = \{\xi_i^{(m)}\}_{m=0}^M$  from
     policy  $\pi_\theta \times \tau_i$  for each task  $i$ 
3:   for each  $\xi_i^{(m)} \in \Xi_i$  do
4:     Find support vectors  $\tilde{\Xi}_{i,m}^{\text{SV}_k}$  (and  $\alpha$ ) given  $\xi_i^{(m)}$ 
5:     Compute loss  $\mathcal{L}(\xi_i^{(m)}) = \text{subdom}_\alpha(\xi_i^{(m)}, \tilde{\Xi}_i)$ 
6:   end for
7:   Update  $\theta$  via policy gradient update rule on  $\mathcal{L}(\xi_i^{(m)})$ 
8: end while

```

For snippet-based optimization (Eq. (7)), the snippet extractor \mathbb{S} produces snippet pairs as support vector candidates. This can uncover supporting snippets from high-quality portions of trajectories that are lower quality overall (and not supporting trajectories). The highest subdominance snippets are then used to compute subdominance losses (Line 5) and to perform policy gradient updates (Line 7).

Algorithm 2 describes a practical approach for snippet-based optimization. We consider the snippet generator, \mathbb{S} , that produces snippets of various lengths starting from states that coincide between the rollout and the demonstration. Unfortunately, demonstrations and rollouts may share very few states (apart from the initial state) in practice. To more effectively uncover supporting snippets, we roll out trajectories from randomly-chosen states along the demonstrator trajectory, and compare these to snippets from the demonstration that begin from that state.

Algorithm 2 Snippet-based subdominance policy gradient

```

1: while  $\theta$  not converged do
2:   Sample demonstration  $\tilde{\xi}_j$  from demonstration set  $\tilde{\Xi}$ 
3:   Sample state  $s_t^{(j)} \sim \tilde{\xi}_j$  such that  $0 < t < |\tilde{\xi}_j|$ 
4:   Set  $s_0 \leftarrow s_t^{(j)}$ 
5:   Sample imitator trajectory  $\xi \sim \pi_\theta(\cdot | s_0)$ 
6:   Find largest support vector snippets pair(s) ( $\xi_{\text{snip}}, \tilde{\xi}_{\text{snip}}$ ) (and
      $\alpha$ ) from snippet pair candidates  $\mathbb{S}(\xi, \tilde{\xi}_{t:T})$ 
7:   Compute loss  $\mathcal{L}(\xi_{\text{snip}}) = \text{subdom}_\alpha(\xi_{\text{snip}}, \tilde{\xi}_{\text{snip}})$ 
8:   Update  $\theta$  via any policy gradient update rule on  $\mathcal{L}(\xi_{\text{snip}})$ 
9: end while

```

When deploying or simulating a policy is expensive, **offline policy gradient methods** that are based entirely on the

set of demonstrated trajectories can instead be employed.

Corollary 6. *Offline policy gradient (MinSubFI_{OFF}) employs importance weighting to estimate the gradient for online subdominance minimization from available demonstrations:*

$$\theta \leftarrow \theta + \eta \sum_{i, \tilde{\xi}_{i,j} \in \tilde{\Xi}_i} \tilde{r}_{\theta, \tilde{\pi}}^{(i,j)} \text{subdom}_\alpha(\tilde{\xi}_{i,j}, \tilde{\Xi}_i) \sum_{(s,a) \in \tilde{\xi}_{i,j}} \nabla_\theta \log \pi_\theta(a|s), \quad (10)$$

where $\tilde{r}_{\theta, \tilde{\pi}}^{(i,j)} = \frac{\pi_\theta(\tilde{\xi}_{i,j})}{\tilde{\pi}(\tilde{\xi}_{i,j})}$ is the importance ratio, and $\tilde{\pi}$ is an estimate of the demonstrator’s policy.

The offline policy gradient method (Corollary 6) is outlined in Algorithm 3 below.

Algorithm 3 Offline, joint stochastic optimization

```

1: Estimate  $\tilde{\pi}_{\text{BC}}$  using behavior cloning on demonstrations  $\tilde{\Xi}$ 
2: while  $\theta$  not converged do
3:   for each  $\tilde{\xi}_{i,j} \in \tilde{\Xi}_i$  do
4:     Find support vectors  $\tilde{\Xi}_{i,j}^{\text{SV}_k}(\alpha_k)$  given  $\tilde{\xi}_{i,j}$ 
5:     for each  $k$  do
6:        $\alpha_k \leftarrow \alpha_k \exp\{-\eta \tilde{r}_{\theta, \tilde{\pi}_{\text{BC}}}^{(i,j)} \sum_{\tilde{\xi}' \in \tilde{\Xi}_{i,j}^{\text{SV}_k}} (f_k(\tilde{\xi}_{i,j}) - f_k(\tilde{\xi}')) +$ 
  

          $\lambda |\tilde{\Xi}| \alpha_k\}$ 
7:     end for
8:     Update  $\theta$  according to Equation (10).
9:   end for
10: end while

```

2.8 Generalization Bound Analysis

We now define the notion of a γ -satisficing stochastic policies and present a generalization bound.

Definition 7. A policy is considered γ -satisficing (or γ -acceptable) for cost features \mathbf{f} and distribution of demonstrated trajectories $P(\tilde{\xi})$, if its trajectories ξ drawn from policy π satisfies with probability at least γ : $P(\xi \in \Omega_{\tilde{\xi}}) \geq \gamma$.

A snippet-based extension of this definition considers $\xi \in \Omega_{\tilde{\xi}}$ if and only if the subdominance is zero for all max-min snippet pairs (Eq. (7)).

Theorem 8. *The policy minimizing the absolute or relative subdominance $\left(\xi^*(\pi_\theta), \tilde{\xi}_i\right)$ (N iid demonstrations) with realizable features that are convex sets has the support vector set $\left\{\tilde{\Xi}_{\text{SV}_k}(\xi^*(\pi_\theta), \alpha_k)\right\}$ and is on average γ -satisficing on the population distribution with: $\gamma = 1 - \frac{1}{N} \left| \bigcup_{k=1}^K \tilde{\Xi}_{\text{SV}_k}(\xi^*(\pi_\theta), \alpha_k) \right|$.*

This bound motivates subdominance minimization for producing demonstrator-acceptable behavior.

2.9 Learning a Cost Feature Representation

Though shaping the imitator’s behavior from demonstrations is much less dependent on a highly-expressive cost model/features under our approach, hand-engineering features can still be a significant burden in many domains. To mitigate this, we propose to learn a set of cost features \mathbf{f}_ψ from pairwise preferences over demonstrations.

Demo Type	Environment	Min	Mean	Max
synthetic	cartpole	10	76	194
	lunarlander	-196	112	284
	hopper	6	939	3441
	halfcheetah	-83	680	1483
	walker	18	968	4293
human	lunarlander	-419	173	303

Table 1: True return statistics of demonstration sets for each environment (100 demonstrations each).

Definition 9. Given pairwise preferences over demonstrations $\tilde{\mathcal{D}} = \{\tilde{\xi}_i \prec \tilde{\xi}_j | \tilde{\xi}_i, \tilde{\xi}_j \in \tilde{\Xi}\}$, and a sufficiently-rich function class \mathcal{F} , a preference-preserving (latent) representation $\mathbf{f}_\psi : \mathcal{S} \rightarrow \mathbb{R}_{\geq 0}^{K'}$ (of dimensionality K') can be learned by minimizing: $\arg \min_{\mathbf{f}_\psi \in \mathcal{F}} \mathbb{E}_{(\tilde{\xi}_i \prec \tilde{\xi}_j) \sim \tilde{\mathcal{D}}} \left[-\log \frac{e^{c_{i,j}}}{e^{c_{i,j}} + e^{c_{j,i}}} \right]$, where $c_{i,j} = \text{subdom}_\alpha(\mathbf{f}_\psi(\tilde{\xi}_i), \mathbf{f}_\psi(\tilde{\xi}_j))$.

Given the learned feature representation, a γ -satisficing policy can be learned via subdominance minimization in Algorithm 1. This differs from the formulation of T-REX in two key aspects. First, under the exponential preference model [Bradley and Terry, 1952; Christiano *et al.*, 2017; Brown *et al.*, 2019; Brown *et al.*, 2020], we employ subdominance between pairs of demonstrations as a loss function, rather than a linear cost function. The second difference emerges from choosing subdominance as the loss function: our formulation permits learning latent representations of *any* dimensionality, rather than just a scalar cost signal; such a vector representation allows us to recover multiple, competing objectives from preferences, rather than arbitrarily extrapolating over a scalar reward signal.

3 Experiments

3.1 Demonstrations

We conduct experiments using a mix of simple, classic control environments (cartpole, lunarlander) and complex robotics environments (Mujoco hopper, halfcheetah, walker) from OpenAI Gym [Brockman *et al.*, 2016]. For each environment, we obtain 100 demonstrations from a suboptimal policy learned using PPO. This ensures that the majority of the resulting demonstrations are suboptimal and noisy. Human demonstrations for the lunarlander used in Section 3.7 are collected from non-expert, human players using the joysticks on an Xbox 360 video game controller. Demonstration return statistics for environment-specific demonstration sets of varying quality are provided in Table 1.

Cost features are environment-specific as follows:

- cartpole: (cart position)², (cart velocity)², and (pole angle)², (pole angular velocity)²;
- lunarlander: (x-position)², (y-position)², (x-velocity)², (y-velocity)², (angle)², (angular velocity)², and control costs;

Environment	learning rate	entropy coeff.	mini-batch	horizon	epochs	clip range	total steps
cartpole	1e-4	0	512	2048	10	0.2	2e6
lunarlander	1e-4	1e-6	2048	2048	10	0.2	2e6
hopper	9.8e-5	1e-2	512	2048	5	0.2	5e6
halfcheetah	9.8e-5	1e-4	256	2048	5	0.2	5e6
walker	2e-5	6e-4	32	512	20	0.1	1e6

Table 2: Values of PPO hyperparameters for each environment.

- hopper: inverse x -velocity, inverse z -velocity, inverse z -position, inverse torso angle, and control cost;
- halfcheetah: three variants of inverse x -velocity, and control cost; and
- walker: inverse x -velocity, inverse z -position, and control cost.

3.2 Baseline Methods

We employ behavior cloning (BC), generative adversarial imitation learning (GAIL) [Ho and Ermon, 2016b], and adversarial inverse reinforcement learning (AIRL) [Fu *et al.*, 2018] as classical imitation baselines. From extrapolative (better-than-demonstrated) imitation approaches, we compare our approach against T-REX [Brown *et al.*, 2019] rather than its more recent extension, D-REX [Brown *et al.*, 2020], since the latter only adds a new method for automatically generating ranked, synthetic demonstrations, while still retaining the core formulation and loss function introduced in T-REX. As a result, we are better able to examine fundamental differences between learning from ranked demonstrations and learning by subdominance minimization. To provide a more comparable baseline, we learn the T-REX cost function C as a linear combination of cost features \mathbf{f} and cost function weights \tilde{w} , rather than as a function mapping from the observation vector ϕ to cost C (abbreviated T-REX_{CF}); this can be thought of as replacing the penultimate layer of T-REX’s cost network with a known, predefined state cost representation. We also train an unaltered version of T-REX (abbreviated T-REX) on our demonstrations and provide these results for reference.

We implement the policy optimization of MinSubFI using Stable Baselines3 [Raffin *et al.*, 2021]. Across all experiments, all baseline methods use the same base policy model paired with Stable Baseline3’s implementation of the PPO algorithm [Schulman *et al.*, 2017]. The experiments are not based on extensive hyperparameter tuning; rather, all policy networks use nearly the same hyperparameters (Table 2).

3.3 Training and Bootstrapping

Using Algorithm 1 and analytically computed α values in step 4, we initialize our Online MinSubFI training with a policy that is pretrained via Offline MinSubFI (Corollary 10); we motivate this choice via an ablation study with different policy initializations in the extended version of this paper. For all of our experiments throughout the paper, we employ a quadratic expansion of the original cost features as the vectorized outer product of the original cost feature vector, $\mathbf{f}_{\text{expanded}} = \text{vec}(\mathbf{f} \cdot \mathbf{f}^\top)$.

We train two snippet-based MinSubFI models: one using fixed snippet lengths and alignments (MinSubFI_{SNIP})

Environment	Baselines					Ours				
	BC	TREX	TREX _{CF}	AIRL	GAIL	MinSubFI _{OFF}	MinSubFI _{ON}	MinSubFI _{SNIP}	MinSubFI _{SNIP*}	MinSubFI _{LCF}
cartpole	0.19	0.04	0.00	0.09	2.24	2.62	2.64	2.68	2.59	2.04
lunarlander	0.00	0.00	0.00	0.02	0.00	0.49	1.03	1.16	1.49	2.24
hopper	0.00	0.00	0.00	0.02	6.40	0.86	1.63	1.29	1.38	1.46
halfcheetah	0.00	0.00	0.00	1.12	3.99	1.93	1.93	1.85	1.77	1.74
walker2d	8.73	0.00	0.00	0.00	0.00	2.15	1.44	1.46	1.54	1.86

Table 3: Relative γ -satisficing values of different versions of MinSubFI on the basic cost features (values greater than 1 are formatted in **bold** and the best of each environment is additionally colored **green**).

Environment	Demonstrations	Baselines					Ours				
		BC	TREX	TREX _{CF}	AIRL	GAIL	MinSubFI _{OFF}	MinSubFI _{ON}	MinSubFI _{SNIP}	MinSubFI _{SNIP*}	MinSubFI _{LCF}
cartpole	116 (74)	70 (37)	199 (0.1)	199 (0.1)	15 (4)	200 (0.0)	200 (0.1)	198 (2)	199 (0.9)	197 (2)	200 (0.0)
lunarlander	113 (132)	164 (27)	-171 (3)	195 (7)	-416 (30)	256 (9)	268 (0.5)	268 (0.6)	268 (0.6)	266 (1)	-269 (153)
hopper	858 (884)	671 (80)	1335 (15)	2657 (28)	11 (4)	601 (30)	570 (33)	1470 (149)	1070 (166)	849 (125)	1373 (305)
halfcheetah	686 (584)	1283 (53)	1017 (7)	1535 (49)	768 (47)	1595 (4)	1626 (10)	1591 (8)	1591 (14)	1576 (10)	1463 (86)
walker2d	891 (1141)	526 (99)	20 (0.0)	90 (5)	-3 (0.1)	489 (82)	1461 (449)	1688 (479)	1861 (481)	1446 (402)	2592 (182)
lunarlander _{human}	173 (118)	-75 (83)	-569 (114)	-310 (86)	-197 (71)	-254 (5)	-25 (21)	193 (5)	6 (10)	-76 (11)	-487 (267)

Table 4: Mean (and standard deviation) of the true episode returns of the **held out demonstrations** and trajectories sampled from different imitation learning methods’ learned policies for all environments using synthetic demonstrations and for lunarlander with human demonstrations (bottom row).

and one with alignments that are selected through optimization (MinSubFI_{SNIP*}). We additionally train an online subdominance minimizer with a learned cost feature space (MinSubFI_{LCF}) of $K' = 3$ dimensions via Definition 9. We use a multi-layer perceptron network with two hidden layers of width 8 as our cost feature architecture. In contrast with TREX, which employs four different levels of preference (or ranks) to categorize demonstration quality, we consider two preference levels (i.e., *acceptable* and *not acceptable*).

3.4 Demonstrator Acceptability Analysis

In Table 3, we evaluate the rate that the imitator satisfies demonstrations (Definition 7), guaranteeing demonstrator satisfaction, relative to the rate that a randomly chosen demonstration satisfies other demonstrations,

$$P(\xi \in \Omega_{\tilde{\xi}}) / P(\tilde{\xi}' \in \Omega_{\tilde{\xi}}),$$

using trajectory-level cost features. Imitation learning methods designed to minimize a predictive loss (BC) or a learned cost function (TREX) produce trajectories with very different cost features than those of the demonstrations, leading to small values in this analysis (with a few exceptions, e.g., BC on walker2d). More specifically, aggressively optimizing an estimated cost function using reinforcement learning often focuses too narrowly on minimizing one or a small subset of cost features at the expense of ignoring one or more other features, allowing them to take unacceptable values. For example, though TREX produces cartpole policies keeping the pole upright (near optimally), it does so with much larger amounts of horizontal motion than demonstrations exhibit, making it potentially unacceptable to the demonstrator. GAIL, which employs a discriminator to help make demonstrator and imitator behavior indistinguishable, achieves high acceptability rates on some environments. However, it does so inconsistently, with no relative satisficing on lunarlander and walker2d.

In contrast, since MinSubFI minimizes an upper bound on the imitator’s satisficing value, it consistently guarantees demonstrator acceptability much more frequently. We also find that online subdominance minimization tends to provide more frequent acceptability guarantees than the offline variant. Additionally, snippet-optimized subdominance minimization frequently provides the large rates of acceptability guarantees in different environments. This is despite the fact that snippet optimization seeks to provide snippet-level demonstrator acceptability, while Table 3 measures trajectory-level acceptability, indicating its general benefit in guiding policy optimization.

In addition, though MinSubFI_{LCF} learns its own space of cost features, it still provides large rates of guaranteed demonstrator acceptance in the original, provided cost feature space for most environments. This provides some evidence that knowing the demonstrator’s cost feature space is unnecessary for providing demonstrator-acceptable behavior, even though it may not be possible to formally guarantee demonstrator acceptance in such settings.

3.5 True Returns Using Full Demonstration Set

Unlike “real-world” imitation learning tasks, the true returns used to construct the demonstrator’s policy are known in our experiments. Though MinSubFI seeks to achieve demonstrator acceptability for all cost functions defined by its cost features (Table 3), it should also provide improvements over demonstrations in terms of true return when the true return can be (approximately) defined by the cost features. We note that having a fixed true return function for all demonstrations is a strong assumption from the perspective of our formulation of satisficing theory, which allows the acceptability set to vary with each demonstration. In Table 4, we evaluate the true returns of the demonstrations and each of the imitation learning methods averaged over three random seeds.

We find that Behavioral Cloning (BC) and AIRL often

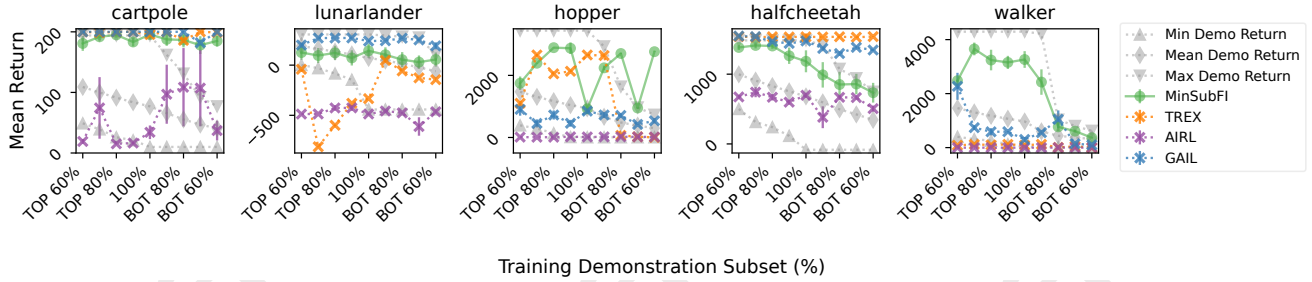


Figure 4: Mean true returns of 100 trajectories rolled out from the learned policies and the minimum, average, and maximum reward of the training set trajectories. Each policy was trained on a subset of demonstrations obtained by removing the best or worst 10%, 20%, 30%, or 40% of the demonstrations. Compared to T-REX (orange) and AIRL (purple), the performance of MinSubFI (green) is more robust to increases in the proportion of suboptimal demonstrations in the dataset.

underperforms relative to the demonstrations (except for halfcheetah), while the relative performance for TREX and GAIL is more mixed. We note that GAIL’s higher relative satisficing performance in Table 3 does not translate into higher true returns (e.g., on hopper). In contrast, the various forms of MinSubFI tend to consistently outperform the demonstrations with only a few exceptions (e.g., MinSubFI_{OFF} on hopper). In terms of the numerical true returns, different variants of MinSubFI provide the highest returns except for hopper, in which TREX_{CF} provides the largest returns. Interestingly, while TREX benefits from using the cost features as the basis for its cost function estimate, cost function learning provides significant improvements for MinSubFI_{LCF} in walker2d and cartpole.

3.6 Demonstration Quality and Performance

Demonstrated behavior is often noisy and suboptimal, making learning from such data a desirable capability. In this section, we control the quality of demonstrations used for imitation. We sort all demonstrations by their total (true) return and then choose a subset by retaining the best or worst 90%, 80%, 70%, or 60% of the original set. We use this demonstration subset to train T-REX and Online MinSubFI. The performance is shown in Figure 4.

For the simple cartpole environment, both TREX and MinSubFI are able to continue outperforming the best demonstrations even when they become worse in quality. For the remaining environments, except for halfcheetah in which TREX performs exceptionally well, MinSubFI tends to provide comparatively better true returns than the baselines as the quality of demonstrations becomes worse.

3.7 Performance with Human Demonstrations

Human-provided demonstrations often exhibit multi-modal patterns and irregular noise distributions, making them harder to learn from. We explore the behavior of our method trained on human-provided demonstrations, using the continuous version of Lunarlander and compare it against imitation baselines. The results, averaged over three seeds, are shown in the last row of Table 4. MinSubFI_{ON} clearly outperforms all baselines, which appear unable to learn from human demonstrations in this setting.

4 Conclusions and Future Work

In this paper, we reframed imitation learning using satisficing theory to develop MinSubFI, a policy gradient approach for seeking to make the imitator’s behavior acceptable to the demonstrator by directly minimizing policy subdominance—based on entire trajectories or selectively chosen snippets. We present both variants for online and offline learning, and show how offline bootstrapping results in significant simulator sample efficiency. Further, we present a feature presentation learning method using offline subdominance-minimization from demonstrations. Using multiple control and robotics environments, we show that MinSubFI frequently guarantees demonstrator acceptability, while existing imitation learning methods rarely do. Further, we show that MinSubFI with learned cost features provides demonstrator acceptability in our hand-specified cost feature space. Despite being designed for the more flexible setting in which the acceptability set can change for each demonstration, our experiments show that MinSubFI provides competitive true returns (without explicit assumptions about a static true cost function, as in other imitation learning methods). We additionally show that MinSubFI is more robust to degradation in the quality of demonstrations used for training compared to existing approaches.

There are many interesting directions for future work. Learning feature representations without supplemental annotations is a challenging problem that would make our method easier to employ in practice. Developing more general strategies for snippet generation could better leverage demonstrations across different tasks or domains. Exploring other methods for guaranteeing high levels of demonstrator acceptability is also of great interest. Finally, conducting experiments in application areas that lack true return functions for evaluation and/or have notions of acceptability that are dynamic and subjective is an important future direction.

Acknowledgments

This research is supported by NSF Award #2312955.

Contribution Statement

Rushit N. Shah and Nikolaos Agadakos contributed equally to the development of the approach, and the design and execution of experiments presented in this paper.

References

- [Abbeel and Ng, 2004] Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, pages 1–8, 2004.
- [Amodei et al., 2016] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.
- [Bradley and Terry, 1952] Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.
- [Brockman et al., 2016] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.
- [Brown et al., 2019] Daniel Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond sub-optimal demonstrations via inverse reinforcement learning from observations. In *International Conference on Machine Learning*, pages 783–792. PMLR, 2019.
- [Brown et al., 2020] Daniel S. Brown, Wonjoon Goo, and Scott Niekum. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Proceedings of the Conference on Robot Learning*, pages 330–359, 2020.
- [Burchfiel et al., 2016] Benjamin Burchfiel, Carlo Tomasi, and Ronald Parr. Distance minimization for reward learning from scored trajectories. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1), 2016.
- [Chen et al., 2020] Letian Chen, Rohan Paleja, and Matthew Gombolay. Learning from suboptimal demonstration via self-supervised reward regression. *arXiv preprint arXiv:2010.11723*, 2020.
- [Christiano et al., 2017] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in Neural Information Processing Systems*, 30, 2017.
- [Fu et al., 2018] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. In *International Conference on Learning Representations*, 2018.
- [Ho and Ermon, 2016a] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*, volume 29, 2016.
- [Ho and Ermon, 2016b] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.
- [Kalman, 1964] Rudolf E. Kalman. When is a linear control system optimal? *Trans ASME, J. Basic Eng.*, pages 51–60, 1964.
- [Ladha et al., 2023] Reza Ladha, Thijs Meenink, Jorrit Smit, and Marc D de Smet. Advantages of robotic assistance over a manual approach in simulated subretinal injections and its relevance for gene therapy. *Gene Therapy*, 30(3-4):264–270, 2023.
- [Levine et al., 2020] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.
- [Memarrast et al., 2023] Omid Memarrast, Linh Vu, and Brian D Ziebart. Superhuman fairness. In *Proceedings of the International Conference on Machine Learning*, volume 202, pages 24420–24435. PMLR, 23–29 Jul 2023.
- [Miller, 1956] George A Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63(2):81, 1956.
- [Ng and Russell, 2000] Andrew Y Ng and Stuart J Russell. Algorithms for inverse reinforcement learning. In *International Conference on Machine Learning*, pages 663–670, 2000.
- [Osa et al., 2018] Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J Andrew Bagnell, Pieter Abbeel, and Jan Peters. An algorithmic perspective on imitation learning. *arXiv preprint arXiv:1811.06711*, 2018.
- [Pomerleau, 1991] Dean A. Pomerleau. Efficient Training of Artificial Neural Networks for Autonomous Navigation. *Neural Computation*, 3(1):88–97, 03 1991.
- [Rafailov et al., 2024] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.
- [Raffin et al., 2021] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.
- [Schulman et al., 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [Simon, 1956] Herbert A Simon. Rational choice and the structure of the environment. *Psychological review*, 63(2):129, 1956.
- [Sutton et al., 1999] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in Neural Information Processing Systems*, 12, 1999.
- [Syed and Schapire, 2007] Umar Syed and Robert E Schapire. A game-theoretic approach to apprenticeship learning. *Advances in Neural Information Processing Systems*, 20, 2007.
- [Tangkaratt et al., 2021] Voot Tangkaratt, Nontawat Charoenphakdee, and Masashi Sugiyama. Robust

imitation learning from noisy demonstrations. In *International Conference on Artificial Intelligence and Statistics*, pages 298–306. PMLR, 2021.

[Vapnik and Chapelle, 2000] Vladimir Vapnik and Olivier Chapelle. Bounds on error expectation for support vector machines. *Neural Computation*, 12(9):2013–2036, 2000.

[Whelan, 2008] Robert Whelan. Effective analysis of reaction time data. *The Psychological Record*, 58:475–482, 2008.

[Wirth *et al.*, 2017] Christian Wirth, Riad Akrou, Gerhard Neumann, and Johannes Fürnkranz. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136):1–46, 2017.

[Wu *et al.*, 2019] Yueh-Hua Wu, Nontawat Charoenphakdee, Han Bao, Voot Tangkaratt, and Masashi Sugiyama. Imitation learning from imperfect demonstration. In *International Conference on Machine Learning*, pages 6818–6827. PMLR, 2019.

[Zhang *et al.*, 2021] Songyuan Zhang, Zhangjie Cao, Dorsa Sadigh, and Yanan Sui. Confidence-aware imitation learning from demonstrations with varying optimality. *arXiv preprint arXiv:2110.14754*, 2021.

[Ziebart *et al.*, 2022] Brian D. Ziebart, Sanjiban Choudhury, Xinyan Yan, and Paul Vernaza. Towards uniformly superhuman autonomy via subdominance minimization. In *Proceedings of the International Conference on Machine Learning*, pages 27654–27670, 2022.