A. Experimental Details

In this section we describe the environments used in our experiments (see Section 4) and the experiment design.

A.1. Highway

We build on the `highway-v0` task from the `highway-env` traffic simulator (Leurent, 2018). The task is specified by:

1. **State space, $\mathcal{S}$:** The kinematic information of the ego vehicle and the five closest vehicles (ordered from closest to the furthest) is used as the Markov state, i.e., $\mathbf{s}_t = \{[x_t, y_t, \dot{x}_t, \dot{y}_t]\}_{\text{ego, other}_1, \ldots, \text{other}_5} \in \mathbb{R}^{6\times 4}$. The ego-car is illustrated in green and the other cars in blue.
2. **Action space, $\mathcal{A}$:** We use a discrete action space, constructed by K-means clustering of the continuous actions of the intelligent driving model (Kesting et al., 2010). We found out that keeping 9 actions was sufficient, i.e., $\mathbf{a}_t \in \{0, \ldots, 8\}$.
3. **Demonstrations, $\mathcal{D}$:** At each time-step, the ego-car observes online the state-action pairs for the 5 closest cars.

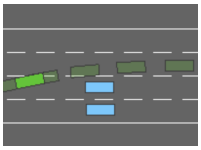


Figure 6. Highway

A.2. Roundabout

We build on the `roundabot-v0` task from the `highway-env` traffic simulator (Leurent, 2018). The task is specified by:

1. **State space, $\mathcal{S}$:** The kinematic information of the ego vehicle and the five closest vehicles (ordered from closest to the furthest) is used as the Markov state, i.e., $\mathbf{s}_t = \{[x_t, y_t, \dot{x}_t, \dot{y}_t]\}_{\text{ego, other}_1, \ldots, \text{other}_3} \in \mathbb{R}^{4\times 4}$. The ego-car is illustrated in green and the other cars in blue.
2. **Action space, $\mathcal{A}$:** We use a discrete action space, constructed by K-means clustering of the continuous actions of the intelligent driving model (Kesting et al., 2010). We found out that keeping 6 actions was sufficient, i.e., $\mathbf{a}_t \in \{0, \ldots, 5\}$.
3. **Demonstrations, $\mathcal{D}$:** At each time-step, the ego-car observes online the state-action pairs for the 3 closest cars.

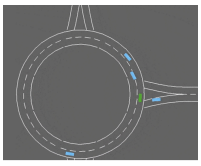


Figure 7. Roundabout

A.3. CoinGrid

We build a simple multi-task grid-world. The task is specified by:

1. **State space, $\mathcal{S}$:** We use a symbolic, multi-channel representation of the 7×7 grid-world (Chevalier-Boisvert et al., 2019): the first three channels specify the presence or absence of the three different coloured boxes, the forth channel was the walls mask and the fifth and last channel was the position and orientation of the agent. We represent the orientation of the agent by 'painting' the cell in front of the agent. Therefore $\mathbf{s}_t \in \{0, 1\}^{7\times 7\times 5}$.
2. **Action space, $\mathcal{A}$:** We use the $\{$`LEFT, RIGHT, FORWARD`$\}$ actions from Minigrid (Chevalier-Boisvert et al., 2018) to navigate the maze, i.e., $\mathbf{a}_t \in \{0, 1, 2\}$.
3. **Demonstrations, $\mathcal{D}$:** At the beginning of training, the agent is given state-action pairs of other agents collecting either red or green coins.

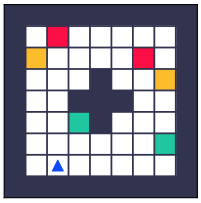


Figure 8. CoinGrid

A.4. Fruitbot

We build on the `Fruitbot` environment from OpenAI's ProcGen benchmark (Cobbe et al., 2020). The task is specified by:

1. **State space, $\mathcal{S}$:** We use the original high-dimensional 64×64 RGB observations, i.e., $\mathbf{s}_t \in [0, 1]^{64\times 64\times 3}$.
2. **Action space, $\mathcal{A}$:** We use the original 15 discrete actions, i.e., $\mathbf{a}_t \in \{0, \ldots, 14\}$.
3. **Demonstrations, $\mathcal{D}$:** At each time-step, the agent observes online the states and actions of 3 trained agents playing the game in parallel: One agent collects both fruits and other objects, one collects other objects and avoids fruits and the last one randomly selects actions.

Figure 9. Fruitbot

B. Implementation Details

For our experiments we used Python (Van Rossum & Drake Jr, 1995). We used JAX (Bradbury et al., 2018; Babuschkin et al., 2020) as the core computational library, Haiku (Hennigan et al., 2020) and Acme (Hoffman et al., 2020) for implementing ΨΦ-learning and the baselines, see Section 4. We also used Matplotlib (Hunter, 2007) for the visualisations and Weightd & Biases (Biewald, 2020) for managing the experiments.

B.1. Computation Graph

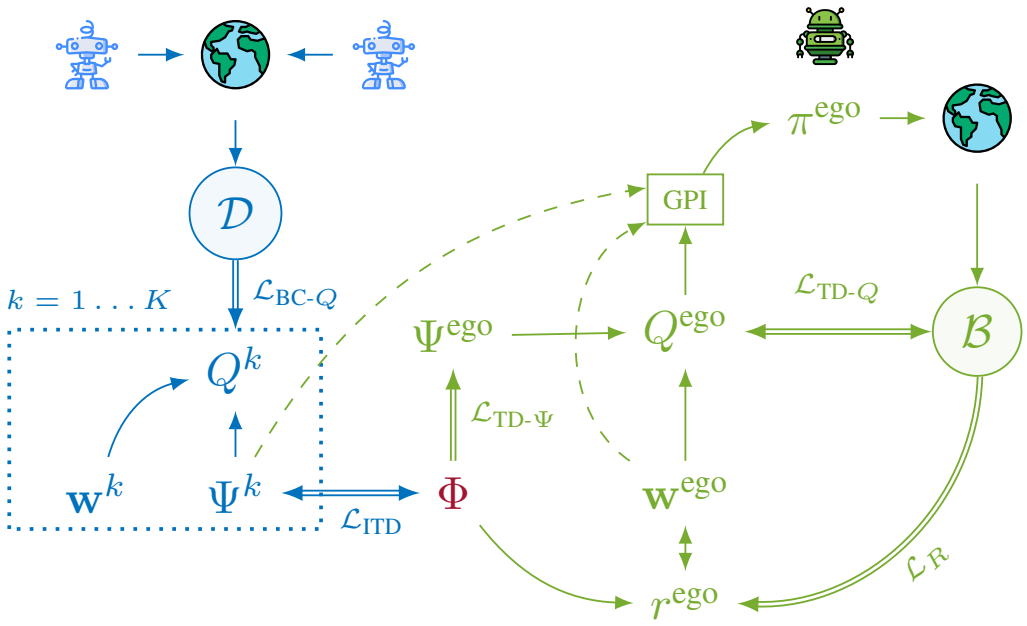


Figure 10. **Computational graph of the ΨΦ-learning algorithm.** Demonstrations $\mathcal{D}$ contain data from other agents for *unknown* tasks. We employ *inverse temporal difference learning* (ITD, see Section 3.1) to recover other agents' successor features (SFs) and preferences. The ego-agent combines the estimated SFs of others along with its own preferences and successor features with generalised policy improvement (GPI, see Section 2.2), generating experience. Both the demonstrations and the ego-experience are used to learn the shared cumulants. Losses $\mathcal{L}_*$ are represented with double arrows and gradients flow according to the pointed direction(s).

B.2. Neural Network Architecture

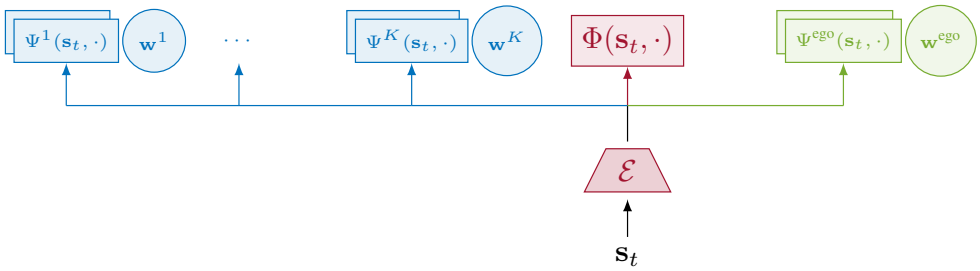


Figure 11. **Neural network architecture of the ΨΦ-learner.** The rectangular nodes are tensors parametrised by MLPs and the circles are learnable vectors. We share an observation network/torso, $\mathcal{E}$, across all the network heads. The network heads that related to the other agents are in blue and trained from demonstrations $\mathcal{D}$. The ego-agent's experience $\mathcal{B}$ is used for training the green heads. The shared cumulants and torso are trained with both $\mathcal{D}$ and $\mathcal{B}$. An ensemble of two successor features approximators is used for the ego- and other-agents for combatting model overestimation, see Section 3.

B.3. Hyperparameters

Table 3. **ΨΦ-learner's hyperparameters per environment**. The tuning was performed on a DQN (Mnih et al., 2013) baseline with population based training (Jaderberg et al., 2017) using Weights & Biases (Biewald, 2020) integration with Ray Tune (Liaw et al., 2018). We selected the best hyperparameters configuration out of 32 trials per environment and used this for our ΨΦ-learner.

	`Highway`	`CoinGrid`	`FruitBot`
Torso network, $\mathcal{E}$	MLP([512, 256])	IMPALA (Espeholt et al., 2018), shallow (no LSTM)	IMPALA (Espeholt et al., 2018), deep (no LSTM)
Cumulants approximator, Φ	MLP([128, 128])	MLP([256, 128])	MLP([256, 128])
Successor features approximator, Ψ	MLP([256, 128])	MLP([512, 256])	MLP([512, 256])
Ensemble size, Ψ	2	2	2
$\mathcal{L}_1$ coefficient	0.05	0.05	0.05
Number of dimensions in Φ	8	4	64
Minibatch size	512	64	32
n-step	4	8	128
Discount factor, γ	1.0	0.9	0.999
Target network update period	100	1000	2500
Optimiser	ADAM (Kingma & Ba, 2014), `lr=1e-3`	ADAM (Kingma & Ba, 2014), `lr=1e-4`	ADAM (Kingma & Ba, 2014), `lr=5e-5`

B.4. Compute Resources

All the experiments were run on Microsoft Azure `Standard_NC6s_v3` machines, i.e., with a 6-core vCPU, 112GB RAM and a single NVIDIA Tesla V100 GPU. The iteration cycle for (i) **Highway** experiments was 3 hours; (ii) **CoinGrid** experiments was 5.5 hours and (iii) **Fruitbot** experiments was 19 hours.

C. Proofs

First, we formalise the statement of Theorem 1.

Theorem 1 (Validity of the ITD Minimiser)**.** *The minimisers of $\mathcal{L}_{BC\text{-}Q}$ and $\mathcal{L}_{ITD}$ are potentially-shaped cumulants that explain the observed reward-free demonstrations.*

Proof. First, we prove the validity of the minimiser of the inverse temporal difference learning for the single-task setting. Next, we show that the result holds true in the vector (i.e., cumulants) case.

Single task. We assume that our demonstrations are generated by an expert, who samples actions from a Boltzmann policy, according to the optimal (for its task) action-value function $Q^{\pi_{\text{expert}}}$ and temperature $\nu > 0$, i.e., $\mathcal{D} = \{(\mathbf{s}, \mathbf{a})\} \sim \pi_{\text{expert}}$ s.t.

$$\pi_{\text{expert}}(\mathbf{a}|\mathbf{s}) \triangleq p(A = \mathbf{a}|\mathbf{s}) = \frac{\exp(\frac{1}{\nu} Q^{\pi_{\text{expert}}}(\mathbf{s}, \mathbf{a}))}{\sum_a \exp(\frac{1}{\nu} Q^{\pi_{\text{expert}}}(\mathbf{s}, a))}, \ \forall \mathbf{s} \in \mathcal{S}, \mathbf{a} \in \mathcal{A}\,. \tag{15}$$

The minimiser of the behavioural cloning loss $\mathcal{L}_{\text{BC-}Q}(\boldsymbol{\theta}_Q)$, i.e. Eqn. (8), for a single expert is s.t.

$$\boldsymbol{\theta}_Q^* \in \arg\min_{\boldsymbol{\theta}_Q} - \mathop{\mathbb{E}}_{(\mathbf{s},\mathbf{a})\sim\mathcal{D}} \log \frac{\exp(Q(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta}_Q))}{\sum_a \exp(Q(\mathbf{s}, a; \boldsymbol{\theta}_Q))} \tag{16}$$

$$\Rightarrow Q(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta}_Q^*) = \frac{1}{\nu} Q^{\pi_{\text{expert}}}(\mathbf{s}, \mathbf{a}) + F(\mathbf{s}), \ \forall \mathbf{s} \in \mathcal{S}, \mathbf{a} \in \mathcal{A}\,, \tag{17}$$

where $F : \mathcal{S} \to \mathbb{R}$ is a state-dependent (bounded potential) function. We arrive at Eqn. (17) by (1) testing $\frac{1}{\nu} Q^{\pi_{\text{expert}}}(\mathbf{s}, \mathbf{a})$ as a solution and noting that the "softmax" function is convex in the exponent (Boyd et al., 2004) and (2) using the translation invariance property of the assumed Boltzmann policy parametrisation, i.e., for any $f : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and $g : \mathcal{S} \to \mathbb{R}$

$$\frac{\exp(f(\mathbf{s}, \mathbf{a}) + g(\mathbf{s}))}{\sum_a \exp(f(\mathbf{s}, a) + g(\mathbf{s}))} = \frac{\exp(g(\mathbf{s})) \exp(f(\mathbf{s}, \mathbf{a}))}{\exp(g(\mathbf{s})) \sum_a \exp(f(\mathbf{s}, a))} = \frac{\exp(f(\mathbf{s}, \mathbf{a}))}{\sum_a \exp(f(\mathbf{s}, a))}\,. \tag{18}$$

The minimiser of the inverse temporal difference learning loss $\mathcal{L}_{\text{ITD}}(\boldsymbol{\theta}_Q, \boldsymbol{\theta}_r)$, Eqn. (9), for a single expert is s.t.

$$\boldsymbol{\theta}_Q^*, \boldsymbol{\theta}_r^* \in \arg\min_{\boldsymbol{\theta}_Q, \boldsymbol{\theta}_r} \mathop{\mathbb{E}}_{(\mathbf{s},\mathbf{a},\mathbf{s}',\mathbf{a}')\sim\mathcal{D}} \|Q(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta}_Q) - r(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta}_r) - \gamma Q(\mathbf{s}', \mathbf{a}'; \boldsymbol{\theta}_Q)\| \tag{19}$$

where $\boldsymbol{\theta}_Q^*$ is minimising $\mathcal{L}_{\text{BC-}Q}(\boldsymbol{\theta}_Q)$ simultaneously, as in Eqn. (17). Therefore, it holds that $\mathcal{L}_{\text{ITD}}(\boldsymbol{\theta}_Q^*, \boldsymbol{\theta}_r^*) = 0$

$$r(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta}_r^*) = Q(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta}_Q^*) - \gamma Q(\mathbf{s}', \mathbf{a}'; \boldsymbol{\theta}_Q^*) \tag{20}$$

$$\overset{(17)}{=} \frac{1}{\nu} Q^{\pi_{\text{expert}}}(\mathbf{s}, \mathbf{a}) + F(\mathbf{s}) - \gamma \frac{1}{\nu} Q^{\pi_{\text{expert}}}(\mathbf{s}', \mathbf{a}') - \gamma F(\mathbf{s}') \tag{21}$$

$$= \frac{1}{\nu} \left[\underbrace{Q^{\pi_{\text{expert}}}(\mathbf{s}, \mathbf{a}) - \gamma Q^{\pi_{\text{expert}}}(\mathbf{s}', \mathbf{a}')}_{r^{\text{expert}}(\mathbf{s},\mathbf{a})} \right] + F(\mathbf{s}) - \gamma F(\mathbf{s}') \tag{22}$$

$$= \frac{1}{\nu} r^{\text{expert}}(\mathbf{s}, \mathbf{a}) + \underbrace{F(\mathbf{s}) - \gamma F(\mathbf{s}')}_{\substack{\text{potential-based reward} \\ \text{shaping function}}}, \tag{23}$$

where r^{expert} is the (unobserved) expert's reward function. We have shown that the minimiser of $\mathcal{L}_{\text{BC-}Q}$ and $\mathcal{L}_{\text{ITD}}$ leads to a reward function $r(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta}_r^*)$ which is a potential-based shaped and scaled reward function of the expert reward function and hence the optimal policy for $r(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta}_r^*)$ is also optimal for $r^{\text{expert}}(\mathbf{s}, \mathbf{a})$ for all $\mathbf{s}, \mathbf{a}$ (Ng et al., 1999).

Multiple tasks. The minimiser of the behavioural cloning loss $\mathcal{L}_{\text{BC-}Q}(\boldsymbol{\theta}_{\Psi^k}, \mathbf{w}^k)$, i.e., Eqn. (8), for the k-th expert is s.t.

$$\boldsymbol{\theta}_{\Psi^k}^*, \mathbf{w}^{k*} \in \arg\min_{\boldsymbol{\theta}_{\Psi^k}, \mathbf{w}^k} - \mathop{\mathbb{E}}_{(\mathbf{s},\mathbf{a})\sim\mathcal{D}^k} \log \frac{\exp(\Psi(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta}_{\Psi^k})^\top \mathbf{w}^k)}{\sum_a \exp(\Psi(\mathbf{s}, a; \boldsymbol{\theta}_{\Psi^k})^\top \mathbf{w}^k)} \tag{24}$$

$$\Rightarrow \Psi(\mathbf{s}, \mathbf{a}; \boldsymbol{\theta}_{\Psi^k})^\top \mathbf{w}^k = \frac{1}{\nu} Q^{\pi_{k\text{-expert}}}(\mathbf{s}, \mathbf{a}) + F^k(\mathbf{s}), \ \forall \mathbf{s} \in \mathcal{S}, \mathbf{a} \in \mathcal{A}\,, \tag{25}$$

where $Q^{\pi_{k\text{-expert}}}$ is the k-agent's action-value function and $H^k : \mathcal{S} \to \mathbb{R}$ a state-dependent (bounded potential) function. Next, the minimiser of the inverse temporal difference learning loss $\mathcal{L}_{\text{ITD}}(\boldsymbol{\theta}_{\Psi^k}, \boldsymbol{\theta}_\Phi)$, Eqn. (9), for the k-th expert is s.t.

$$\boldsymbol{\theta}^*_{\Psi^k}, \boldsymbol{\theta}^*_\Phi \in \underset{\boldsymbol{\theta}_{\Psi^k}, \boldsymbol{\theta}_\Phi}{\arg\min} \underset{(\mathbf{s},\mathbf{a},\mathbf{s}',\mathbf{a}')\sim\mathcal{D}^k}{\mathbb{E}} \|\Psi(\mathbf{s},\mathbf{a};\boldsymbol{\theta}_{\Psi^k}) - \Phi(\mathbf{s},\mathbf{a};\boldsymbol{\theta}_\Phi) - \gamma\Psi(\mathbf{s}',\mathbf{a}';\boldsymbol{\theta}_{\Psi^k})\| \tag{26}$$

where $\boldsymbol{\theta}^*_{\Psi^k}$ is minimising $\mathcal{L}_{\text{BC-}Q}(\boldsymbol{\theta}_{\Psi^k}, \mathbf{w}^k)$ simultaneously, as in Eqn. (25). Therefore, it holds that for $\mathcal{L}_{\text{ITD}}(\boldsymbol{\theta}^*_{\Psi^k}, \boldsymbol{\theta}^*_\Phi) = 0$

$$\Phi(\mathbf{s},\mathbf{a};\boldsymbol{\theta}^*_\Phi) = \Psi(\mathbf{s},\mathbf{a};\boldsymbol{\theta}^*_{\Psi^k}) - \gamma\Psi(\mathbf{s}',\mathbf{a}';\boldsymbol{\theta}^*_{\Psi^k}) \tag{27}$$

$$\Phi(\mathbf{s},\mathbf{a};\boldsymbol{\theta}^*_\Phi)^\top \mathbf{w}^{k*} = \Psi(\mathbf{s},\mathbf{a};\boldsymbol{\theta}^*_{\Psi^k})^\top \mathbf{w}^{k*} - \gamma\Psi(\mathbf{s}',\mathbf{a}';\boldsymbol{\theta}^*_{\Psi^k})^\top \mathbf{w}^{k*} \tag{28}$$

$$\overset{(25)}{=} \frac{1}{\nu} Q^{\pi_{k\text{-expert}}}(\mathbf{s},\mathbf{a}) + F^k(\mathbf{s}) - \gamma\frac{1}{\nu} Q^{\pi_{k\text{-expert}}}(\mathbf{s}',\mathbf{a}') - \gamma F^k(\mathbf{s}') \tag{29}$$

$$= \frac{1}{\nu}\left[\underbrace{Q^{\pi_{k\text{-expert}}}(\mathbf{s},\mathbf{a}) - \gamma Q^{\pi_{k\text{-expert}}}(\mathbf{s}',\mathbf{a}')}_{r^{k\text{-expert}}(\mathbf{s},\mathbf{a})}\right] + F^k(\mathbf{s}) - \gamma F^k(\mathbf{s}') \tag{30}$$

$$= \frac{1}{\nu} r^{k\text{-expert}}(\mathbf{s},\mathbf{a}) + \underbrace{F^k(\mathbf{s}) - \gamma F^k(\mathbf{s}')}_{\substack{\text{potential-based reward} \\ \text{shaping function}}}, \tag{31}$$

We have shown that the minimiser of $\mathcal{L}_{\text{BC-}Q}$ and $\mathcal{L}_{\text{ITD}}$ leads to agent-agnostic cumulants $\Phi(\mathbf{s},\mathbf{a};\boldsymbol{\theta}^*_\Phi)$ and agent-specific preference vector $\mathbf{w}^{k*}$, which when dot-producted, form a potential-based shaped and scaled reward function of the k-th expert reward function and hence the optimal policy for $\Phi(\mathbf{s},\mathbf{a};\boldsymbol{\theta}^*_\Phi)^\top \mathbf{w}^{k*}$ is also optimal for $r^{k\text{-expert}}(\mathbf{s},\mathbf{a})$ for all $\mathbf{s},\mathbf{a}$ (Ng et al., 1999). The result holds for all $k \in \{1 \dots K\}$ since no assumptions were made for the proof about k. □

Next, we formalise the statement of Theorem 2. When not specified the norm $\|\cdot\|$ refers to the 2-norm. Given a function $F : \mathcal{X} \to \mathbb{R}^d$ for some finite set $\mathcal{X}$, we will write $F(x)$ to denote the value of the function on input x and F to denote the matrix representation of this function in $\mathbb{R}^{|\mathcal{X}|\times d}$.

Theorem 2 (Generalisation Bound of ΨΦ-Learning)**.** *Let $\mathcal{C} = (\mathcal{S}, \mathcal{A}, P, \gamma)$ be a CMP with a finite state space. Let $\phi : \mathcal{S} \to \mathbb{R}^d$, and let $\Phi = \phi(\mathcal{S}) \in \mathbb{R}^{|\mathcal{S}|\times d}$. Let $(r_i)_{i=1}^k$ denote a set of reward functions on $\mathcal{C}$, $\tilde{\Psi}^i$ be a collection of successor features approximations for policies $(\pi^i)_{i=1}^k$ (π_i optimal for r_i) with true successor feature values Ψ^i, and w_i the best least-squares linear approximator of r_i given Φ, with errors*

$$\|\Phi w_i - r_i\|_\infty < \delta_r \text{ and } \|\tilde{\Psi}^i - \Psi^i\| < \delta_\Psi \quad \forall i.$$

Let w' be a new preference vector for a reward function r', with maximal error δ_r as well. Let $\tilde{Q}^i = \tilde{\Psi}^i w'$. Let π^* be the optimal policy for the ego task w' and let π be the GPI policy obtained from $\{\tilde{Q}^{\pi_i}\}$, with δ_r, δ_Ψ the reward and successor feature approximation errors. Then for all s, a

$$Q^*(s,a) - Q^\pi(s,a) \leq \frac{2}{1-\gamma}\left[(\phi_{\max}\|w_j - w'\| + 2\delta_r) + \|w'\|\delta_\Psi + \frac{1}{(1-\gamma)}\delta_r\right] \tag{32}$$

Barreto et al. (2017) construct their bound on the sub-optimality of the GPI policy as a function of the error of the value approximations $\widetilde{Q}^i$. Because we bound the reward approximation error, rather than the value approximation error, we require an additional step to obtain a bound on the errors of the value funciton approximations. To prove Theorem 1, we must therefore first use the following lemma to bound the effect of the *reward approximation error* on the value approximation error. While this result is straightforward, we include a short proof for completeness.

Lemma 1. *Fix some policy π. Let r be reward vector and let w be the least-squares solution to $\min \|\Phi w - r\|$. Let Ψ^π be the true successor features for Φ under policy π, and let Q^π be the value. Let $\delta_r = R(\mathcal{S}) - \Phi w$, $\delta_{max} = \|\delta_r\|_\infty$. Then letting $\tilde{Q} = \Psi w$, we have*

$$\|Q^\pi - \tilde{Q}\|_\infty \leq \frac{1}{1-\gamma}\delta_r \tag{33}$$

Proof.

$$\|Q^\pi - \tilde{Q}\|_\infty \leq \sum \gamma^t \|P^{\pi t}(\Phi w - r)\|_\infty \tag{34}$$

$$\leq \sum \gamma^t \|P^{\pi t}\delta_r\|_\infty = \sum_t \gamma^t \max_{s'} |\sum_{s\in\mathcal{S}} (P^\pi)^t(s',s)\delta_r(s)| \tag{35}$$

Since P^π is a stochastic matrix, so are all of its powers, and so the rows of $(P^\pi)^t$ sum to 1.

$$\leq \sum_t \gamma^t \max_{s'} |\sum P^{\pi\, t}(s, s')\delta_{\max}| = \sum \gamma^t \delta_{\max} \tag{36}$$

$$= \frac{1}{1-\gamma}\delta_{\max} \tag{37}$$

□

We now prove the main result.

Proof. We follow the proof of Barreto et al. (2017, Theorem 2), with additional error terms to account for the reward and successor feature approximation errors.

$$\begin{aligned}
Q^*(s,a) - Q^\pi(s,a) &\leq Q^*(s,a) - Q^{\pi_j}(s,a) + \frac{2}{1-\gamma}\epsilon && \text{(Barreto et al., 2017, Theorem 1)} \\
&\leq \frac{2}{1-\gamma}\|r_j - r'\|_\infty + \frac{2}{1-\gamma}\epsilon && \text{(Barreto et al., 2017, Lemma 1)} \\
&\leq \frac{2}{1-\gamma}\|\phi w_j + \delta_j - \phi w' - \delta'\|_\infty + \frac{2}{1-\gamma}\epsilon \\
&\leq \frac{2}{1-\gamma}(\phi_{\max}\|w_j - w'\| + \delta_r + \delta_r) + \frac{2}{1-\gamma}\epsilon \\
&\leq \frac{2}{1-\gamma}(\phi_{\max}\|w_j - w'\| + 2\delta_r) + \frac{2}{1-\gamma}\|\tilde{\Psi}^j w' - \Psi_j w' + \Psi_j w' - Q_j\| \\
&\leq \frac{2}{1-\gamma}(\phi_{\max}\|w_j - w'\| + 2\delta_r) + \frac{2}{1-\gamma}\|\tilde{\Psi}^j w' - \Psi_j w'\| + \|\Psi_j w' - Q_j\| \\
&\leq \frac{2}{1-\gamma}(\phi_{\max}\|w_j - w'\| + 2\delta_r) + \frac{2}{1-\gamma}\|w'\|\delta_\Psi + \frac{2}{1-\gamma}\|\Psi_j w' - Q_j\| \\
&\leq \frac{2}{1-\gamma}(\phi_{\max}\|w_j - w'\| + 2\delta_r) + \frac{2}{1-\gamma}\|w'\|\delta_\Psi + \frac{2}{1-\gamma}(\frac{1}{1-\gamma}\delta_r) && \text{(Lemma 1)} \\
&= \frac{2}{1-\gamma}\left[(\phi_{\max}\|w_j - w'\| + 2\delta_r) + \|w'\|\delta_\Psi + \frac{1}{(1-\gamma)}\delta_r\right]
\end{aligned}$$

□

D. Algorithms

Algorithm 1: Inverse Temporal Difference Learning

Input :

$\mathcal{D} = \{(\mathbf{s}_1, \mathbf{a}_1, \dots, \mathbf{a}_T; k)_{k=1}^K\}$ No-reward demonstrations

$\lambda_{\mathbf{w}}$ $\mathcal{L}_1$ loss coefficient

Output :

$\boldsymbol{\theta}_\Phi$ Parameters of cumulants network

$\{\boldsymbol{\theta}_{\Psi^k}\}_{k=1}^K$ Parameters of successor features approximators

$\{\mathbf{w}^k\}_{k=1}^K$ Preferences vectors for the K agents

// initialisations

Initialise parameters $\boldsymbol{\theta}_\Phi, \{\boldsymbol{\theta}_{\Psi^k}, \mathbf{w}^k\}_{k=1}^K$

while *budget* **do**

  Sample trajectories $\{\tau_i = (s_1^{(i)}, a_1^{(i)}, \dots, s_T^{(i)}, a_T^{(i)}; k^{(i)})\}_{i=1}^N \sim \mathcal{D}$

  Calculate behavioural cloning loss $\mathcal{L}_{\text{BC-}Q}(\boldsymbol{\theta}_{\Psi^k}, \mathbf{w}^k)$ on samples $\{\tau_i\}_{i=1}^N$ ▷ see Eqn. (8)

  $\boldsymbol{\theta}_{\Psi^k} \overset{\alpha}{\leftarrow} \nabla_{\boldsymbol{\theta}_{\Psi^k}} \mathcal{L}_{\text{BC-}Q}(\boldsymbol{\theta}_{\Psi^k}, \mathbf{w}^k)$ ▷ update Ψs

  $\mathbf{w}^k \overset{\alpha}{\leftarrow} \nabla_{\mathbf{w}^k} \left(\mathcal{L}_{\text{BC-}Q}(\boldsymbol{\theta}_{\Psi^k}, \mathbf{w}^k) + \lambda_{\mathbf{w}} \|\mathbf{w}^k\|_1\right)$ ▷ update ws

  Calculate inverse temporal difference loss $\mathcal{L}_{\text{ITD}}(\boldsymbol{\theta}_\Phi, \boldsymbol{\theta}_{\Psi^k})$ on samples $\{\tau_i\}_{i=1}^N$ ▷ see Eqn. (9)

  $\boldsymbol{\theta}_\Phi \overset{\alpha}{\leftarrow} \nabla_{\boldsymbol{\theta}_\Phi} \mathcal{L}_{\text{ITD}}(\boldsymbol{\theta}_{\Psi_k}, \boldsymbol{\theta}_\Phi)$ ▷ update Φ

  $\boldsymbol{\theta}_{\Psi_k} \overset{\alpha}{\leftarrow} \nabla_{\boldsymbol{\theta}_{\Psi_k}} \mathcal{L}_{\text{ITD}}(\boldsymbol{\theta}_{\Psi_k}, \boldsymbol{\theta}_\Phi)$ ▷ update Ψs

Algorithm 2: $\Psi\Phi$-Learning

Input :

$\mathcal{D} = \{(\mathbf{s}_1, \mathbf{a}_1, \dots, \mathbf{a}_T; k)_{k=1}^K\}$ No-reward demonstrations

$\lambda_{\mathbf{w}}$ $\mathcal{L}_1$ loss coefficient

Output :

$\boldsymbol{\theta}_{\Psi^{\text{ego}}}$ Ego successor features approximator

$\boldsymbol{\theta}_{\Phi}$ Parameters of cumulants network

$\{\boldsymbol{\theta}_{\Psi^k}\}_{k=1}^K$ Parameters of successor features approximators

$\{\mathbf{w}^k\}_{k=1}^K$ Preferences vectors for the K agents

`// initialisations`

Empty replay buffer for ego-experience $\mathcal{B} = \{\}$

Initialise parameters $\boldsymbol{\theta}_{\Psi^{\text{ego}}}, \mathbf{w}^{\text{ego}}, \boldsymbol{\theta}_{\Phi}, \{\boldsymbol{\theta}_{\Psi^k}, \mathbf{w}^k\}_{k=1}^K$

while *budget* **do**

`// agent-environment interaction`

Reset episode, $\mathbf{s} \leftarrow$ `env.reset()`, $t \leftarrow 0$

while *not done* **do**

$\mathbf{w}^{\text{ego}} \leftarrow \arg\min_w \mathcal{L}_{\text{R}}(\boldsymbol{\theta}_{\Phi}, w; \mathcal{B})$ ▷ ego-task inference, see Eqn. (10)

$\mathbf{a} \leftarrow \pi_{\text{GPI}}^{\text{ego}}\left(\mathbf{s}; \Psi^{\text{ego}}, \mathbf{w}^{\text{ego}}, \{\boldsymbol{\theta}_{\Psi^k}\}_{k=1}^K\right)$ ▷ GPI, see Eqn. (13)

Step in the environment, $\mathbf{s}', r^{\text{ego}},$ `done` $\leftarrow$ `env.step(`$\mathbf{a}$`)`

Append transition in the replay buffer, $\mathcal{B} \leftarrow \mathcal{B} \cup (\mathbf{s}, \mathbf{a}, r^{\text{ego}}, \mathbf{s}')$

$\mathbf{s}' \leftarrow \mathbf{s}, t \leftarrow t + 1$

(online demonstrations) Append demonstrations in $\mathcal{D}$ ▷ optional

`// parameter updates/learning`

$\boldsymbol{\theta}_{\Phi}, \{\boldsymbol{\theta}_{\Psi^k}, \mathbf{w}^k\}_{k=1}^K \leftarrow \texttt{ITD}\left(\mathcal{D}, \lambda_{\mathbf{w}}, \boldsymbol{\theta}_{\Phi}, \{\boldsymbol{\theta}_{\Psi^k}, \mathbf{w}^k\}_{k=1}^K\right)$ ▷ see Algorithm. (1)

Sample transitions $\{(\mathbf{s}^{(i)}, \mathbf{a}^{(i)}, r^{\text{ego},(i)}, \mathbf{s}'^{(i)})\} \sim \mathcal{B}$

Calculate the reward loss $\mathcal{L}_{\text{R}}(\boldsymbol{\theta}_{\Phi}, \mathbf{w}^{\text{ego}})$ ▷ see Eqn. (10)

$\boldsymbol{\theta}_{\Phi} \overset{\alpha}{\leftarrow} \nabla_{\boldsymbol{\theta}_{\Phi}} \mathcal{L}_{\text{R}}(\boldsymbol{\theta}_{\Phi}, \mathbf{w}^{\text{ego}})$ ▷ update Φ

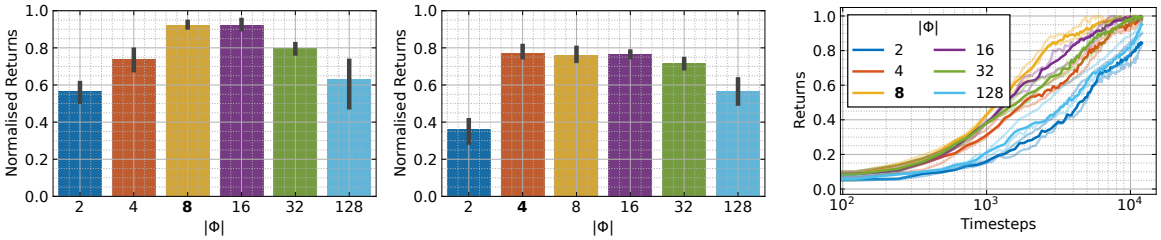
Calculate TD losses $\mathcal{L}_Q(\boldsymbol{\theta}_{\Psi^{\text{ego}}})$ and $\mathcal{L}_{\text{TD-}\Psi}(\boldsymbol{\theta}_{\Psi^{\text{ego}}})$ ▷ see Eqn. (11,12)

$\boldsymbol{\theta}_{\Psi^{\text{ego}}} \overset{\alpha}{\leftarrow} \nabla_{\boldsymbol{\theta}_{\Psi^{\text{ego}}}} \left(\mathcal{L}_Q(\boldsymbol{\theta}_{\Psi^{\text{ego}}}) + \frac{1}{|\Psi|}\mathcal{L}_{\text{TD-}\Psi}(\boldsymbol{\theta}_{\Psi^{\text{ego}}})\right)$ ▷ update Ψ^{ego}

E. Visualisations

(a) CoinGrid (b) ϕ_1 (c) ϕ_2 (d) ϕ_3 (e) ϕ_4

Figure 12. Qualitative evaluation of the learned cumulants in the CoinGrid task. Cumulants ϕ_1, ϕ_2, and ϕ_3 seem to capture the red, green, and yellow blocks, respectively. The yellow blocks are captured by both and ϕ_4. Therefore, linear combinations of the learned cumulants can represent arbitrary rewards in the environment, which involve stepping on the coloured blocks.



(a) ITD for Roundabout (b) ITD for CoinGrid (c) ΨΦ-learning for Highway Multi-Task

Figure 13. Sensitivity of our ITD (see Section 3.1) and ΨΦ-learning (see Section 3.2) algorithms to the dimensionality of the learned cumulants. We consistently observe across all three experiments (a)-(c) that for a small number of Φ dimensions the cumulants are not expressive enough to capture the axis of variation of the different agents' reward functions (including the ego-agent in (c)). We also note that the performance of both ITD and ΨΦ-learning is relative robust for a medium and large number of Φ dimensions. We attribute this to the used sparsity prior, i.e., $\mathcal{L}_1$ loss, to the preferences $\mathbf{w}$. In our experiments we selected the smallest number of Φ dimensions that demonstrated good performance to keep the number of model parameters as small as possible (in bold in the figures and reported in Table 3).