A Behavior-first approach to Multi-Intention Inverse Reinforcement Learning

Anonymous authors
Paper under double-blind review

Abstract

Inverse Reinforcement Learning (IRL) seeks to infer reward functions from expert demonstrations. When demonstrations come from multiple experts with different intentions, the problem becomes Multi-Intention IRL (MI-IRL). MI-IRL methods generally follow a reward-first paradigm: they ask which reward, if followed, could have generated each trajectory. This leads to trajectory similarity being based on reward likelihood rather than on the relationships between behaviors. In deep generative MI-IRL methods, this paradigm further couples behavior clustering and reward learning, making such methods dependent on prior knowledge of the true number of modes K^*. This limits adaptability to unseen behaviors, and restricts the analysis to learned rewards rather than relationships across behaviors. In contrast, we approach MI-IRL from a different standpoint. We propose Contrastive Multi-Intention IRL (CoMI-IRL), a transformer-based unsupervised behavior-first framework: it first learns a representation of trajectory dynamics that captures intrinsic behavioral similarity, then clusters trajectories in that space, and finally learns a reward for each cluster. Our experiments show that CoMI-IRL outperforms existing approaches without needing K^* or labels, while allowing for visual interpretation of behavior relationships and adaptation to unseen behavior without full retraining.

1 Introduction

Inverse Reinforcement Learning (IRL) deals with tasks where the reward function is unknown or difficult to define by recovering a reward function from demonstrations (Ng & Russell, 2000). IRL has progressed from modeling rewards as linear combination of features (Ng & Russell, 2000; Abbeel & Ng, 2004) , through ambiguity reduction methods (Ziebart et al., 2008; Ramachandran & Amir, 2007), to deep learning-based methods in generative adversarial (Ho & Ermon, 2016; Fu et al., 2017) settings. However, these approaches assume all demonstrations share a unique goal. This is often not the case, as datasets can contain non-optimal and conflicting modes of behavior collected from multiple sources. Such problem is known as Multi-Intention IRL (MI-IRL) (Babes et al., 2011).

MI-IRL addresses reward modeling in three ways: (i) as a linear combination of features in Expectation-Maximization (EM) (Babes et al., 2011; Bighashdel et al., 2023), (ii) through Bayesian methods (Choi & Kim, 2012; Michini & How, 2012), and (iii) as neural networks in generative frameworks (Li et al., 2017; Fu et al., 2023; Hausman et al., 2017), with the latter achieving most promising results. All three ways ask which reward function, if followed, would most likely have generated each trajectory. In this *reward-first* paradigm, trajectory similarity is defined through reward likelihood rather than through intrinsic relationships between behaviors. In deep generative methods, this paradigm leads to the coupling of behavior clustering and reward learning: a latent code c, sampled from a categorical distribution with K fixed categories, simultaneously defines behavioral modes and conditions reward learning, such that the two tasks cannot be disentangled.

This coupling, inherent to reward-first approaches, introduces three key limitations. First, fixing K assumes knowledge of the true number of behavioral modes K^*, often unavailable and hard to specify, akin to the challenge of selecting cluster numbers in data analysis (Wang, 2010; Tibshirani et al., 2001; Pan et al., 2013), which can lead to the risk of misspecification. In continual settings, where new behaviors are added, models

with fixed K require full retraining. Methods inferring K from data have been explored (e.g.,nonparametric Bayesian IRL (Choi & Kim, 2012)), however their application in continuous spaces is challenging due to computational limitations (Ramponi et al., 2020). Second, in coupled approaches the failure in one task can cascade and complicate diagnosis (i.e. it's hard to verify if the model failed to learn the reward or to cluster behavior correctly). Finally, this design hinders interpretability, since coupled methods allow to analyze behaviors only by deploying the latent-conditioned policies, unrelated to the original dynamics of the dataset. As these limitations are a byproduct of the reward-first paradigm, we explore an alternative solution.

In this paper, we adopt a *behavior-first* paradigm on MI-IRL, asking which trajectories share similar dynamics rather than which reward function would most likely have generated them. This leads to a decoupled perspective, separating "what happened" (behavior clustering), from "why it happened" (reward learning). We propose Contrastive Multi-Intention IRL (CoMI-IRL), a transformer-based[1] unsupervised framework that learns trajectory embeddings via contrastive training. The learned representations are grouped for their intrinsic patterns, not by reward likelihood. By inducing a similarity structure in the embedding space, we can discover clusters of behavioral modes via graph-based connectivity without knowing K. This reduces MI-IRL to independent single-intention IRL per cluster, also enabling direct adaptation to new behaviors.

In our experiments, we analyze how the relationship between the assumed (K) and true (K^*) number of behavioral modes (i.e., $K \neq K^*$,$K = K^*$) impacts clustering and reward learning. We show how our approach avoids this problem, facilitates the interpretation of relationships in the dataset and adapts to new behaviors without full retraining. Our results demonstrate that our design outperforms other approaches in both clustering quality and reward learning without requiring K, alleviating the limitations of coupled methods, enhancing MI-IRL adaptability to new behaviors.

2 Related Work

In this Section, we provide an overview of the current reward-first methods in the field of MI-IRL, and we further explore other methods that also started to look at similar tasks from a behavior-first perspective, especially in the area of trajectory embedding.

EM-based Methods Early parametric EM-based methods (Babes et al., 2011; Snoswell et al., 2021; Ramponi et al., 2020) typically require knowledge of K and model the reward function as a linear combination of handcrafted features (Ng & Russell, 2000; Levine et al., 2010), and the performance highly depends on the how well these are selected (Choi & Kim, 2013). In CoMI-IRL, as in other deep MI-IRL methods, we do not need to handcraft features.

Nonparametric Bayesian Methods When K is unknown, it can be inferred through Bayesian solutions (Choi & Kim, 2012; Bighashdel et al., 2021), with some works learning the reward more flexibly through Gaussian Process kernels (Levine et al., 2011) or deep learning (Bighashdel et al., 2021). However, this comes at the cost of a significantly larger overhead and computational costs (Ramponi et al., 2020; Wulfmeier et al., 2015; Zhu et al., 2024). The application in continuous state-action spaces as the ones considered in our work is sparsely addressed in literature (Almingol & Montesano, 2015; Rajasekaran et al., 2017) due to the computational limitations of apply Bayesian methods in such spaces. In CoMI-IRL, we work specifically on continuous state-action spaces, where the application of current methods in this line area is limited.

Deep Generative MI-IRL Methods Deep Generative MI-IRL method transfer the IRL problem to multi-expert settings by learning multiple rewards, often via latent conditioning. Different methods extends GAIL with MI maximization between trajectories and latent codes c, enabling mode separation without explicit clustering (Li et al., 2017; Hausman et al., 2017; Fu et al., 2023). However, maximizing a lower bound on MI in these works depends on the generator's ability to cover the distribution. If the policy fails to capture a specific mode, the clustering signal degrades. In contrast, CoMI-IRL's behavior-first decoupled design employs a discriminative objective that optimizes the embedding geometry for separability, independent of the policy's performance.

[1]The repository will be made available on GitHub after acceptance or upon request.

The approaches mentioned so far cluster trajectories via a reward-first likelihood-based mechanisms: in EM-based methods, it is the likelihood of a trajectory being generated by a certain reward function; Bayesian solutions cluster via posterior inference over the components of the mixture model; and generative approaches cluster through the probability of generation from a conditioned policy. A connection between likelihood-based and generative-based approaches in IRL was established by Finn et al. (2016), which persists in the multi-intention case. In contrast, CoMI-IRL clusters via contrastive embedding similarity. In each method, behavioral structure is discovered through reward-generating plausibility rather than through intrinsic dynamic similarity. With CoMI-IRL, instead, we define similarity by trajectory dynamics in a learned embedding space, adopting a behavior-first paradigm.

Trajectory Embedding Methods The use of sequence models and contrastive techniques to learn trajectory embeddings gathered a lot of attention in recent works. Vivekanandan et al. (2025) employ Transformer encoders with triplet losses on spatially augmented short trajectories to learn motion priors for forecasting. TrajCL (Chang et al., 2023) employs dual-feature (spatial/structural) self-attention Transformers with contrastive augmentations (shifting/masking) for similarity queries, focusing on spatial ranking without intention handling. Kujanpää et al. (2025) proposes Discrete Style Diffusion Policy (DSDP) as a Behavioral Cloning method, combining InfoNCE with Lookup-Free Quantization (LFQ) to extract discrete styles on sub-trajectories for conditional diffusion behavior cloning, tying embeddings to generation rather than reward inference. Ge et al. (2025) introduce Variational Trajectory Embeddings (VTE), a method designed to capture a continuous ability vector representing a trajectory's proficiency on a single task. In VTE, first a probabilistic skill extractor (based on a Hierarchical State Space Model and the Learning Options via Compression (LOVE) (Jiang et al., 2022) framework) processes a trajectory to yield a sequence of latent skill distributions. Secondly, a transformer-based Variational Autoencoder (VAE) gets this skill sequence in input to compute final embedding's posterior. The model is trained to maximize the VAE's Evidence Lower Bound (ELBO), pairing a KL divergence regularizer and an action reconstruction loss, where a policy conditioned on the embedding must replicate the trajectory's actions.

In contrast with VTE's variational approach, which relies on intermediate skill representations and a reconstruction objective to model how a single task is performed, CoMI-IRL learns directly from raw state-action sequences using a purely contrastive objective. Its goal is to learn a discriminative embedding space for behavioral clustering, enabling a fully decoupled and adaptive MI-IRL method.

3 Background

In this section, we formulate the MI-IRL problem and review the key concepts in contrastive learning applied in our work.

3.1 Multi-intention IRL

In multi-intention IRL, a finite-state Markov Decision Process without Rewards (MDP\R) is the tuple $(S, A, T, \gamma, b_0, \tau_1, \ldots, \tau_D)$, where S is the state space, A is the action space, $T : S \times A \times S \to [0, 1]$ is the transition probability function, $\gamma \in [0, 1)$ is the discount factor, b_0 is the probability of starting in state s, and $\tau_i = (s_0, a_0, s_1, a_1, \ldots, s_T, a_T)$ is the ith trajectory in a dataset of D trajectories. Each trajectory is generated from the optimal policy of one of K^* behavioral modes, each corresponding to an intention mathematically represented by a reward function. Trajectories are assumed to be without labels, and the goal is to infer the number of behavioral modes and each respective reward function.

3.2 Contrastive Learning

Contrastive Learning (CL) is a self-supervised technique for learning representations by mapping similar (positive) examples closer in a an embedding space, and dissimilar (negative) objects apart. CL is often applied to unsupervised sequence embedding methods with pretrained transformers to capture global semantics without requiring labels. For example, BERT (Devlin et al., 2019) prepends a special token called *CLS* to the beginning of every sequence. As the model processes the sequence, this token learns to collect information from all other elements, ending up as a single vector that summarizes the entire input (Qin et al., 2023;

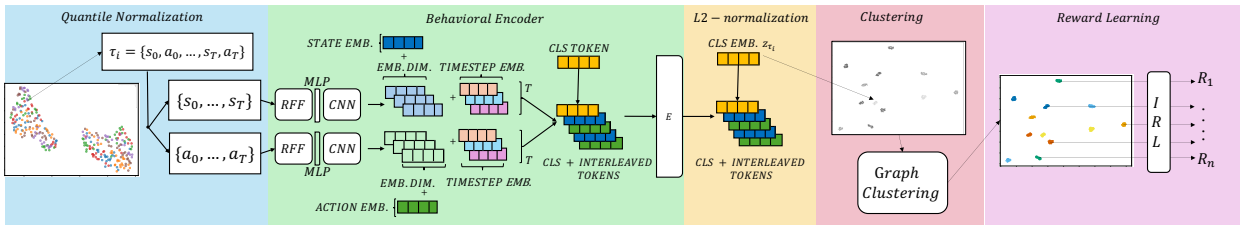

Figure 1: **CoMI-IRL pipeline.** States and actions of each quantile normalized input trajectory τ_i pass through a Random Fourier Feature (*RFF*) encoding with Gaussian mapping, a shallow *MLP* and a 1D *CNN* layer. After adding timestep and modality embeddings (distinguishing states and actions), we interleave the two sequences and prepend a *CLS* token summarizing the trajectory. On the resulting embeddings, we apply graph-clustering to get cluster labels and apply IRL independently on each cluster.

Wang et al., 2024; Zou et al., 2024). We use this concept to represent trajectories and distinguish behaviors: we treat each τ_i as a "sentence" and use the *CLS* token as a compact summary of the behavior.

Contrastive learning methods benefits from data augmentation (Deng et al., 2023), but strong augmentations could disrupt behavior coherence. To form positive pairs without disrupting the coherence of the behavior, weak augmentations can be used. An effective strategy to form positive pairs is to create two dropout-augmented views of the input by passing each input sequence through the model twice, while all other trajectories in the batch serve as negative elements, following the intuition from Gao et al. (2021). These dropout-augmented views can then be passed into a contrastive loss such as the symmetric InfoNCE (Oord et al., 2018; Chen et al., 2020), a central component of our approach that encourages informative and distinguishable embeddings. Since InfoNCE is asymmetric with respect to the anchor, the symmetric version treats each view as an anchor in turn, ensuring both views contribute equally to the contrastive signal. The loss for a single view $z^1_{\tau_i}$ is

$$\ell(z^1_{\tau_i}, z^2_{\tau_i}) = -\log \frac{\exp(\mathrm{sim}(z^1_{\tau_i}, z^2_{\tau_i})/\rho)}{\sum_{k=1,k\neq i}^{2N} \exp(\mathrm{sim}(z^1_{\tau_i}, z^2_{\tau_k})/\rho)}, \tag{1}$$

and symmetric for $z^2_{\tau_i}$, where sim is cosine similarity, ρ is temperature and N is the number of trajectories in the batch. In addition, we want to further distinguish embeddings so that learned representations can be consistently clustered. For this, regularization terms such as a Deep InfoMax (DIM) (Hjelm et al., 2019) can help, by maximizing mutual information via Jensen-Shannon divergence (JSD) so that each local element of the input informs the whole representation. DIM trains a discriminator f_ϕ to give positive scores to pairs $f_\phi(z_{\tau_i}(\theta), z_{\tau_i(q)}(\theta))$ with the global summary paired with its own local features (joint pairs), and negative scores to pairs $f_\phi(z_{\tau_i}(\theta), z_{\tau_j(q)}(\theta))$ where $i \neq j$, so that:

$$\mathcal{L}_{\mathrm{DIM}} = -\mathbb{E}_{P_J}[\log \sigma(f_\phi(z_{\tau_i}, z_{\tau_i(q)}))] - \mathbb{E}_{P_M}[\log(1 - \sigma(f_\phi(z_{\tau_i}, z_{\tau_j(q)})))], \tag{2}$$

where $\sigma(x) = \frac{1}{(1+e^{-x})}$ is the sigmoid function.

4 Contrastive Multi-Intention IRL

In this section we introduce CoMI-IRL, a behavior-first framework that decouples behavior clustering and reward learning. Using a fully unsupervised contrastive approach, CoMI-IRL maps *quantile normalized* demonstrations into an embedding space via a *Behavioral Encoder (BE)* with *L2-normalized* outputs, followed by *Clustering*, defining single-intention groups for downstream *Reward Learning* (Figure 1). At its core, the BE, a Transformer Encoder Vaswani et al. (2017) with a pre-processing block, generates trajectory representations as *CLS* tokens to represent complex behaviors based on their intrinsic patterns (Section 4.1). As similar trajectories are grouped together, cluster labels are obtained via a graph-based method for the subsequent IRL step (Section 4.2). CoMI-IRL can be adapted to new behavior without full retraining (Section 4.3).

4.1 Behavioral Encoder Architecture

The first step is to embed trajectories in a latent space such that similar behaviors are grouped together. We base our approach on the Transformer model (Vaswani et al., 2017) as it allows to create representations encoding information between distant timesteps through its attention mechanism. We quantile normalize the input trajectories (Bolstad et al., 2003), independently mapping each dimension to a normal distribution, since the state–action spaces contain high-frequency elements with substantially different magnitudes.

Further, as states S_{τ_i} and actions A_{τ_i} of each input trajectory $i-$th have different dynamics, we encode them with two different pre-processing heads, one for each sequence (Chen et al., 2021). Each head applies a Random Fourier Features (*RFF*) encoding with Gaussian mapping (Tancik et al., 2020; Zheng et al., 2022; Li et al., 2021), followed by a shallow *MLP* and a 1D *CNN*, to capture short-range temporal patterns in the trajectory. The *RFF* encoding overcomes known *MLP* spectral bias towards low-frequency functions by injecting a rich set of sinusoidal features at multiple random orientation and frequencies. Each block's output is enriched with a timestep (used as positional encoding) and either a state or action embedding to further differentiate. Next, we interleave them to recreate the single input sequence. We prepend a *CLS* token to this sequence that will function as the trajectory summary. The resulting sequence is then passed through the encoder to capture the trajectory's global dependencies, summarized in the *CLS* embedding, which for simplicity we name z_{τ_i}. Finally, we L2-normalize the *CLS* token of the encoder's output, restricting the output space to the unit hypersphere (Wang & Isola, 2020).

4.1.1 Contrastive Framework and Loss Function

As we aim to learn a discriminative representation z_{τ_i} of each input trajectory τ_i, we apply the symmetric InfoNCE loss at multiple granularity levels, with DIM included as a regularization term (cf. Section 3.2). First, we generate two dropout-augmented views of each input τ_i: $z^1_{\tau_i}$ and $z^2_{\tau_i}$, forming a positive pair. This maps similar trajectories close together in the embedding space, allowing for the emergence of areas with distinct behavioral modes. This loss component is defined as:

$$\mathcal{L}_{\text{CLS}} = \frac{1}{2N} \sum_{i=1}^{2N} \ell_{\text{InfoNCE}}(z^1_{\tau_i}, z^2_{\tau_i}). \tag{3}$$

Second, we sample n segments of length L, with starting point $q \in \{0, T-L\}$ from each input trajectory. Each segment is defined as $\tau_i(q, L) = \{s_q, a_q, ..., s_{q+L}, a_{q+L}\}$, and passed through the encoder to create n segment embeddings $z^k_{\tau_i(q,L)}$. Aiming to promote consistency between local (segment-level) and trajectory representations, we apply InfoNCE between each trajectory embeddings and its segment embeddings. This loss component, $\mathcal{L}_{\text{SEG}}$, performs the same operation as Equation 3, but applied between z_{τ_i} and $z^k_{\tau_i(q,L)}$. To push representations even closer together in each area, we apply InfoNCE between all unique pairs of segment embeddings:

$$\mathcal{L}_{\text{PAIR}} = \frac{1}{\binom{n}{2}} \sum_{k=1}^{n-1} \sum_{j=k+1}^{n} \ell_{\text{InfoNCE}}(z^k_{\tau_i(q,L)}, z^j_{\tau_i(q,L)}). \tag{4}$$

We also add a DIM component to the loss, $\mathcal{L}_{\text{DIM}}$, which follows Equation 2, with the goal of keeping local structure consistent and further stabilize representations. This component explicitly maximize the mutual information between trajectory representations and their local features without forcing token-level separation across trajectories. Combining the components with α, β, γ and δ as balancing coefficients, we get our complete loss:

$$\mathcal{L} = \alpha\mathcal{L}_{\text{CLS}} + \beta\mathcal{L}_{\text{DIM}} + \gamma\mathcal{L}_{\text{SEG}} + \delta\mathcal{L}_{\text{PAIR}}. \tag{5}$$

4.2 Clustering and Reward Learning

In principle, any clustering method not requiring K^* could be used. However, density-based methods such as HDBSCAN (Campello et al., 2013) can be highly sensitive to hyperparameters, require clear density gaps (not

often available in continuous manifolds), and its adaptation relies on the fixed density structure of the original tree. Therefore, we cluster the embeddings using a graph-based community detection approach not requiring K^* via local connectivity. We construct a weighted k-nearest neighbor graph over trajectory embeddings, where nodes represent trajectories and edge weights are defined by cosine similarity between embedding. This choice is motivated by the fact that our contrastive objective optimizes for directional alignment. Rather than assuming a fixed global scale, we analyze the connectivity structure of the graph to identify stable behavioral partitions. In cases where the graph decomposes into multiple connected components, these components directly define clusters. If the graph remains fully connected, we apply community detection using the Leiden algorithm (Traag et al., 2018) to identify coherent subgraphs by optimizing modularity. Graph parameters are selected based on clustering stability across resolution levels, avoiding reliance on external clustering objectives or prior assumptions about the number of modes k.

We augment graph edge weights with trajectory-level Jacobian features estimated through finite differences, as they showed low correlation with the embeddings (detailed in the Appendix). Incorporating Jacobian similarity preserves the decoupling from reward learning, while helping separate distinct behavioral patterns. This is especially useful in embedding spaces with a continuous manifold structure, where embedding similarity alone may be insufficient to separate closely related behaviors. These statistical features summarize average local state–action sensitivities along each trajectory and aim to capture behavioral dynamics independently of reward structure or task specification.

For each discovered cluster, we independently apply a single-intention IRL algorithm (we test our framework with deep learning based methods to allow non-linear reward functions) to learn a reward function for each cluster. This decoupling allows reward learning to operate on coherent behaviors identified purely from behavioral similarity.

4.3 Adaptation to new behaviors

When unseen trajectories arrive, CoMI-IRL adapts through encoder finetuning followed by two-stage clustering to handle embedding drift.

4.3.1 Encoder Finetuning

To incorporate unseen behaviors and limit catastrophic forgetting, we restructure the embedding space to map new trajectories while preserving the geometry induced by past data. This is done by finetuning the BE on the old and new data with the addition of a stability regularizer to limit the latent drift of the learned structure. Specifically, we adopt a *less-forget constraint* (Hou et al., 2019) loss, which preserves the geometric configuration by fixing the original embeddings and computing a novel distillation loss on the new embeddings. This regularizer implicitly implements rehearsal by revisiting previously observed trajectories during finetuning.

Formally, let b_i be the L2-normalized embedding of a rehearsed sample generated by the frozen (reference) encoder, and let a_i be the corresponding embedding produced by the finetuned encoder. The stability loss is defined as: $\mathcal{L}_{\text{stab}} = \frac{1}{N}\sum_{i=1}^{N}\left(1 - a_i^\top b_i\right)$. This loss encourages new embeddings to align with the orientation of the original representation by penalizing cosine dissimilarity.

4.3.2 Finetuned Clustering and New Reward Learning

Despite stability regularization, finetuning can cause embedding drift that invalidates original cluster assignments computed in the pre-finetuning space. We address this with a two-stage clustering procedure (further detailed in Appendix): First, we re-cluster only the seen data leveraging the known expected cluster count K_{baseline} from the baseline space. The algorithm prioritizes configurations producing exactly K_{baseline} clusters, yielding recovered centroids and radii that represent baseline modes in the new embedding space. Online trajectories within a novelty threshold θ of any recovered centroid are assigned to that cluster; those outside all radii are marked as novel candidates. Novel candidates are then sub-clustered via connected component analysis or Leiden clustering, following the same process described in Section 4.2. Recovered clusters retain their baseline agents, novel clusters trigger the training of new policies.

5 Experiments

In our experiments, we first analyze how the relationship between the assumed (K) and true (K^*) number of behavioral modes (i.e., $K \neq K^*$,$K = K^*$) impacts clustering and reward learning. Second, we visualize the embedding space and analyze the dataset as partitioned by our model. Finally, we test the adaptability of our approach to new behaviors.

Our main baseline is Ess-InfoGAIL (Fu et al., 2023), one of the current state-of-the-art reward-first coupled methods for MI-IRL which requires a small amount of labeled data and K fixed a priori. While methods inferring K exist, they are often not computationally feasible on continuous spaces (cf. Section 2). We also include baselines combining different clustering algorithms (K-Means (Lloyd, 1982; MacQueen, 1967), and our graph-based clustering) as versions of our behavior-first approach without contrastive embeddings, some requiring a fixed number of clusters K and others inferring it. We test CoMI-IRL and each of the baseline methods using Generative Adversarial Imitation Learning (GAIL) (Ho & Ermon, 2016) and Adversarial IRL (AIRL) (Fu et al., 2017) as reward learning algorithms, with an exception for Ess-InfoGAIL, as it is based on GAIL and not easily transferable to AIRL. We deploy all methods on five MuJoCo environments: Reacher-v4, Pusher-v4, Walker2D-v4, Hopper-v4 and HalfCheetah-v4. For each environment (described in more detail, including objectives and setup, in the Appendix), we apply RL to pre-train K^* expert policies, each corresponding to K^* different behavior modes, following Fu et al. (2023), and for each policy we sample a set of 100 expert demonstrations.

We test all approaches over five seeds, reporting the clustering performance through Normalized Mutual Information (NMI) (Xu et al., 2003), Adjusted Rand Index (ARI) (Rand, 1971), Silhouette Score (Sil) (Rousseeuw, 1987), and the reward learning performance through Average Task Reward (ATR) (Fu et al., 2023). NMI quantifies the correlation between two clusterings (in this case, the ground truth and the result of the clustering phase), while ARI measures their similarity. Both assume values between 0 and 1, with a higher value indicating a better performance. Sil measures how well-separated and distinct clusters are in an unsupervised clustering analysis, and ranges from -1 to 1. To calculate the ATR of each method in each specific environment, we generate 100 rollouts for each behavioral mode using each of the considered methods, and we normalize all the results related to the expert policy, meaning that an ATR of 1 corresponds to the learned policy behaving as well as the expert one. In Appendix C, we report all the configurations and hyperparameters used for each experiment and for each method. We also report computational considerations (which show comparable running times between CoMI-IRL and Ess-InfoGAIL) in Appendix F.

5.1 Clustering and Reward Analysis

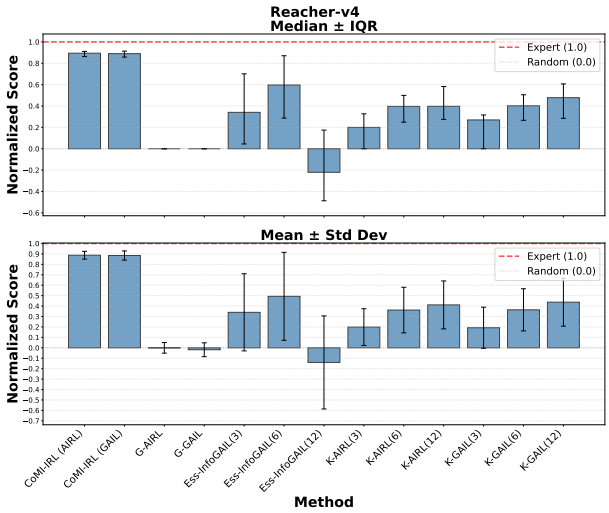

Figure 2: Median±IQR and Mean±Std for Reacher.

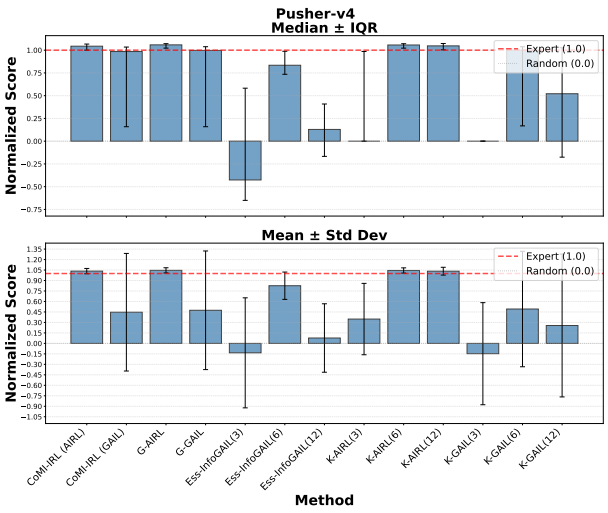

Figure 3: Median±IQR and Mean±Std for Pusher

In this experiment, we show how the relationship between K and K^* impacts clustering and reward learning. Ess-InfoGAIL and KMeans-based baselines(K-GAIL/AIRL) require fixed K, while the Graph-based baseline (G-GAIL/AIRL) and CoMI-IRL infer it from data. As Ess-InfoGAIL requires a small subset of labeled demonstrations (1% of the unlabeled demonstrations) for each behavioral mode, in the $K<K^*$ case we are simulating a scenario in which the labeling underspecifies the number of latent behaviors. To do so, we remove the labeled trajectories for three modes out of six, but we keep the entire unlabeled dataset. For $K>K^*$ case, we duplicate the labeled trajectories, mimicking an overspecification in the number of latent behaviors.

In Table 1, we report the mean and standard deviation of the clustering quality metrics (NMI, ARI, Sil) for each model in each environment. In Figures 2 to 6, we report the reward learning performance in terms of ATR. For each model on each environment, we report the mean±standard deviation, and we also include the median±interquartile range (IQR), to account for potential outlier across seeds and modes and provide a more comprehensive evaluation. For all conditions, we test the models on the modes they have been trained on, either fixed or inferred.

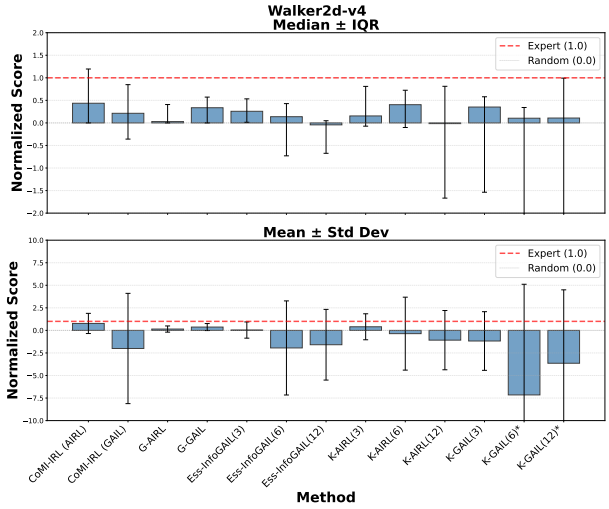

Figure 4: Median±IQR and Mean±Std for Walker2d.

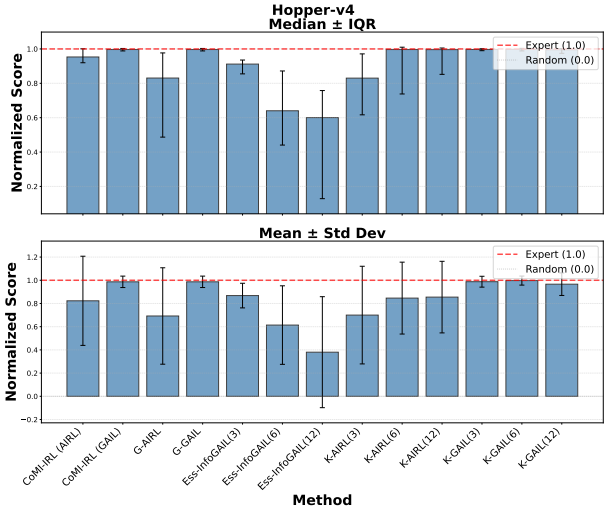

Figure 5: Median±IQR and Mean±Std for Hopper.

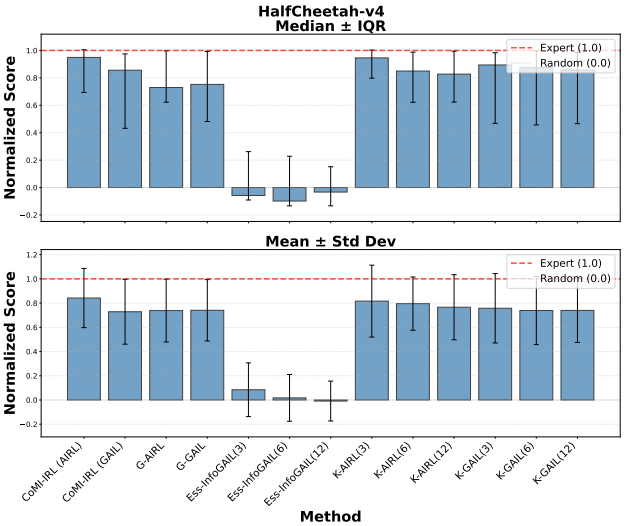

Figure 6: Median±IQR and Mean±Std for HalfCheetah.

Table 1: Clustering performance of all model categories across the considered environments. Results are reported as mean ± standard deviation over five seeds. Higher values are better for all metrics, with best results bolded. For Reacher and Pusher, $K^* = 6$, while for Walker, Hopper and HalfCheetah $K^* = 3$.

Metric	Model	Reacher	Pusher	Walker	Hopper	HalfCheetah
NMI ↑	(K=3)-GAIL/AIRL	0.16 ± 0.02	0.72 ± 0.00	0.53 ± 0.02	$\mathbf{1.00 \pm 0.00}$	0.71 ± 0.06
	(K=6)-GAIL/AIRL	0.43 ± 0.04	$\mathbf{1.00 \pm 0.00}$	0.45 ± 0.03	0.76 ± 0.00	0.73 ± 0.00
	(K=12)-GAIL/AIRL	0.41 ± 0.01	0.87 ± 0.01	0.45 ± 0.02	0.66 ± 0.01	0.68 ± 0.02
	G-GAIL/AIRL	0.01 ± 0.00	$\mathbf{1.00 \pm 0.00}$	0.50 ± 0.00	$\mathbf{1.00 \pm 0.00}$	0.90 ± 0.00
	(K=3) Ess-InfoGAIL	0.53 ± 0.03	0.74 ± 0.06	0.45 ± 0.07	$\mathbf{1.00 \pm 0.00}$	0.76 ± 0.06
	(K=6) Ess-InfoGAIL	0.68 ± 0.03	0.90 ± 0.04	0.36 ± 0.08	0.58 ± 0.01	0.62 ± 0.06
	(K=12) Ess-InfoGAIL	0.54 ± 0.03	0.62 ± 0.07	0.34 ± 0.04	0.42 ± 0.04	0.35 ± 0.11
	CoMI-IRL (Ours)	$\mathbf{0.84 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$	$\mathbf{0.78 \pm 0.04}$	$\mathbf{1.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$
ARI ↑	(K=3)-GAIL/AIRL	0.08 ± 0.01	0.48 ± 0.00	0.48 ± 0.05	$\mathbf{1.00 \pm 0.00}$	0.66 ± 0.10
	(K=6)-GAIL/AIRL	0.26 ± 0.02	$\mathbf{1.00 \pm 0.00}$	0.33 ± 0.05	0.58 ± 0.01	0.66 ± 0.00
	(K=12)-GAIL/AIRL	0.21 ± 0.02	0.75 ± 0.05	0.30 ± 0.03	0.41 ± 0.01	0.51 ± 0.08
	G-GAIL/AIRL	0.00 ± 0.00	$\mathbf{1.00 \pm 0.00}$	0.45 ± 0.00	$\mathbf{1.00 \pm 0.00}$	0.93 ± 0.00
	(K=3) Ess-InfoGAIL	0.45 ± 0.03	0.69 ± 0.08	0.40 ± 0.09	$\mathbf{1.00 \pm 0.00}$	0.67 ± 0.13
	(K=6) Ess-InfoGAIL	0.57 ± 0.05	0.78 ± 0.05	0.25 ± 0.07	0.36 ± 0.04	0.45 ± 0.06
	(K=12) Ess-InfoGAIL	0.30 ± 0.04	0.33 ± 0.06	0.15 ± 0.04	0.13 ± 0.03	0.17 ± 0.12
	CoMI-IRL (Ours)	$\mathbf{0.63 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$	$\mathbf{0.81 \pm 0.05}$	$\mathbf{1.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$
Sil ↑	(K=3)-GAIL/AIRL	0.09 ± 0.00	0.16 ± 0.00	0.06 ± 0.02	0.63 ± 0.00	0.31 ± 0.00
	(K=6)-GAIL/AIRL	0.08 ± 0.00	0.15 ± 0.00	0.04 ± 0.01	0.09 ± 0.00	0.41 ± 0.00
	(K=12)-GAIL/AIRL	0.08 ± 0.00	0.06 ± 0.01	0.02 ± 0.02	0.20 ± 0.01	0.33 ± 0.06
	G-GAIL/AIRL	0.12 ± 0.00	0.15 ± 0.00	0.10 ± 0.00	0.63 ± 0.00	0.30 ± 0.00
	(K=3) Ess-InfoGAIL	0.11 ± 0.03	0.37 ± 0.08	0.03 ± 0.01	0.18 ± 0.02	0.26 ± 0.16
	(K=6) Ess-InfoGAIL	0.03 ± 0.01	0.41 ± 0.06	0.00 ± 0.02	0.03 ± 0.14	0.22 ± 0.09
	(K=12) Ess-InfoGAIL	-0.06 ± 0.03	0.17 ± 0.04	-0.06 ± 0.04	0.30 ± 0.18	0.22 ± 0.07
	CoMI-IRL (Ours)	$\mathbf{0.87 \pm 0.05}$	$\mathbf{0.98 \pm 0.01}$	$\mathbf{0.59 \pm 0.02}$	$\mathbf{0.91 \pm 0.08}$	$\mathbf{0.88 \pm 0.06}$

The results show how, without the embeddings, decoupling can lead to high performance in case of $K{=}K^*$ and clearly separable behavioral modes (such as in the Pusher, Hopper and HalfCheetah cases, where K-Means with $K{=}K^*$ and Graph clustering have perfect NMI/ARI, and ATR comparable with CoMI-IRL), with the simple behavior-first methods outperforming Ess-InfoGAIL. When trajectories are not clearly separable in the original space (such as in the Reacher and Walker2d cases) or $K{\neq}K^*$, clustering performance drops. This degrades the reward learning, causing mode collapse in certain Walker cases, denoted by high variance and a negative ATR. CoMI-IRL, by working on a restructured space, outperforms the baselines both in clustering and reward learning. Ess-InfoGAIL fails on HalfCheetah, likely due to the complexity of the environment. In the clearly separable cases (Pusher and Hopper), it is clear the connection between the clustering quality and ATR performance for Ess-InfoGAIL. In both environments, Ess-InfoGAIL manages to achieve good performance only when $K{=}K^*$, with a clear decline (in Hopper) or collapse (in Pusher) when $K{\neq}K^*$.

In the $K{<}K^*$ case, Ess-InfoGAIL is forced to explain all K^* behavioral modes with K latent codes, meaning that trajectories from other behavioral modes in the dataset will contaminate the training of the K considered classes, as it shows with a low NMI and ATR. In the $K{>}K^*$ case, it is forced to use all latent codes even if the dataset does not require all the modes, also leading to a lower NMI.

We test the statistical significance of the performance through a Wilcoxon signed-rank test (Wilcoxon, 1945) with Benjamini-Hochberg False Discovery Rate (FDR) correction ($\alpha = 0.05$) (Benjamini & Hochberg, 1995) and Cohen's d_z for the paired effect size (Cohen, 2013). The results, presented for each environment in Appendix G, provides evidence supporting the observed ATR performance. CoMI-IRL demonstrates significant advantages over Ess-InfoGAIL in all environments (minimum $p_{\text{FDR}} < 6.5 \times 10^{-15}$, effect sizes $d_z = 1.07$–4.65 across environments). These effect sizes, averaging $d_z = 2.4$ across CoMI-IRL versus Ess-InfoGAIL comparisons, suggest that the differences are statistically significant. Overall, the statistical tests are consistent with the hypothesis that a decoupled behavior-first paradigm offers advantages over a reward-

first coupled one in these evaluated tasks. CoMI-IRL finding K=12 classes suggests that the K^* pre-trained expert policies actually produced a finer-grained behavioral structure with twelve different behavioral modes, and we investigate this further.

5.2 Behavior Visualization

In Figure 7, for each environment, we visualize the original space of trajectories and the respective embedding space built by the BE. For both, we use UMAP (McInnes et al., 2018) as it preserves more of the global structure of the space. By visualizing the embeddings produced by the BE, it is possible to analyze how behaviors relate to each other without introducing any assumption in terms of reward.

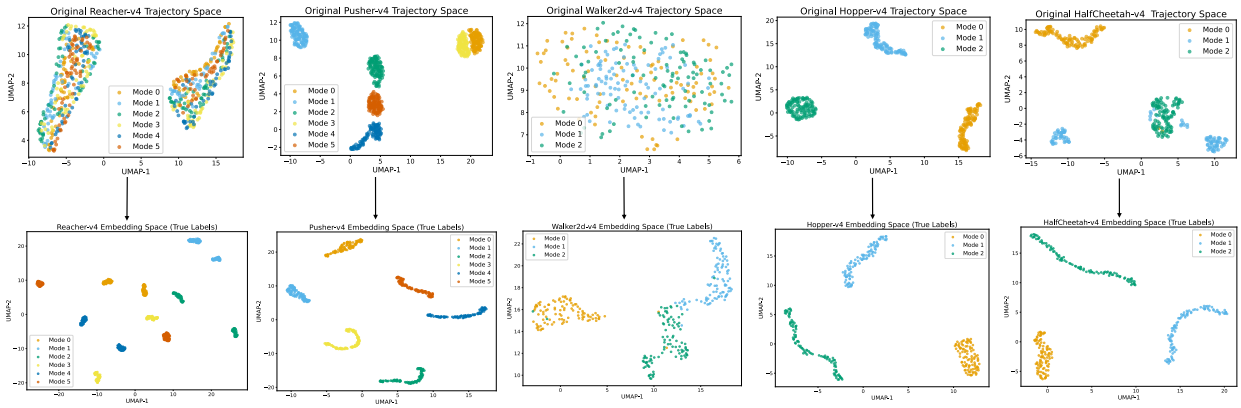

Figure 7: 2D UMAP visualizations of the original trajectory space at the top of each environment (from left to right: Reacher, Pusher, Walker2D, Hopper, HalfCheetah), and the respective embedding space resulting from the BE at the bottom.

As visible in Figure 7, the trajectories of Pusher and Hopper are already fairly separable in the original space, explaining K-GAIL and G-GAIL perfect clustering performance when $K = K^*$. Ess-InfoGAIL approaches the task following the reward-first paradigm and does not consider the inherit similarity between trajectories of the original space. As a result, Ess-InfoGAIL does not manage to perfectly disentangle the modes when the original space is not already separable, highlighting an advantage of the decoupling.

In Reacher, our model notably detects double the amount of true behavioral clusters (twelve instead of six). To further investigate this finer-grained behavioral structure, we analyzed the angular composition of the dataset as partitioned by our method. This analysis revealed that our method successfully disentangled the behavioral modes and discovered sub-modes, creating clusters that are consistent with respect to the configuration of the elbow joint angle (positive or negative), shown in Table 2. The behavior-first perspective discovers what the data actually contains rather than confirming what was assumed, highlighting a concrete advantage of the behavior-first perspective over reward-first methods.

We further analyze this case and evaluate how the trained policies perform in terms of the elbow angle. We generate 50 rollouts for each agent and we analyze the angle consistency. The results are reported in Table 3. CoMI-IRL achieves low variance across all modes, consistently identifying one "Up" and one "Down" agent per mode, allowing for controllability. Ess-InfoGAIL exhibits high variance for Mode 1 ($\pm$0.52 rad), indicating that the latent code fails to capture a consistent physical strategy across seeds. In the other modes the model shows consistency, however this also means that it does not allow for selection of a preferred elbow configuration, collapsing into one configuration per mode that cannot be selected or controlled.

Looking at the MI-IRL task from a behavior-first perspective offers a level of nuance interpretability and controllability not captured by the expert definition of K, and that entangled models such as Ess-InfoGAIL don't permit due to the latent code structuring that reduces their flexibility. Ablation experiments over the loss components (see Appendix E.1) support DIM (cf. Section 3) as a regularization term that improves stability and performance, and ablation experiments over the architecture confirm the pre-processing block

($RFF+MLP+CNN$) as a crucial component on overlapping spaces, highlighting the segment and pairwise losses as key loss components. We also studied the effect of the Jacobian reweighting (see Appendix E.2, which showed a small impact in well-conditioned learned spaces (such as Reacher or Pusher), where the number of increased edges is between 1%and5%. On Walker, which has more heterogeneous action effects, it leads to a decrease of edges between clusters, and a 13% increase of within-cluster edges.

Table 2: **Disentanglement of elbow angle.** CoMI-IRL splits each target mode into two distinct strategy clusters: "Elbow-Down" ($\theta < 0$) and "Elbow-Up" ($\theta > 0$). Values are in radians (Mean ± SD).

Mode	Elbow-Down		Elbow-Up	
	ID	θ	ID	θ
0	C8	-0.92 ± 0.14	C4	$+1.96 \pm 0.30$
1	C7	-0.77 ± 0.19	C5	$+1.30 \pm 0.24$
2	C10	-1.18 ± 0.31	C6	$+0.80 \pm 0.19$
3	C11	-1.99 ± 0.24	C2	$+0.90 \pm 0.20$
4	C9	-2.08 ± 0.18	C1	$+1.62 \pm 0.24$
5	C0	-1.63 ± 0.20	C3	$+2.16 \pm 0.22$

Table 3: **Strategy Consistency.** Comparison of learned elbow angles (θ in radians, Mean ± SD). CoMI-IRL consistently identifies two distinct strategies per mode, whereas Ess-InfoGAIL collapses them, leading to high variance (e.g., Mode 1).

	CoMI-IRL (Ours)		Ess-InfoGAIL
Mode	Down (θ)	Up (θ)	Angle (θ)
0	-0.85 ± 0.02	$+1.79 \pm 0.04$	$+1.90 \pm 0.03$
1	-0.71 ± 0.03	$+1.15 \pm 0.03$	$\mathbf{+0.21 \pm 0.52}$
2	-1.09 ± 0.01	$+0.71 \pm 0.01$	-0.18 ± 0.18
3	-1.82 ± 0.05	$+0.84 \pm 0.02$	-1.86 ± 0.06
4	-2.02 ± 0.04	$+1.51 \pm 0.07$	-1.89 ± 0.17
5	-1.54 ± 0.06	$+1.95 \pm 0.08$	$+1.94 \pm 0.04$

5.3 Adaptation to new behaviors

In this experiment, we evaluate CoMI-IRL's adaptability to new behaviors. Entangled methods cannot adapt to unseen behaviors without full retraining. We consider the case of an initial dataset containing $K_{\text{base}} = K^*/2$ behavioral modes, with the other half received at a later moment. We hereby show how CoMI-IRL does not need to retrain the entire framework, but it is adaptable to unseen behaviors and maintains previously learned knowledge. For Reacher and Pusher, we remove half of the modes from the original dataset, apply CoMI-IRL for the remaining ones, and then finetune adding the new behaviors. For Walker-2D, Hopper and HalfCheetah, which only have three modes (forward, backwards, still), we remove the third mode. We report NMI, ARI, Silhouette and ATR (using CoMI-IRL(AIRL)) for K_{base} and $K_{\text{finetuned}}$ in Table 4.

The results show how CoMI-IRL adapts to new behaviors through finetuning. In the first four environments, CoMI-IRL manages to maintain performance between stages. The finetuned model correctly identifies K on Reacher and Pusher, but it oversegments on Hopper and HalfCheetah, even though performance was not significantly hurt.

On Walker-2D, NMI/ARI are high on K_{base} as the first two modes (moving forward or backward) are separable. The two-stage clustering allows the model to more selectively isolate novel trajectories, leading to higher NMI/ARI and ATR on $K_{\text{finetuned}}$ compared to the full dataset experiment. In this case, the oversegmentation of the new mode leads to an unbalanced set of clusters, dividing the new trajectories into a big and a small cluster. The performance of the agent trained on the small cluster leads to higher standard deviation, while the performance of the agent trained on the big cluster raise the mean ATR.

6 Limitations & Future Work

While results are encouraging, there are limitations. CoMI-IRL suffers from known Transformers limitations (Sanford et al., 2023), relying on the learned space quality and hyperparameter choice. Nonetheless, visualizing the embedding space allows for a qualitative analysis of the results before the IRL phase. The clustering algorithm also plays a determinant role, and our method automatically finds a stable structure for the space considered. A deeper analysis of the Jacobian edge reweighting hyperparameters is part of future studies. While CoMI-IRL has been shown to adapt, future work includes a deeper study on the scalability to multiple modes and the robustness to catastrophic forgetting. The stability regularizer with rehearsal

Table 4: **Adaptation Performance.** NMI, ARI, Silhouette, and ATR scores before (K_{base}) and after ($K_{\text{finetuned}}$) adapting to unseen behaviors. K^* denotes the total number of modes in the final dataset and $\hat{K}$ the number of inferred ones.

Stage	NMI	ARI	Sil	ATR	$\hat{\mathbf{K}}$
Reacher-v4 ($K^* = 6$)					
K_{base}	0.76±0.00	0.57±0.00	0.85±0.00	0.93±0.11	6±0.00
$K_{\text{finetuned}}$	0.84±0.00	0.63±0.00	0.81±0.00	0.93±0.10	12±0.00
Pusher-v4 ($K^* = 6$)					
K_{base}	1.00±0.00	1.00±0.00	0.76±0.00	1.10±0.14	3±0.00
$K_{\text{finetuned}}$	1.00±0.00	1.00±0.00	0.76±0.00	1.09±0.12	6±0.00
Hopper-v4 ($K^* = 3$)					
K_{base}	1.00±0.00	1.00±0.00	1.00±0.00	0.97±0.06	2±0.00
$K_{\text{finetuned}}$	0.87±0.01	0.84±0.01	0.86±0.03	0.93±0.05	5±0.00
HalfCheetah-v4 ($K^* = 3$)					
K_{base}	1.00±0.00	1.00±0.00	1.00±0.00	0.96±0.13	2±0.00
$K_{\text{finetuned}}$	0.90±0.02	0.87±0.03	0.86±0.03	0.91±0.10	4±0.00
Walker2D-v4 ($K^* = 3$)					
K_{base}	0.98±0.02	0.99±0.01	0.54±0.05	0.57±0.31	2±0.00
$K_{\text{finetuned}}$	0.90±0.01	0.88±0.01	0.25±0.04	0.87±0.62	4±0.00

mechanism and the structural isolation given by the decoupling suggest a good degree of robustness, but further work is needed. Other future directions include a higher focus on the reward learning phase, taking into account the connection between clusters representing different ways of achieving the same task. Finally, interpretability results need to be assessed on a user study with real-world data, as real-world scenarios might have more subtle differences, with a focus on understanding the fine-grained distinctions and their relevance. This could also be explored in combination with potentially introducing human feedback in the process, to study the alignment between demonstrated behavioral modes and stated intentions.

7 Conclusions

In this work, we highlighted how the current literature approaches the MI-IRL problem following a reward-first paradigm: by asking which reward function, if followed, would most likely have generated each trajectory in a dataset, without considering the intrinsic relationships between behaviors. We presented CoMI-IRL, a transformer-based unsupervised framework for MI-IRL which follows a behavior-first paradigm and decouples behavior clustering from reward learning: it first learns trajectory embeddings, clusters them, and then applies single-reward IRL to each discovered cluster. In our experiments on the MuJoCo benchmark, CoMI-IRL produced better clustering performance and downstream policy performance compared to the baseline. We analyzed how the relationship between K and K^* impacts existing methods, while our approach infers K from data and shows adaptability to unseen behaviors. Decoupling also allowed us to analyze the dataset as partitioned by our model in terms of behavioral similarity and not through the learned rewards, facilitating the interpretation of similarities in the dataset. Beyond the empirical improvements, CoMI-IRL represents a shift from a reward-first paradigm, in which behavioral structure is discovered through reward-generating likelihood, to a behavior-first paradigm, in which reward learning follows from behavioral structure discovered through dynamic similarity. This shift is conceptually prior to the specific architectural choices made here, and opens directions for MI-IRL research such as unsupervised behavioral analysis, sub-mode discovery, and adaptation to new behaviors without reward re-specification.

References

Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proc. of ICML*, volume 69 of *ACM International Conference Proceeding Series*, 2004. doi: 10.1145/1015330.1015430. URL https://doi.org/10.1145/1015330.1015430.

Javier Almingol and Luis Montesano. Learning multiple behaviours using hierarchical clustering of rewards. In *2015 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pp. 4608–4613, 2015. doi: 10.1109/IROS.2015.7354033.

Monica Babes, Vukosi N. Marivate, et al. Apprenticeship learning about multiple intentions. In *Proc. of ICML*, pp. 897–904, 2011. URL https://icml.cc/2011/papers/478_icmlpaper.pdf.

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300, 1995.

Ariyan Bighashdel, Panagiotis Meletis, et al. Deep adaptive multi-intention inverse reinforcement learning. In *Proc. PKDD, volume*, volume 12975 of *LNCS*, pp. 206–221, 2021. doi: 10.1007/978-3-030-86486-6_13. URL https://doi.org/10.1007/978-3-030-86486-6_13.

Ariyan Bighashdel, Pavol Jancura, and Gijs Dubbelman. Model-free inverse reinforcement learning with multi-intention, unlabeled, and overlapping demonstrations. *Mach. Learn.*, 112(7):2263–2296, 2023. doi: 10.1007/S10994-022-06273-X. URL https://doi.org/10.1007/s10994-022-06273-x.

Benjamin M Bolstad, Rafael A Irizarry, et al. A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, 19(2):185–193, 2003. doi: https://doi.org/10.1093/bioinformatics/19.2.185.

Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Proc. PAKDD*, pp. 160–172, 2013. doi: 10.1007/978-3-642-37456-2_14.

Yanchuan Chang, Jianzhong Qi, et al. Contrastive trajectory similarity learning with dual-feature attention. In *2023 IEEE 39th Int. Conf. on data engineering (ICDE)*, pp. 2933–2945. IEEE, 2023. doi: https://doi.org/10.1109/ICDE55515.2023.00224.

Lili Chen, Kevin Lu, et al. Decision transformer: Reinforcement learning via sequence modeling. In *Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 15084–15097, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/7f489f642a0ddb10272b5c31057f0663-Abstract.html.

Ting Chen, Simon Kornblith, et al. A simple framework for contrastive learning of visual representations. In *Proc. of ICML*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1597–1607, 2020. URL http://proceedings.mlr.press/v119/chen20j.html.

Jaedeug Choi and Kee-Eung Kim. Nonparametric bayesian inverse reinforcement learning for multiple reward functions. In *Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 314–322, 2012. URL https://proceedings.neurips.cc/paper/2012/hash/140f6969d5213fd0ece03148e62e461e-Abstract.html.

Jaedeug Choi and Kee-Eung Kim. Bayesian nonparametric feature construction for inverse reinforcement learning. In *Proc. IJCAI*, pp. 1287–1293, 2013. URL http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6572.

Jacob Cohen. *Statistical power analysis for the behavioral sciences*. routledge, 2013.

Xiaozhi Deng, Dong Huang, et al. Strongly augmented contrastive clustering. *Pattern Recognition*, 139: 109470, 2023. doi: http://dx.doi.org/10.1016/j.patcog.2023.109470.

Jacob Devlin, Ming-Wei Chang, et al. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL-HLT*, pp. 4171–4186, Minneapolis, Minnesota, 2019. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

Chelsea Finn, Paul Christiano, et al. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *CoRR*, 2016. doi: https://doi.org/10.48550/arXiv.1611.03852.

Huiqiao Fu, Kaiqiang Tang, et al. Ess-infogail: Semi-supervised imitation learning from imbalanced demonstrations. In *Adv. Neural Inf. Process. Syst. (NIPS)*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/bcf26768143c94bd36e363cd4bf5daf0-Abstract-Conference.html.

Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. *ArXiv preprint*, abs/1710.11248, 2017. URL https://arxiv.org/abs/1710.11248.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple contrastive learning of sentence embeddings. In *Proc. of EMNLP*, pp. 6894–6910, Online and Punta Cana, Dominican Republic, 2021. doi: 10.18653/v1/2021.emnlp-main.552. URL https://aclanthology.org/2021.emnlp-main.552.

Zichang Ge, Changyu Chen, et al. On learning informative trajectory embeddings for imitation, classification and regression. 2025.

Adam Gleave, Mohammad Taufeeque, et al. imitation: Clean imitation learning implementations, 2022. URL https://arxiv.org/abs/2211.11972.

Karol Hausman, Yevgen Chebotar, et al. Multi-modal imitation learning from unstructured demonstrations using generative adversarial nets. In *Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 1235–1245, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/632cee946db83e7a52ce5e8d6f0fed35-Abstract.html.

R. Devon Hjelm, Alex Fedorov, et al. Learning deep representations by mutual information estimation and maximization. In *Proc. of ICLR*, 2019. URL https://openreview.net/forum?id=Bklr3j0cKX.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 4565–4573, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/cc7e2b878868cbae992d1fb743995d8f-Abstract.html.

Saihui Hou, Xinyu Pan, et al. Learning a unified classifier incrementally via rebalancing. In *IEEE Conference on CVPR*, pp. 831–839, 2019. doi: 10.1109/CVPR.2019.00092. URL http://openaccess.thecvf.com/content_CVPR_2019/html/Hou_Learning_a_Unified_Classifier_Incrementally_via_Rebalancing_CVPR_2019_paper.html.

Yiding Jiang, Evan Zheran Liu, et al. Learning options via compression. In *Adv. Neural Inf. Process. Syst. (NIPS)*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/8567a53e58a9fa4823af356c76ed943c-Abstract-Conference.html.

Kalle Kujanpää, Daulet Baimukashev, et al. Discrete contrastive learning for diffusion policies in autonomous driving. 2025.

Sergey Levine, Zoran Popovic, and Vladlen Koltun. Feature construction for inverse reinforcement learning. In *Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 1342–1350, 2010. URL https://proceedings.neurips.cc/paper/2010/hash/a8f15eda80c50adb0e71943adc8015cf-Abstract.html.

Sergey Levine, Zoran Popovic, and Vladlen Koltun. Nonlinear inverse reinforcement learning with gaussian processes. In *Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 19–27, 2011. URL https://proceedings.neurips.cc/paper/2011/hash/c51ce410c124a10e0db5e4b97fc2af39-Abstract.html.

Yang Li, Si Si, et al. Learnable fourier features for multi-dimensional spatial positional encoding. In *Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 15816–15829, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/84c2d4860a0fc27bcf854c444fb8b400-Abstract.html.

Yunzhu Li, Jiaming Song, and Stefano Ermon. Infogail: Interpretable imitation learning from visual demonstrations. In *Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 3812–3822, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/2cd4e8a2ce081c3d7c32c3cde4312ef7-Abstract.html.

Stuart Lloyd. Least squares quantization in pcm. *IEEE trans.s on information theory*, 28(2):129–137, 1982. doi: http://dx.doi.org/10.1109/TIT.1982.1056489.

J MacQueen. Multivariate observations. In *Proc. ofthe 5th Berkeley Symposium on Mathematical Statisticsand Probability*, volume 1, pp. 281–297, 1967.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. 2018.

Bernard Michini and Jonathan P. How. Bayesian nonparametric inverse reinforcement learning. In *Proc ECML PKDD*, volume 7524 of *Lecture Notes in Computer Science*, pp. 148–163, 2012. doi: 10.1007/978-3-642-33486-3_10. URL `https://doi.org/10.1007/978-3-642-33486-3_10`.

Andrew Y. Ng and Stuart J. Russell. Algorithms for inverse reinforcement learning. In *Proc. of ICML*, pp. 663–670, 2000.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. 2018.

Wei Pan, Xiaotong Shen, and Binghui Liu. Cluster analysis: unsupervised learning via supervised learning with a non-convex penalty. *The J. of Machine Learning Research*, 14(1):1865–1889, 2013.

Guanghui Qin, Yukun Feng, and Benjamin Van Durme. The NLP task effectiveness of long-range transformers. In *Proceedings of European Chapter of the ACL*, pp. 3774–3790, Dubrovnik, Croatia, 2023. doi: 10.18653/v1/2023.eacl-main.273. URL `https://aclanthology.org/2023.eacl-main.273`.

Antonin Raffin, Ashley Hill, et al. Stable-baselines3: Reliable reinforcement learning implementations. *J. Mach. Learn. Res.*, 22:268:1–268:8, 2021. URL `http://jmlr.org/papers/v22/20-1364.html`.

Nasim Rahaman, Aristide Baratin, et al. On the spectral bias of neural networks. In *Proc. of ICML*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5301–5310, 2019. URL `http://proceedings.mlr.press/v97/rahaman19a.html`.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 1177–1184, 2007. URL `https://proceedings.neurips.cc/paper/2007/hash/013a006f03dbc5392effeb8f18fda755-Abstract.html`.

Siddharthan Rajasekaran, Jinwei Zhang, and Jie Fu. Inverse reinforce learning with nonparametric behavior clustering. *ArXiv preprint*, abs/1712.05514, 2017. URL `https://arxiv.org/abs/1712.05514`.

Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *Proc. IJCAI*, pp. 2586–2591, 2007.

Giorgia Ramponi, Amarildo Likmeta, et al. Truly batch model-free inverse reinforcement learning about multiple intentions. In *Proc. AISTATS*, volume 108 of *Proceedings of Machine Learning Research*, pp. 2359–2369, 2020. URL `http://proceedings.mlr.press/v108/ramponi20a.html`.

William M Rand. Objective criteria for the evaluation of clustering methods. *J. of the American Statistical association*, 66(336):846–850, 1971.

Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. of computational and applied mathematics*, 20:53–65, 1987.

Clayton Sanford, Daniel J Hsu, and Matus Telgarsky. Representational strengths and limitations of transformers. *Advances in Neural Information Processing Systems*, 36:36677–36707, 2023.

Aaron J Snoswell, Surya PN Singh, and Nan Ye. Limiirl: lightweight multiple-intent inverse reinforcement learning. 2021.

Matthew Tancik, Pratul P. Srinivasan, et al. Fourier features let networks learn high frequency functions in low dimensional domains. In *Adv. Neural Inf. Process. Syst. (NIPS)*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/55053683268957697aa39fba6f231c68-Abstract.html.

Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. *J. of the royal statistical society: series b (statistical methodology)*, 63(2):411–423, 2001. doi: http://dx.doi.org/10.1111/1467-9868.00293.

Vincent A. Traag, Ludo Waltman, and Nees Jan van Eck. From louvain to leiden: guaranteeing well-connected communities. *ArXiv preprint*, abs/1810.08473, 2018. URL https://arxiv.org/abs/1810.08473.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 5998–6008, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Abhishek Vivekanandan, Christian Hubschneider, and J Marius Zöllner. Contrast & compress: Learning lightweight embeddings for short trajectories. 2025.

Ao Wang, Fengyuan Sun, et al. [cls] token tells everything needed for training-free efficient mllms. 2024.

Junhui Wang. Consistent selection of the number of clusters via crossvalidation. *Biometrika*, 97(4):893–904, 2010. doi: http://dx.doi.org/10.1093/biomet/asq061.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *Proc. of ICML*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9929–9939, 2020. URL http://proceedings.mlr.press/v119/wang20k.html.

Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945. ISSN 00994987. URL http://www.jstor.org/stable/3001968.

Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. Maximum entropy deep inverse reinforcement learning. 2015.

Da Xu, Chuanwei Ruan, Evren Körpeoglu, Sushant Kumar, and Kannan Achan. Self-attention with functional time representation learning. In *Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 15889–15899, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/cf34645d98a7630e2bcca98b3e29c8f2-Abstract.html.

Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. *Proc. of the 26th annual Int. ACM SIGIR Conf. on Research and development in informaion retrieval*, 2003. doi: http://dx.doi.org/10.1145/860435.860485.

Jianqiao Zheng, Sameera Ramasinghe, et al. Trading positional complexity vs deepness in coordinate networks. In *European Conf. on Computer Vision*, pp. 144–160. Springer, 2022. doi: http://dx.doi.org/10.1007/978-3-031-19812-0_9.

Hao Zhu, Brice De La Crompe, Gabriel Kalweit, Artur Schneider, Maria Kalweit, Ilka Diester, and Joschka Boedecker. Multi-intention inverse q-learning for interpretable behavior representation. *Trans. Mach. Learn. Res.*, 2024, 2024. URL https://openreview.net/forum?id=hrKHkmLUFk.

Brian D. Ziebart, Andrew L. Maas, et al. Maximum entropy inverse reinforcement learning. In *Proc. of AAAI*, pp. 1433–1438, 2008.

Yixiong Zou, Shuai Yi, Yuhua Li, and Ruixuan Li. A closer look at the CLS token for cross-domain few-shot learning. In *Adv. Neural Inf. Process. Syst. (NIPS)*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/9b77f07301b1ef1fe810aae96c12cb7b-Abstract-Conference.html.

Appendix

In this document, we provide further material regarding the CoMI-IRL paper, including details on the environments (Section A), the Behavioral Encoder architecture (Section B), hyperparameters C and the graph-based clustering algorithm (Section D). We provide further results in Section E, including an ablation study on the architecture and loss components introduced in the main paper, and further considerations on the effect of the Jacobian Reweighting. Finally, we report the average wallclock time of CoMI-IRL and Ess-InfoGAIL for the considered environments.

A Environments

Reacher consists a 2-DoF robot arm which goal is to move the robot's end effector close to a target position. Pusher consists a 4-DoF robot arm which goal is to move a target cylinder to a goal position using the robot's end effector. Both environments have six goals and the goal location has been removed from the observation Fu et al. (2023). Walker-2D is a 6-DoF bipedal robot consisting of two legs and feet attached to a common base which goal is to make coordinate both sets of legs and feet to move the robot in the right direction, including running forward, running backward and balancing. HalfCheetah is a 6-DoF 2D robot which goal is to apply torque to the joints to make the cheetah run in the right direction (forward or backward) or balacing in place. Hopper is a 3-DoF one-legged robot which goal is to make hops that move the robot either forward, backward, or stay balanced in place. An illustration of each environment is provided in Figure 8.

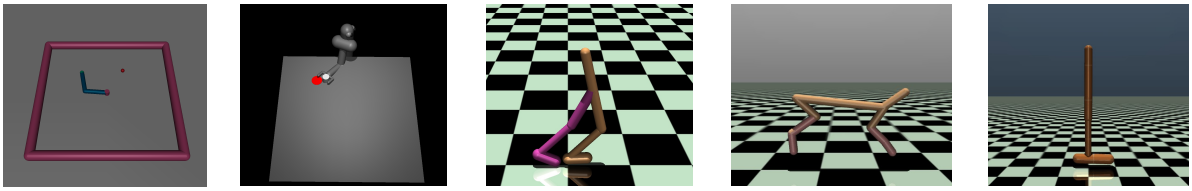

Figure 8: **Left to Right:** Reacher, Pusher, Walker2d, HalfCheetah, Hopper.

B Additional Architecture Details

We presented the architecture of the Behavioral Encoder (BE) in Section 3.1 of the main paper. As trajectories might contain high-frequency elements, and Multi-Layer Perceptrons ($MLPs$) knowingly struggle with high-frequency functions Rahaman et al. (2019), we use Random Fourier Features with Gaussian mapping (RFF) Rahimi & Recht (2007); Xu et al. (2019); Zheng et al. (2022); Li et al. (2021). In each head, this mapping injects a rich set of sinusoidal features at multiple random orientation and frequencies to overcome known MLP spectral bias towards low-frequency functions so that, for each $s^j_{\tau_i} \in S_{\tau_i}$ and $a^j_{\tau_i} \in A_{\tau_i}$,

$$RFF(s^j_{\tau_i}) = [\sin(2\pi s^j_{\tau_i} W), \cos(2\pi s^j_{\tau_i} W)]^T, W \sim \mathcal{N}(0, \sigma^2), \tag{6}$$

with σ chosen as a hyperparameter. The output is subsequently processed by a shallow MLP, and then followed by a 1D (CNN) to capture short-range temporal patterns within the behavioral trajectory, ensuring local coherency. Our approach benefits from this technique both for the internal MLPs in the Transformer, and because RFF can be seen as a generalized case of the sinusoidal positional encoding Tancik et al. (2020) needed by the transformer to have information on the order of the sequence.

C Implementation Details

In this section, we report the hyperparameters used in our experiments for AIRL and GAIL (Table 5), for Ess-InfoGAIL (Table 6), and for the encoder (Table 7). The agents are learned using the imitation library Gleave et al. (2022) and Stable-Baselines3 Raffin et al. (2021), while we use the original repository for Ess-InfoGAIL.

AIRL Hyperparameters
Buffer: batch=256, capacity=2048
Disc: updates=32, reward_hid=(32,), pot_hid=(32,32)
PPO: lr=3e-4, batch=64, ent=0.01, γ=0.99
clip=0.2, epochs=10
Training: 250k timesteps/cluster (1.5M for Walker2d-v4)

Table 5: AIRL configuration. Disc=discriminator, pot_hid=potential hidden sizes.

Hyperparameter	Value	Hyperparameter	Value
use_obs_norm	True	lr_actor	3e-3
auto_lr	True	lr_critic	3e-3
rollout_length	5000	lr_prior	3e-3
batch_size	1000	lr_disc	5e-3
num_steps	2000000	lr_q	1e-2
eval_interval	200000	epoch_ppo	20
obs_horizon	8	epoch_disc	50
obs_his_steps	1	epoch_prior	1
im_ratio	1	surrogate_loss_coef	4.0
disc_grad_penalty	0.1	value_loss_coef	5.0
disc_coef	20	us_coef	1.0
ss_coef	4.0	info_max_coef1	3.0
info_max_coef2	0.05	info_max_coef3	0.5
begin_weight	20	reward_i_coef	1.0
reward_us_coef	0.1	reward_ss_coef	0.1
reward_t_coef	0.005		

Table 6: Hyperparameters used for Ess-InfoGAIL.

Param	Reach	Push	Walk	Hopp	HC
Shared: emb=32, heads=4, layers=2, hid=1024					
drop=0.1, m_s=64, m_a=32					
Loss: $\alpha/\beta/\gamma/\delta$=0.5/1.0/0.5/1.0; epochs=100					
lr	1e-4	1e-4	5e-4	1e-4	1e-4
inp_ch	6	20	17	11	17
max_len	50	100	100	100	100
batch	64	32	64	64	64
σ_s	0.01	0.001	0.001	0.01	0.01
σ_a	0.1	0.001	0.001	0.01	0.01

Table 7: BE hyperparameters. Reach=Reacher-v4, Push=Pusher-v4, Walk=Walker2d-v4, Hopp=Hopper-v4, HC=HalfCheetah-v4. Shared parameters use abbreviated names: emb=embedding dim, heads=attention heads, hid=hidden dim, drop=dropout, m_s/m_a=Gaussian means (state/action), σ_s/σ_a=Gaussian sigmas.

D Graph-Based Clustering Description

This appendix provides detailed descriptions of the graph-based clustering methods used in CoMI-IRL: the joint k-resolution sweep for the main clustering and the two-stage target-aware clustering for post-finetuning adaptation.

D.1 Joint k-Resolution Sweep

After training the behavior encoder, we cluster trajectory embeddings using graph-based community detection. Our approach operates on the graph structure of the embedding space, making it suitable for continuous behavioral manifolds where modes may not have clear density gaps.

Graph Construction Given N trajectory, we take the L2-normalized embeddings $\{z_i\}_{i=1}^N$ and we construct a weighted k-nearest neighbor graph $\mathcal{G} = (V, E, W)$. For each node i, we find its k nearest neighbors using cosine distance and assign RBF kernel weights based on cosine similarity:

$$w_{ij} = \exp\left(\frac{\mathrm{sim}(z_i, z_j)}{\sigma}\right), \quad \mathrm{sim}(z_i, z_j) = 1 - d_{\cos}(z_i, z_j) \tag{7}$$

where $d_{\cos}(z_i, z_j) = 1 - \tilde{z}_i^\top \tilde{z}_j$ and $\sigma = 1.0$. We make the graph undirected by taking the maximum weight: $w_{ij} = \max(w_{ij}, w_{ji})$.

Automatic Structure Detection Before applying community detection, we check if the graph naturally decomposes into isolated components. For $k \in \{15, 30, 50, 75\}$, we compute the connected components of the k-NN graph. If multiple significant components exist (size $\geq$ `min_cluster_size`), use them directly as clusters, otherwise proceed to the joint sweep. This handles environments where modes form distinct, well-separated islands in the embedding space (e.g., Reacher-v4, Pusher-v4).

Joint k-Resolution Sweep For clusters spread out on a continuous manifold (e.g., Walker2d-v4), we jointly optimize over the graph connectivity parameter k and the Leiden resolution parameter γ, following Algorithm 1.

Algorithm 1 Joint k-Resolution Sweep

Require: Embeddings Z, range $[k_{\min}, k_{\max}]$, resolution set Γ, min cluster size m
Ensure: Cluster labels, optimal (k^*, γ^*)
$\mathcal{R} \leftarrow \emptyset$ {Results collection}
for $k \in \mathrm{linspace}(k_{\min}, k_{\max}, n_k)$ **do**
  Build k-NN graph $\mathcal{G}_k$
  for $\gamma \in \Gamma$ **do**
    $\ell_{k,\gamma} \leftarrow$ Leiden clustering with resolution γ
    Filter clusters with size $< m$ as noise
    $\mathcal{R} \leftarrow \mathcal{R} \cup \{(k, \gamma, \ell_{k,\gamma})\}$
  end for
end for
Stability-based Selection:
for each $(k, \gamma) \in \mathcal{R}$ **do**
  $\mathcal{N}_{k,\gamma} \leftarrow$ neighbors with $|k' - k| \leq 15$ or $|\gamma' - \gamma| \leq 0.3$
  $\mathrm{stability}_{k,\gamma} \leftarrow \frac{1}{|\mathcal{N}_{k,\gamma}|} \sum_{(k',\gamma') \in \mathcal{N}_{k,\gamma}} \mathrm{ARI}(\ell_{k,\gamma}, \ell_{k',\gamma'})$
end for
$(k^*, \gamma^*) \leftarrow \arg\max_{k,\gamma} \mathrm{stability}_{k,\gamma}$
return ℓ_{k^*,γ^*}, (k^*, γ^*)

The resolution set used is $\Gamma = \{0.01, 0.025, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.5, 0.7, 1.0, 1.5, 2.0\}$. Of the tested environments, Pusher and Reacher did not require the joint sweep as their structure already decomposed into isolated components. Further details are provided in Section F.

Leiden Community Detection The Leiden algorithm Traag et al. (2018) optimizes modularity with a resolution parameter:

$$Q_\gamma = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \gamma \frac{k_i k_j}{2m}\right] \delta(c_i, c_j) \tag{8}$$

where A_{ij} is the adjacency weight, $k_i = \sum_j A_{ij}$ is the weighted degree, $m = \frac{1}{2}\sum_{ij} A_{ij}$ is the total edge weight, c_i is the community of node i, and γ controls granularity (higher values yield more clusters).

Rather than selecting by a single metric, we use partition stability as the primary criterion. For each (k, γ) configuration, we compute the average Adjusted Rand Index (ARI) between its partition and those of neighboring configurations in parameter space. Configurations whose partitions are robust to small parameter perturbations receive higher stability scores. This approach avoids over-fitting to specific parameter choices and identifies genuine cluster structure.

Jacobian-Based Edge Reweighting For environments with continuous behavioral manifolds, embedding similarity alone may not separate subtly distinct modes. In all environments, we augment edge weights with Jacobian-based behavioral features. For each trajectory, we compute:

- State differences: $\Delta s_t = s_{t+1} - s_t$
- Control sensitivity: $\gamma_t = \|\Delta s_t\|/(\|a_t\| + \epsilon)$
- Covariance-based features from $\mathrm{Cov}(\Delta s, a)$

The resulting 8-dimensional feature vector captures:

- Mean/std/max of control sensitivity
- Temporal variance of sensitivity
- Top singular value and ratio from state-action covariance
- Mean/std of action magnitude

Given these statistical features $\{b_i\}$, we update edge weights:

$$w_{ij}^{\text{new}} = w_{ij} \cdot [1 + \alpha \cdot (2 \cdot b_{ij} - 1)] \tag{9}$$

where $w_{ij}^{\text{behav}} = \exp(-\|b_i - b_j\|^2/2\sigma_b^2)$ and $\alpha \in [0, 1]$ controls the behavioral contribution, which we set at 0.3. We modulate edge weights by behavioral similarity: edges between trajectories with similar dynamics $b_{ij}>0.5$ are strengthened, while edges between behaviorally dissimilar trajectories $b_{ij}<0.5$ are weakened.

Before deciding to use Jacobian features, we test if they are redundant with the learned embeddings by computing the Pearson and Spearman correlation between embedding-based and Jacobian-based pairwise similarities. If the average correlation exceeds 0.7, Jacobian features are skipped as the encoder already captures the relevant dynamics. However, all tests showed an average correlation lower than 0.45, meaning that the Jacobian features are capturing different information.

After clustering, we build a registry storing per-cluster metadata, which is used for novel detection:

$$\text{Registry}[k] = \left\{ \begin{array}{l} \mu_k = \frac{1}{|C_k|}\sum_{i \in C_k} \tilde{z}_i \quad \text{(L2-normalized centroid)} \\ r_k^{95} = \text{quantile}_{95}\left(\{d_{\cos}(\tilde{z}_i, \mu_k)\}_{i \in C_k}\right) \quad \text{(radius)} \\ n_k = |C_k| \quad \text{(count)} \end{array} \right\} \tag{10}$$

D.2 Two-Stage Clustering Details

After finetuning the encoder on combined seen and online data, the embedding space shifts due to adaptation. This invalidates baseline cluster labels computed in the pre-finetuning space. We address this with a two-stage procedure that leverages knowledge of the expected baseline structure.

Let f_θ denote the baseline encoder and $f_{\theta'}$ the finetuned encoder. For a seen trajectory τ, we can denote the baseline embedding as $z = f_\theta(\tau)$ and the finetuned embedding as $z' = f_{\theta'}(\tau)$.

Despite stability regularization, $z \neq z'$. More critically, the relative positions of clusters may change. A key factor is that we know the expected number of clusters K_{baseline} from before the finetuning. Even though embeddings have drifted, the relative separability of seen modes is typically preserved. We re-cluster *only the seen data* $Z_{\text{seen}} = \{z'_i\}_{i=1}^{N_{\text{seen}}}$ using target-aware selection, following Algorithm 2.

Algorithm 2 Stage 1: Target-Aware Recovery

Require: Finetuned seen embeddings Z_{seen}, target K_{baseline}
Ensure: Recovered labels, centroids, radii

```
// Check for isolated components first
for k ∈ {15, 30, 50, 75} do
    Build k-NN graph, compute connected components
    Count significant components (size ≥ m)
    if n_significant = K_baseline then
        Use components as recovered clusters
        return
    end if
end for
// Target-aware (k, γ) sweep
R ← ∅, E ← ∅ {All results, exact matches}
for (k, γ) ∈ SearchGrid do
    Run Leiden, filter small clusters
    n_c ← number of valid clusters
    Compute silhouette s
    R ← R ∪ {(k, γ, n_c, s, labels)}
    if n_c = K_baseline then
        E ← E ∪ {|R| − 1}
    end if
end for
// Selection
if E ≠ ∅ then
    Select from E with best silhouette
else
    Score = −2 · |n_c − K_baseline| + s
    Select configuration with best score
end if
Compute recovered centroids {μ_k^rec} and radii {r_k^rec}
return labels, centroids, radii
```

The scoring function $-2 \cdot |n_c - K| + s$ heavily penalizes deviation from the target cluster count while still rewarding cluster quality via silhouette. Using the recovered cluster structure from the first stage, we assign online trajectories following Algorithm 3.

Algorithm 3 Stage 2: Anchored Assignment

Require: Online embeddings Z_{online}, recovered centroids $\{\mu_k^{\text{rec}}\}$, radii $\{r_k^{\text{rec}}\}$, threshold θ
Ensure: Labels for online data, novel cluster IDs

```
for each online embedding z do
    d_k ← 1 − z^⊤ μ_k^rec / (‖z‖ ‖μ_k^rec‖) for all k
    if min_k d_k ≤ θ · r_k^rec then
        Assign to nearest cluster: ℓ ← arg min_k d_k
    else
        Mark as novel candidate
    end if
end for
// Sub-cluster novel candidates
if n_novel_candidates ≥ m then
    Build k-NN graph on novel candidates
    Compute connected components
    if multiple components then
        Each component → separate novel cluster
    else
        Run Algorithm 1 with restricted grid search
    end if
end if
return labels, novel_cluster_ids
```

The threshold θ controls sensitivity to novel modes. We set $\theta = 0.1$ on Reacher and Pusher, as clusters are well-separated, while we set $\theta = 0.05$ on Walker, as the continuous manifold creates the risk of over-assignment to existing clusters at mode boundaries. The recovered radius r_k^{rec} is computed as the 95th percentile of within-cluster cosine distances, expanded by factor 1.2 to account for minor embedding shifts.

Output Structure The two-stage procedure produces:

- **Recovered clusters** (IDs $0, \ldots, K_{\text{baseline}} - 1$): Correspond to baseline modes in the new space. Previously trained IRL agents are directly assigned.
- **Novel clusters** (IDs $\geq K_{\text{baseline}}$): Contain trajectories from genuinely new behavioral modes. New IRL agents are trained for these.

D.3 Hyperparameters

	Description	Value
$k_{\min}$	Min neighbors for graph	$\max(5, N/50)$
$k_{\max}$	Max neighbors for graph	$\min(100, N/3)$
σ	RBF kernel bandwidth	1.0
m	Min cluster size	$\max(5, 0.02 \cdot N)$
α_{behav}	Jacobian weight	0.3
θ_{novelty}	Novelty threshold	0.05–0.1
r_k^{95} exp	Factor for recovered radii	1.2

Table 8: Graph clustering hyperparameters

E Additional Results

In this Appendix, we provide additional results for a more comprehensive overview.

E.1 Ablations

To validate the architectural choices of the pre-processing block composed by *RFF+MLP+CNN* (called preBlock in the results) within the BE, we run an ablation study over 3 seeds. To validate the loss composition of global contrastive, segment contrastive and DIM Hjelm et al. (2019) regularization, we run an ablations over 10 seeds, reporting in Table 9 the clustering quality of each version. While the versions with and without the DIM regularization are very similar (BE+CS and BE+DCS), we use the full version (BE+DCS) in our experiments as it showed higher stability and slightly better performance in most cases. Since Hopper and HalfCheetah are clearly separable, we skip their ablations for now.

Table 9: **Ablation Study.** Impact of architecture (removing the *RFF+MLP+CNN* preprocessing) over 3 seeds and loss components on clustering quality over 10 seeds. Loss keys: **C**=Global Contrastive, **D**=DIM, **S**=Segment/Pairwise.

Configuration	NMI	ARI	Sil
	Reacher-v4		
Architecture			
BE (Full)	0.84±0.00	0.63±0.00	0.87±0.06
BE \ (preBlock)	0.36±0.32	0.21±0.23	0.29±0.25
Loss Design			
BE + C	0.44±0.10	0.26±0.08	0.39±0.07
BE + DC	0.46±0.14	0.27±0.10	0.37±0.09
BE + CS	0.82±0.04	0.60±0.06	0.82±0.10
BE + DCS	**0.83**±0.01	**0.6**2±0.02	**0.84**±0.06
	Pusher-v4		
Architecture			
BE (Full)	1.0±0.00	1.0±0.00	0.99±0.01
BE \ (preBlock)	1.0±0.00	1.0±0.00	0.94±0.01
Loss Design			
BE + C	0.85±0.10	0.77±0.18	0.47±0.10
BE + DC	0.81±0.22	0.73±0.32	0.50±0.16
BE + CS	0.99±0.01	0.99±0.01	0.97±0.02
BE + DCS	**1.0**±0.00	**1.0**±0.00	**0.98**±0.01
	Walker2d-v4		
Architecture			
BE (Full)	0.75±0.05	0.77±0.07	0.60±0.01
BE \ (preBlock)	0.73±0.05	0.69±0.10	0.37±0.02
Loss Design			
BE + C	0.42±0.04	0.32±0.04	0.49±0.04
BE + DC	0.42±0.03	0.34±0.06	0.49±0.05
BE + CS	**0.80**±0.06	**0.82**±0.06	0.57±0.04
BE + DCS	0.79±0.06	0.80±0.10	**0.57±0.03**

E.2 Jacobian Reweighting Effect

For each environment, we present the effects of applying the Jacobian reweighing of graph edges in terms of edge weight. While for Pusher and Reacher (Figure 9 and Figure 10) the impact is less prominent due to the already well-conditioned structure of the two spaces, so that the number of increased edges is very small (1% for Pusher and 5% for Reacher), in Walker (Figure 11) the reweighting has a higher effect, since it has more heterogeneous action effects on the different features. Between clusters, most edges are decrease, noticeable by the lighter color shade, while the 13% of increased edges are mostly within-cluster edges. On Hopper and HalfCheetah, where the average correlation is high and redundancy has been detected, we have no Jacobian reweighing of graph edges.

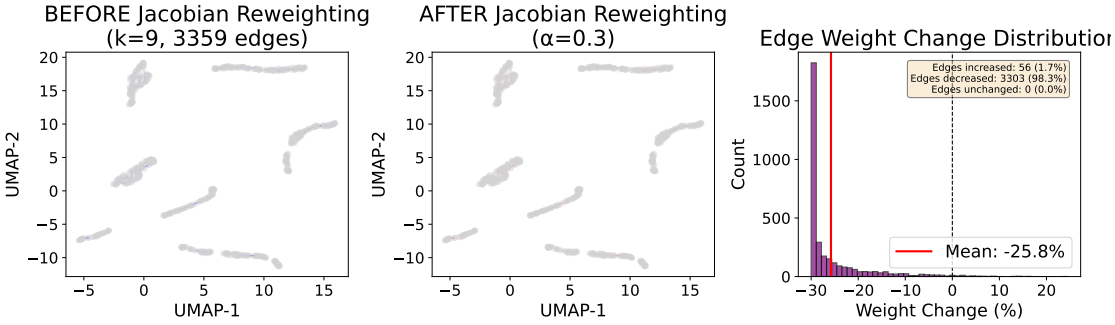

Figure 9: Jacobian Edge Reweighting on Pusher.

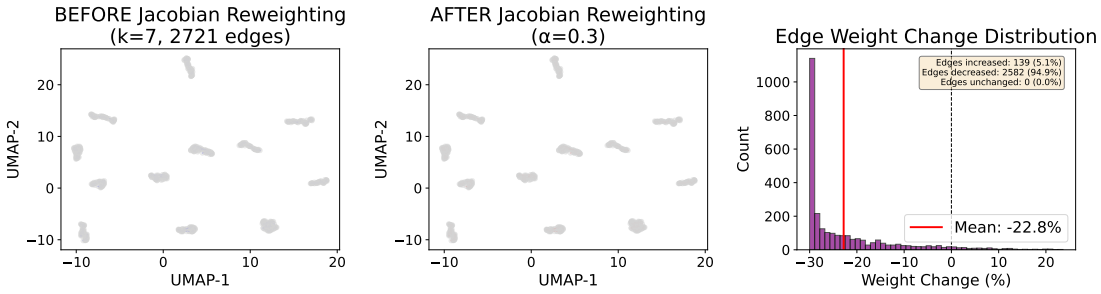

Figure 10: Jacobian Edge Reweighting on Reacher.

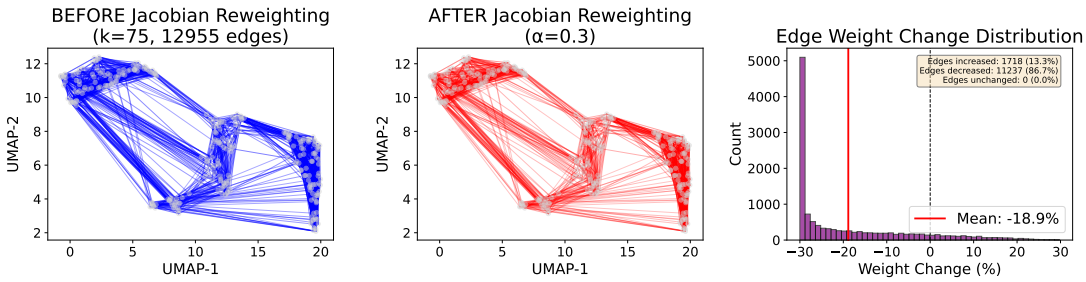

Figure 11: Jacobian Edge Reweighting on Walker.

F Computational considerations

We ran our experiments on a 2020 MacBook Pro with 2 GHz Quad-Core Intel Core i5. In Section F, we report the average wall-clock times of Ess-InfoGAIL and CoMI-IRL (divided by phase) for each environment. It is important to notice that while the BE's training on GPU would take much less (~ 5 minutes), transformers training time also increases with the amount of training samples in the dataset.

	Reacher	**Pusher**	**Walker**	**Hopper**	**HalfCheetah**
CoMI-IRL					
BE	~ 16	~ 25	~ 17	~ 18	~ 18
Clustering (s)	~ 0	~ 0	~ 20	~ 0	~ 0
IRL	~ 14	~ 14	~ 48	~ 20	~ 48
Total	~ 30	~ 39	~ 65	~ 38	~ 66
Ess-InfoGAIL					
$K = 3$	~ 35	~ 39	~ 55	~ 48	~ 38
$K = 6$	~ 41	~ 46	~ 76	~ 57	~ 47
$K = 12$	~ 48	~ 58	~ 103	~ 70	~ 58

Table 10: Wall-clock time for the different components of CoMI-IRL and total, and the Ess-InfoGAIL experiments with different values of K. While all values are reported in minutes, the clustering values are reported in seconds. In CoMI-IRL's IRL phase, the reported time is the time of a single cluster, as it's fully parallelizable.

G Statistical Significance

In Tables 11 to 15, we present the statistical significance pairwise comparison, ordered by significance according to the Wilcoxon signed-rank with Benjamini-Hochberg FDR correction, with $\alpha = 0.05$. We report the effect size through Cohen's d_z, where positive values indicate Method 1 outperforms Method 2. The comparisons between CoMI-IRL and Ess-InfoGAIL are bolded.

Method 1	Method 2	$p_{\mathbf{FDR}}$	d_z
CoMI-IRL (AIRL)	G-AIRL	$2.5e-18$	13.02
CoMI-IRL (AIRL)	K-GAIL(K=12)	$2.5e-18$	2.23
CoMI-IRL (AIRL)	K-AIRL(K=6)	$2.5e-18$	3.93
CoMI-IRL (AIRL)	**Ess-InfoGAIL(K=3)**	$2.5e-18$	4.52
CoMI-IRL (AIRL)	**Ess-InfoGAIL(K=12)**	$2.5e-18$	2.83
CoMI-IRL (AIRL)	K-GAIL(K=3)	$2.5e-18$	7.83
CoMI-IRL (AIRL)	K-GAIL(K=6)	$2.5e-18$	4.03
CoMI-IRL (AIRL)	G-GAIL	$2.5e-18$	11.93
CoMI-IRL (AIRL)	K-AIRL(K=3)	$2.5e-18$	8.17
CoMI-IRL (GAIL)	K-GAIL(K=12)	$2.5e-18$	2.24
CoMI-IRL (GAIL)	K-AIRL(K=6)	$2.5e-18$	3.86
CoMI-IRL (GAIL)	G-AIRL	$2.5e-18$	12.35
CoMI-IRL (GAIL)	**Ess-InfoGAIL(K=3)**	$2.5e-18$	4.65
CoMI-IRL (GAIL)	**Ess-InfoGAIL(K=12)**	$2.5e-18$	2.86
CoMI-IRL (GAIL)	K-GAIL(K=3)	$2.5e-18$	8.20
CoMI-IRL (GAIL)	K-GAIL(K=6)	$2.5e-18$	3.97
CoMI-IRL (GAIL)	G-GAIL	$2.5e-18$	12.45
CoMI-IRL (GAIL)	K-AIRL(K=3)	$2.5e-18$	8.61
CoMI-IRL (GAIL)	K-AIRL(K=12)	$2.5e-18$	2.38
CoMI-IRL (AIRL)	K-AIRL(K=12)	$2.6e-18$	2.37
CoMI-IRL (AIRL)	**Ess-InfoGAIL(K=6)**	$2.9e-18$	2.43
CoMI-IRL (GAIL)	**Ess-InfoGAIL(K=6)**	$4.4e-18$	2.37
G-GAIL	K-AIRL(K=12)	$4.2e-12$	-0.92
K-GAIL(K=12)	G-GAIL	$5.5e-12$	0.95
K-AIRL(K=12)	G-AIRL	$4.2e-11$	0.85
K-GAIL(K=12)	G-AIRL	$6.0e-11$	0.87
Ess-InfoGAIL(K=12)	K-GAIL(K=12)	$3.8e-9$	-0.72
Ess-InfoGAIL(K=12)	K-AIRL(K=12)	$5.8e-9$	-0.71
K-GAIL(K=12)	K-AIRL(K=3)	$1.2e-8$	0.70
K-GAIL(K=3)	K-GAIL(K=12)	$1.4e-8$	-0.70
K-AIRL(K=3)	K-AIRL(K=12)	$3.7e-8$	-0.67
K-GAIL(K=3)	K-AIRL(K=12)	$5.1e-8$	-0.67
K-GAIL(K=6)	G-GAIL	$3.3e-7$	0.57
G-GAIL	K-AIRL(K=6)	$3.4e-7$	-0.56
Ess-InfoGAIL(K=6)	G-GAIL	$5.3e-6$	0.49
Ess-InfoGAIL(K=3)	K-GAIL(K=12)	$8.0e-6$	-0.55
Ess-InfoGAIL(K=6)	Ess-InfoGAIL(K=12)	$8.0e-6$	0.50
Ess-InfoGAIL(K=3)	K-AIRL(K=12)	$9.4e-6$	-0.52
K-GAIL(K=6)	G-AIRL	$1.1e-5$	0.50
K-AIRL(K=6)	G-AIRL	$1.4e-5$	0.49
Ess-InfoGAIL(K=12)	K-GAIL(K=6)	$1.4e-5$	-0.49
Ess-InfoGAIL(K=12)	K-AIRL(K=6)	$1.6e-5$	-0.49
G-GAIL	K-AIRL(K=3)	$1.5e-4$	-0.38
Ess-InfoGAIL(K=6)	G-AIRL	$2.4e-4$	0.43
Ess-InfoGAIL(K=3)	G-GAIL	$3.6e-4$	0.36
K-GAIL(K=3)	G-GAIL	$5.1e-4$	0.36
K-GAIL(K=6)	K-GAIL(K=12)	$7.6e-4$	-0.36
K-GAIL(K=12)	K-AIRL(K=6)	$8.9e-4$	0.36
Ess-InfoGAIL(K=3)	Ess-InfoGAIL(K=12)	$1.1e-3$	0.34
K-GAIL(K=6)	K-AIRL(K=3)	$1.2e-3$	0.33
K-GAIL(K=3)	K-GAIL(K=6)	$1.6e-3$	-0.33
K-AIRL(K=3)	K-AIRL(K=6)	$1.7e-3$	-0.32
Ess-InfoGAIL(K=6)	K-GAIL(K=3)	$1.8e-3$	0.33
Ess-InfoGAIL(K=6)	K-AIRL(K=3)	$2.0e-3$	0.33
K-GAIL(K=3)	K-AIRL(K=6)	$2.4e-3$	-0.32
Ess-InfoGAIL(K=12)	K-AIRL(K=3)	$2.7e-3$	-0.31
Ess-InfoGAIL(K=12)	K-GAIL(K=3)	$2.7e-3$	-0.31
K-AIRL(K=3)	G-AIRL	$2.7e-3$	0.30
K-AIRL(K=6)	K-AIRL(K=12)	$4.0e-3$	-0.33
K-GAIL(K=6)	K-AIRL(K=12)	$5.0e-3$	-0.33
K-GAIL(K=3)	G-AIRL	$6.9e-3$	0.28
Ess-InfoGAIL(K=12)	G-AIRL	$1.3e-2$	-0.23
Ess-InfoGAIL(K=3)	**Ess-InfoGAIL(K=6)**	$1.7e-2$	-0.24
Ess-InfoGAIL(K=3)	G-AIRL	$2.0e-2$	0.26

Table 11: Pairwise agent comparison on Reacher-v4 (100 episodes/mode). Wilcoxon signed-rank test with Benjamini–Hochberg FDR correction ($\alpha = 0.05$). All rows shown have $p_{\mathrm{FDR}} < 0.05$. d_z is Cohen's d_z (paired effect size); positive values indicate Method 1 outperforms Method 2.

Method 1	Method 2	$p_{\mathbf{FDR}}$	d_z
Ess-InfoGAIL(K=12)	G-AIRL	$5.7e-18$	-2.67
K-AIRL(K=3)	G-AIRL	$5.7e-18$	-4.40
CoMI-IRL (AIRL)	**Ess-InfoGAIL(K=6)**	$5.7e-18$	2.05
CoMI-IRL (AIRL)	K-GAIL(K=3)	$5.7e-18$	3.85
Ess-InfoGAIL(K=6)	G-AIRL	$5.7e-18$	-2.07
K-GAIL(K=3)	G-AIRL	$5.7e-18$	-4.03
CoMI-IRL (AIRL)	**Ess-InfoGAIL(K=12)**	$5.7e-18$	2.61
CoMI-IRL (AIRL)	**Ess-InfoGAIL(K=3)**	$5.7e-18$	3.60
Ess-InfoGAIL(K=3)	G-AIRL	$5.7e-18$	-3.65
CoMI-IRL (AIRL)	K-AIRL(K=3)	$6.4e-18$	4.19
K-GAIL(K=6)	G-AIRL	$1.3e-17$	-1.86
CoMI-IRL (AIRL)	K-GAIL(K=6)	$1.8e-17$	1.83
K-GAIL(K=12)	G-AIRL	$1.5e-16$	-1.15
CoMI-IRL (AIRL)	K-GAIL(K=12)	$4.2e-16$	1.15
K-AIRL(K=6)	G-AIRL	$1.4e-15$	-1.57
CoMI-IRL (AIRL)	K-AIRL(K=6)	$8.9e-15$	1.55
CoMI-IRL (GAIL)	G-AIRL	$5.3e-14$	-0.72
G-GAIL	G-AIRL	$6.7e-14$	-0.68
CoMI-IRL (AIRL)	CoMI-IRL (GAIL)	$4.0e-12$	0.70
CoMI-IRL (AIRL)	G-GAIL	$4.6e-12$	0.67
Ess-InfoGAIL(K=3)	K-AIRL(K=12)	$8.7e-12$	-1.06
K-GAIL(K=3)	K-AIRL(K=12)	$1.4e-11$	-1.09
Ess-InfoGAIL(K=12)	K-AIRL(K=12)	$5.5e-11$	-0.90
K-AIRL(K=12)	G-AIRL	$5.0e-10$	-0.88
K-GAIL(K=12)	K-AIRL(K=12)	$2.0e-8$	-0.52
K-AIRL(K=3)	K-AIRL(K=12)	$2.2e-8$	-0.88
K-GAIL(K=3)	G-GAIL	$2.3e-8$	-0.63
CoMI-IRL (GAIL)	K-GAIL(K=3)	$5.0e-8$	0.60
CoMI-IRL (AIRL)	K-AIRL(K=12)	$5.2e-8$	0.86
Ess-InfoGAIL(K=6)	K-AIRL(K=12)	$6.6e-8$	-0.42
Ess-InfoGAIL(K=3)	G-GAIL	$2.6e-7$	-0.56
CoMI-IRL (GAIL)	**Ess-InfoGAIL(K=3)**	$7.7e-7$	0.52
K-GAIL(K=3)	K-AIRL(K=6)	$9.0e-7$	-0.60
Ess-InfoGAIL(K=3)	K-AIRL(K=6)	$1.0e-6$	-0.58
K-GAIL(K=6)	K-AIRL(K=12)	$1.6e-6$	-0.55
G-GAIL	K-AIRL(K=3)	$1.9e-6$	0.46
K-GAIL(K=6)	G-GAIL	$3.2e-6$	-0.44
CoMI-IRL (GAIL)	K-AIRL(K=3)	$3.9e-6$	0.43
Ess-InfoGAIL(K=6)	K-AIRL(K=6)	$5.2e-6$	-0.43
Ess-InfoGAIL(K=6)	K-GAIL(K=3)	$1.8e-5$	0.55
Ess-InfoGAIL(K=3)	**Ess-InfoGAIL(K=6)**	$2.2e-5$	-0.53
CoMI-IRL (GAIL)	K-GAIL(K=6)	$2.8e-5$	0.40
Ess-InfoGAIL(K=12)	G-GAIL	$4.0e-5$	-0.43
CoMI-IRL (GAIL)	**Ess-InfoGAIL(K=12)**	$7.1e-5$	0.40
K-GAIL(K=12)	G-GAIL	$1.0e-4$	-0.33
K-GAIL(K=6)	K-AIRL(K=6)	$1.3e-4$	-0.32
CoMI-IRL (GAIL)	K-GAIL(K=12)	$2.8e-4$	0.30
Ess-InfoGAIL(K=12)	K-AIRL(K=6)	$3.5e-4$	-0.41
K-AIRL(K=3)	K-AIRL(K=6)	$6.0e-4$	-0.45
Ess-InfoGAIL(K=6)	G-GAIL	$1.2e-3$	-0.26
Ess-InfoGAIL(K=6)	**Ess-InfoGAIL(K=12)**	$1.4e-3$	0.34
K-GAIL(K=3)	K-GAIL(K=6)	$2.4e-3$	-0.29
CoMI-IRL (GAIL)	**Ess-InfoGAIL(K=6)**	$2.6e-3$	0.23
K-GAIL(K=3)	K-GAIL(K=12)	$4.5e-3$	-0.20

Table 12: Pairwise agent comparison on Pusher-v4 (100 episodes/mode). Wilcoxon signed-rank test with Benjamini–Hochberg FDR correction ($\alpha = 0.05$). All rows shown have $p_{\mathrm{FDR}} < 0.05$. d_z is Cohen's d_z (paired effect size); positive values indicate Method 1 outperforms Method 2.

Table 13: Pairwise statistical comparisons for Walker2d-v4 (Benjamini–Hochberg FDR correction, $\alpha = 0.05$). Only significant pairs shown ($p_{\mathrm{FDR}} < 0.05$). d_z = Cohen's d_z (paired effect size).

Method 1	**Method 2**	$p_{\mathbf{FDR}}$	d_z
K-GAIL(6)	G-GAIL	$1.02e-11$	-0.34
Ess-InfoGAIL(12)	G-GAIL	$1.02e-11$	-0.43
Ess-InfoGAIL(3)	G-GAIL	$2.16e-10$	-0.80
K-GAIL(3)	G-GAIL	$2.62e-10$	-0.43
CoMI-IRL (AIRL)	**Ess-InfoGAIL(12)**	$2.87e-10$	0.47
Ess-InfoGAIL(6)	G-GAIL	$6.29e-10$	-0.33
CoMI-IRL (AIRL)	**Ess-InfoGAIL(3)**	$5.68e-9$	0.64
CoMI-IRL (AIRL)	K-GAIL(6)	$6.03e-9$	0.37
CoMI-IRL (AIRL)	K-AIRL(3)	$6.15e-9$	0.57
CoMI-IRL (AIRL)	K-GAIL(3)	$1.16e-8$	0.50
G-GAIL	K-AIRL(3)	$1.16e-8$	0.45
CoMI-IRL (AIRL)	**Ess-InfoGAIL(6)**	$2.57e-8$	0.39
G-GAIL	K-AIRL(12)	$5.96e-8$	0.41
K-GAIL(12)	G-GAIL	$1.55e-7$	-0.39
CoMI-IRL (AIRL)	K-AIRL(12)	$9.68e-7$	0.47
G-GAIL	K-AIRL(6)	$1.71e-6$	0.22
Ess-InfoGAIL(12)	G-AIRL	$3.53e-6$	-0.35
CoMI-IRL (AIRL)	K-AIRL(6)	$7.00e-6$	0.35
CoMI-IRL (AIRL)	G-AIRL	$8.64e-6$	0.53
G-GAIL	G-AIRL	$1.23e-5$	0.55
CoMI-IRL (AIRL)	K-GAIL(12)	$2.04e-5$	0.41
K-GAIL(6)	G-AIRL	$1.55e-4$	-0.30
Ess-InfoGAIL(3)	G-AIRL	$5.65e-4$	-0.31
Ess-InfoGAIL(6)	G-AIRL	$1.21e-3$	-0.24
K-GAIL(3)	G-AIRL	$1.97e-3$	-0.25
Ess-InfoGAIL(12)	K-AIRL(6)	$2.10e-3$	-0.21
K-AIRL(3)	G-AIRL	$2.22e-3$	-0.14
K-GAIL(12)	G-AIRL	$8.15e-3$	-0.35
CoMI-IRL (GAIL)	G-GAIL	$8.81e-3$	-0.40
Ess-InfoGAIL(3)	**Ess-InfoGAIL(12)**	$9.73e-3$	0.30
K-AIRL(12)	G-AIRL	$1.03e-2$	-0.31
CoMI-IRL (GAIL)	K-GAIL(6)	$1.42e-2$	0.01
Ess-InfoGAIL(12)	K-AIRL(3)	$1.50e-2$	-0.31
K-GAIL(6)	K-AIRL(6)	$2.00e-2$	-0.27
Ess-InfoGAIL(12)	K-GAIL(3)	$2.02e-2$	-0.22
CoMI-IRL (AIRL)	**CoMI-IRL (GAIL)**	$3.31e-2$	0.41
Ess-InfoGAIL(6)	K-AIRL(6)	$4.86e-2$	-0.12

Table 14: Pairwise statistical comparisons for Hopper-v4 (Benjamini–Hochberg FDR correction, $\alpha = 0.05$). Only significant pairs shown ($p_{\mathrm{FDR}} < 0.05$). d_z = Cohen's d_z (paired effect size).

Method 1	**Method 2**	$p_{\mathbf{FDR}}$	d_z
Ess-InfoGAIL(12)	G-GAIL	$9.28e-18$	-1.88
CoMI-IRL (GAIL)	**Ess-InfoGAIL(12)**	$9.28e-18$	1.88
Ess-InfoGAIL(3)	G-GAIL	$9.28e-18$	-2.77
CoMI-IRL (GAIL)	**Ess-InfoGAIL(3)**	$9.28e-18$	2.77
G-GAIL	K-AIRL(3)	$9.28e-18$	3.02
CoMI-IRL (GAIL)	K-AIRL(3)	$9.28e-18$	3.02
Ess-InfoGAIL(6)	G-GAIL	$9.81e-18$	-2.43
CoMI-IRL (GAIL)	**Ess-InfoGAIL(6)**	$9.81e-18$	2.43
CoMI-IRL (GAIL)	K-GAIL(3)	$3.81e-17$	2.42
K-GAIL(3)	G-GAIL	$5.21e-17$	-2.42
CoMI-IRL (GAIL)	K-AIRL(6)	$2.76e-16$	1.78
G-GAIL	K-AIRL(6)	$2.76e-16$	1.78
CoMI-IRL (AIRL)	**Ess-InfoGAIL(3)**	$6.46e-15$	1.42
K-GAIL(6)	G-GAIL	$6.06e-15$	-1.54
CoMI-IRL (GAIL)	K-GAIL(6)	$6.06e-15$	1.54
CoMI-IRL (GAIL)	G-AIRL	$6.06e-15$	0.71
G-GAIL	G-AIRL	$6.17e-15$	0.71
CoMI-IRL (AIRL)	**Ess-InfoGAIL(6)**	$2.05e-14$	1.29
K-AIRL(3)	G-AIRL	$6.79e-15$	-1.30
CoMI-IRL (AIRL)	K-GAIL(3)	$1.39e-14$	1.31
CoMI-IRL (AIRL)	**Ess-InfoGAIL(12)**	$1.24e-13$	1.07
Ess-InfoGAIL(3)	G-AIRL	$1.72e-13$	-1.10
K-GAIL(3)	G-AIRL	$6.39e-13$	-1.02
Ess-InfoGAIL(6)	G-AIRL	$2.00e-12$	-0.98
CoMI-IRL (AIRL)	K-AIRL(6)	$3.22e-12$	1.02
Ess-InfoGAIL(12)	G-AIRL	$3.24e-12$	-0.86
G-GAIL	K-AIRL(12)	$2.34e-11$	1.03
CoMI-IRL (GAIL)	K-AIRL(12)	$2.34e-11$	1.03
Ess-InfoGAIL(12)	K-GAIL(12)	$4.46e-11$	-0.73
CoMI-IRL (AIRL)	K-GAIL(6)	$5.40e-11$	0.91
K-AIRL(6)	G-AIRL	$2.31e-10$	-0.78
Ess-InfoGAIL(3)	K-GAIL(12)	$4.84e-10$	-0.63
CoMI-IRL (GAIL)	K-GAIL(12)	$2.23e-09$	0.89
K-GAIL(12)	G-GAIL	$2.30e-09$	-0.89
Ess-InfoGAIL(12)	K-AIRL(12)	$6.43e-09$	-0.57
K-GAIL(12)	K-AIRL(3)	$9.08e-09$	0.69
K-GAIL(6)	G-AIRL	$9.42e-09$	-0.67
Ess-InfoGAIL(3)	K-AIRL(12)	$1.04e-07$	-0.54
K-AIRL(3)	K-AIRL(12)	$1.40e-07$	-0.60
Ess-InfoGAIL(6)	K-GAIL(12)	$4.40e-07$	-0.51
CoMI-IRL (AIRL)	K-AIRL(12)	$5.45e-06$	0.53
Ess-InfoGAIL(6)	K-GAIL(6)	$1.40e-05$	-0.44
Ess-InfoGAIL(6)	K-AIRL(12)	$1.51e-05$	-0.44
Ess-InfoGAIL(3)	K-GAIL(6)	$2.85e-05$	-0.26
K-GAIL(3)	K-GAIL(12)	$2.85e-05$	-0.57
K-GAIL(3)	K-AIRL(12)	$2.86e-05$	-0.49
CoMI-IRL (AIRL)	K-GAIL(12)	$6.90e-05$	0.44
K-GAIL(6)	K-AIRL(3)	$4.34e-04$	0.31
K-AIRL(12)	G-AIRL	$1.22e-03$	-0.33
Ess-InfoGAIL(3)	K-GAIL(3)	$1.23e-03$	-0.29
Ess-InfoGAIL(6)	K-AIRL(6)	$2.75e-03$	-0.27
K-GAIL(3)	K-AIRL(3)	$3.80e-03$	0.22
Ess-InfoGAIL(3)	K-AIRL(6)	$3.80e-03$	-0.21
K-AIRL(3)	K-AIRL(6)	$5.18e-03$	-0.26
K-GAIL(12)	K-AIRL(6)	$5.42e-03$	0.38
K-AIRL(6)	K-AIRL(12)	$8.25e-03$	-0.31
CoMI-IRL (AIRL)	G-AIRL	$2.58e-02$	0.23
K-GAIL(6)	K-GAIL(12)	$3.09e-02$	-0.31
K-GAIL(6)	K-AIRL(12)	$3.48e-02$	-0.24
CoMI-IRL (GAIL)	G-GAIL	$3.82e-02$	-0.14
Ess-InfoGAIL(12)	K-AIRL(3)	$4.78e-02$	0.22

Table 15: Pairwise statistical comparisons for HalfCheetah-v4 (Benjamini–Hochberg FDR correction, $\alpha = 0.05$). Only significant pairs shown ($p_{\text{FDR}} < 0.05$). d_z = Cohen's d_z (paired effect size).

Method 1	Method 2	p_{FDR}	d_z
CoMI-IRL (GAIL)	**Ess-InfoGAIL(12)**	$4.14e{-18}$	2.90
Ess-InfoGAIL(6)	G-GAIL	$4.14e{-18}$	−3.08
CoMI-IRL (AIRL)	**Ess-InfoGAIL(12)**	$4.14e{-18}$	3.59
CoMI-IRL (AIRL)	**Ess-InfoGAIL(6)**	$4.14e{-18}$	3.49
Ess-InfoGAIL(3)	G-GAIL	$4.14e{-18}$	−2.93
CoMI-IRL (AIRL)	**Ess-InfoGAIL(3)**	$4.14e{-18}$	3.35
CoMI-IRL (GAIL)	**Ess-InfoGAIL(3)**	$4.14e{-18}$	2.77
CoMI-IRL (GAIL)	**Ess-InfoGAIL(6)**	$4.14e{-18}$	2.83
Ess-InfoGAIL(6)	G-AIRL	$4.14e{-18}$	−3.06
Ess-InfoGAIL(12)	G-AIRL	$4.14e{-18}$	−3.28
Ess-InfoGAIL(12)	G-GAIL	$4.14e{-18}$	−3.30
Ess-InfoGAIL(3)	G-AIRL	$4.64e{-18}$	−2.78
K-GAIL(3)	G-GAIL	$2.01e{-17}$	−1.77
CoMI-IRL (AIRL)	K-GAIL(3)	$2.01e{-17}$	2.00
CoMI-IRL (AIRL)	K-AIRL(3)	$2.35e{-17}$	1.89
CoMI-IRL (GAIL)	K-GAIL(3)	$3.64e{-17}$	1.67
K-GAIL(3)	G-AIRL	$8.32e{-17}$	−1.69
K-AIRL(3)	G-AIRL	$9.80e{-17}$	−1.63
G-GAIL	K-AIRL(3)	$1.48e{-16}$	1.61
CoMI-IRL (AIRL)	K-AIRL(6)	$1.65e{-16}$	1.41
CoMI-IRL (AIRL)	K-GAIL(6)	$1.96e{-16}$	1.50
CoMI-IRL (GAIL)	K-AIRL(3)	$6.63e{-16}$	1.51
K-GAIL(6)	G-GAIL	$1.33e{-15}$	−1.31
CoMI-IRL (GAIL)	K-GAIL(6)	$3.04e{-15}$	1.24
K-GAIL(6)	G-AIRL	$7.60e{-15}$	−1.23
K-AIRL(6)	G-AIRL	$9.94e{-15}$	−1.17
G-GAIL	K-AIRL(6)	$7.00e{-14}$	1.16
CoMI-IRL (GAIL)	K-AIRL(6)	$1.16e{-13}$	1.11
Ess-InfoGAIL(6)	K-GAIL(12)	$2.74e{-12}$	−0.96
Ess-InfoGAIL(6)	K-AIRL(12)	$6.89e{-12}$	−0.97
Ess-InfoGAIL(3)	K-GAIL(12)	$9.59e{-12}$	−0.94
Ess-InfoGAIL(3)	K-AIRL(12)	$1.45e{-11}$	−0.94
CoMI-IRL (AIRL)	K-GAIL(12)	$1.84e{-11}$	0.93
CoMI-IRL (AIRL)	K-AIRL(12)	$2.49e{-11}$	0.86
Ess-InfoGAIL(12)	K-GAIL(12)	$3.53e{-11}$	−1.05
Ess-InfoGAIL(12)	K-AIRL(12)	$3.53e{-11}$	−1.05
CoMI-IRL (GAIL)	K-GAIL(12)	$2.55e{-09}$	0.73
K-GAIL(12)	G-AIRL	$4.41e{-09}$	−0.69
K-GAIL(12)	G-GAIL	$1.54e{-08}$	−0.77
G-GAIL	K-AIRL(12)	$1.36e{-07}$	0.67
K-AIRL(12)	G-AIRL	$1.65e{-07}$	−0.66
CoMI-IRL (GAIL)	K-AIRL(12)	$1.06e{-06}$	0.62
Ess-InfoGAIL(12)	K-AIRL(6)	$2.21e{-06}$	−0.59
Ess-InfoGAIL(3)	K-AIRL(6)	$2.21e{-06}$	−0.54
Ess-InfoGAIL(12)	K-GAIL(6)	$2.22e{-06}$	−0.56
CoMI-IRL (AIRL)	G-AIRL	$2.56e{-06}$	0.36
Ess-InfoGAIL(6)	K-GAIL(6)	$2.82e{-06}$	−0.59
Ess-InfoGAIL(6)	K-AIRL(6)	$2.82e{-06}$	−0.61
Ess-InfoGAIL(3)	K-GAIL(6)	$3.48e{-06}$	−0.52
K-GAIL(3)	K-AIRL(12)	$6.78e{-06}$	−0.55
K-GAIL(3)	K-GAIL(12)	$1.15e{-05}$	−0.53
K-AIRL(3)	K-AIRL(12)	$5.46e{-05}$	−0.52
K-GAIL(12)	K-AIRL(3)	$1.37e{-04}$	0.50
CoMI-IRL (AIRL)	G-GAIL	$2.74e{-04}$	0.38
CoMI-IRL (AIRL)	**CoMI-IRL (GAIL)**	$3.38e{-04}$	0.38
Ess-InfoGAIL(3)	K-GAIL(3)	$9.12e{-04}$	−0.38
Ess-InfoGAIL(3)	K-AIRL(3)	$1.10e{-03}$	−0.36
Ess-InfoGAIL(12)	K-GAIL(3)	$2.80e{-03}$	−0.37
Ess-InfoGAIL(12)	K-AIRL(3)	$3.83e{-03}$	−0.37
Ess-InfoGAIL(6)	K-GAIL(3)	$5.68e{-03}$	−0.34
K-GAIL(6)	K-AIRL(12)	$5.72e{-03}$	−0.32
Ess-InfoGAIL(6)	K-AIRL(3)	$7.20e{-03}$	−0.34