A PROOF FOR THEOREM 4.1

Theorem 4.1 shows that the LIL problem (as a bilevel optimization problem), *i.e*,

$$\max_{\pi} \underbrace{\mathbb{E}_{(s,a)\in\mathcal{D}_E}[\log \pi(a|s)]}_{①} + \underbrace{\mathbb{E}_{s\in\mathcal{D}_E}[\log P_\pi(s)]}_{②}, \quad \text{s.t.} \quad P_\pi = \arg\max_{P} \mathbb{E}_{s\in\mathcal{D}_\pi}[\log P(s)]. \tag{2}$$

can be relaxed to a two-stage optimization problem, *i.e*,

$$\max_{\pi} \mathbb{E}_{(s,a)\in\mathcal{D}_E}[\log \pi(a|s)] + \mathbb{E}_{s\in\mathcal{D}_\pi}[\log P_E(s)], \quad \text{s.t.} \quad P_E = \arg\max_{P} \mathbb{E}_{s\in\mathcal{D}_E}[\log P(s)]. \tag{3}$$

which estimates the expert state distribution P_E (*in Stage #1*), and trains the learner policy π based on P_E (*in Stage #2*).

Proof. Given sufficient expert data $\mathcal{D}_E$, the expert state distribution $P_E(s)$ can be estimated by maximizing the likelihood of $\mathcal{D}_E$ as below, which is naturally an upper-bound of component ② in eq.(2).

$$P_E = \arg\max_{P} \mathbb{E}_{s\in\mathcal{D}_E}[\log P(s)], \text{with } \mathbb{E}_{s\in\mathcal{D}_E}[\log P_E(s)] \geq \max_{\pi} \mathbb{E}_{s\in\mathcal{D}_E}[\log P_\pi(s)].$$

This indicates that ideally the LIL problem in eq.(2) aims to learn an optimal policy π^* with a state distribution P_{π^*} matching the expert distribution P_E, say, $P_{\pi^*} = P_E$. Then, by importance sampling (Neal, 2001), we have,

$$\mathbb{E}_{s\sim P_E}\Big[\log P_\pi(s)\Big] = \mathbb{E}_{s\sim P_\pi}\Big[\frac{P_E(s)}{P_\pi(s)} \log P_\pi(s)\Big].$$

With sufficient expert data $\mathcal{D}_E$, ② in eq.(2) can be expressed with $\mathbb{E}_{s\in\mathcal{D}_\pi}[\frac{P_E(s)}{P_\pi(s)} \log P_\pi(s)]$. For ease of notation, we define ②$=g(\pi, P_\pi) \triangleq \mathbb{E}_{s\in\mathcal{D}_\pi}[\frac{P_E(s)}{P_\pi(s)} \log P_\pi(s)]$ which is a function of policy π and its state distribution P_π. We denote the optimal policy as π^*. Since the optimal state distribution $P_{\pi^*} = P_E$, we have the following inequality, *i.e*,

$$g(\pi^*, P_{\pi^*}) = g(\pi^*, P_E) = \mathbb{E}_{s\in\mathcal{D}_{\pi^*}}[\log P_{\pi^*}(s)] \geq \max_{\pi} g(\pi, P_E) = \max_{\pi} \mathbb{E}_{s\in\mathcal{D}_\pi}[\log P_E(s)],$$

where equality holds when $\pi = \pi^*$, at least. Therefore, eq.(2) can be rewritten as

$$\begin{aligned}
&\max_{\pi} \Big(\mathbb{E}_{(s,a)\in\mathcal{D}_E}[\log \pi(a|s)] + \mathbb{E}_{s\in\mathcal{D}_E}[\log P_\pi(s)]\Big) \\
\overset{\pi=\pi^*}{=}\ &\mathbb{E}_{(s,a)\in\mathcal{D}_E}[\log \pi^*(a|s)] + \mathbb{E}_{s\in\mathcal{D}_E}[\log P_{\pi^*}(s)] \\
\geq\ &\max_{\pi} \mathbb{E}_{(s,a)\in\mathcal{D}_E}[\log \pi(a|s)] + \max_{\pi} \mathbb{E}_{s\in\mathcal{D}_\pi}[\log P_E(s)] \\
\geq\ &\max_{\pi} \Big(\mathbb{E}_{(s,a)\in\mathcal{D}_E}[\log \pi(a|s)] + \mathbb{E}_{s\in\mathcal{D}_\pi}[\log P_E(s)]\Big),
\end{aligned}$$

as the log likelihood terms are all non-positive. This completes the proof. □

B EXPERIMENT SETUPS

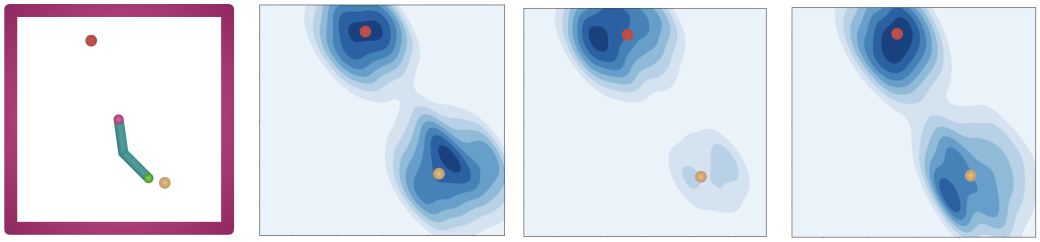
Resource usage and running time. All experiments are run on GeForce RTX2080. The training of $P_{E\omega^*}$ takes roughly 6 hours for CartPole and Reacher, and 12 hours for Hopper and Walker for 200 iterations.

Evaluation setup. We use the same amount of environment interactions and expert demonstrations in GAIL, GPRIL, DRIL, and SLIL, where information of each task is shown in Tab. 2. For all approaches, we evaluate their learner policy performances in every iteration. The experiments with multiple mode task settings all use 18 expert trajectories. We do not subsample expert trajectories for any of the experiment tasks. The evaluation score is achieved via evaluating the mean and std of 50 trajectories generated with the learner policy.

Model details. All baselines and SLIL shares the same policy network structure in all experiments – tanh nonlinearlities sandwiched with two hidden layers of 100 units. Consistent with GAIL Ho &

Table 2: Parameters for baselines SLIL.

Task	Training iterations	Number of (s, a) per iteration	Expert performance	Random policy performance
CartPole-v0	200	200	200±0	17± 4
Hopper-v2	1000	1000	3624±19	8± 6
Reacher-v2	200	1000	-4.5±1.7	-93.7 ±4.8
Walker-v2	1000	1000	7002±33	-2±3



(a) Reacher w/ 2 targets. (b) True target visitation. (c) GAIL target visitation. (d) SLIL (Ours) target visitation.

Figure 15: Results obtained by SLIL (Ours) and baselines on mode coverage. (a): A Reacher task, with two targets in different colors. (b)-(d) show the mode coverage (*i.e*, state distribution) with expert policy (b), GAIL policy (c), and our SLIL policy (d). All the distributions are visualized using KDE (Sheather & Jones, 1991).

Ermon (2016), value functions in SLIL has the the same neural network architecture as the policy networks and employs generalized advantage estimation Schulman et al. (2015b) with $\lambda = 0.99$ and = $\gamma = 0.95$ to decrease the gradient variance. The DCNF model features a hypernetwork-Ha et al. (2016) with hidden dimension of 32 and width of 64. To train it, we use Dopri5 ODE integrator.

Hyperparameter details. For training P_{E,ω^*}, we schedule a linear noise level decay with $\sigma_0 = 1$ and $\sigma_{200} = 0$. The learning rate is $1e-4$ for all tasks. The training of π also has a learning rate at $1e-4$ with gradient clip 0.1.

C More Experiment Results

We further show the learning curves of DCNF and SoftFlow to better understand the efficacy of the denoising mechanism in Fig. 14. The figure unfolds that in each training epoch the training loss of DCNF lies below SoftFlow. This likely indicates that by decreasing the noise level in each training iteration, the denoising mechanism expose different levels of regularization for optimization. Such types of regularization enable the DCNF model to find a better local minimum than SoftFlow. We therefore hypothesize that the denoising mechanism is able to make the loss landscape more smooth than SoftFlow, and reaches a better minimum entailing better expert state distribution learning. We omit the learning curves in other tasks as similar observations are made.

Fig. 15 shows the mode coverage results of SLIL (ours) and baselines over the Reacher task with two target modes. It shows two mode targets from expert demonstrations Fig. 15(b), where GAIL only visited the red one and SLIL visited both targets. Tab. 3 shows detailed results of SLIL and baselines in tasks with single expert demonstration mode. Fig. 16 shows the learning curves of the test tasks with a single mode when demonstration number is 4. In all tasks, SLIL has more stable training curves (with less mean return perturbation) with higher convergence speed.

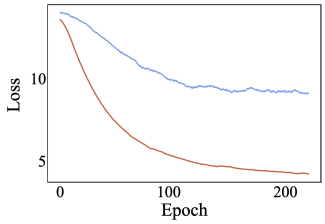
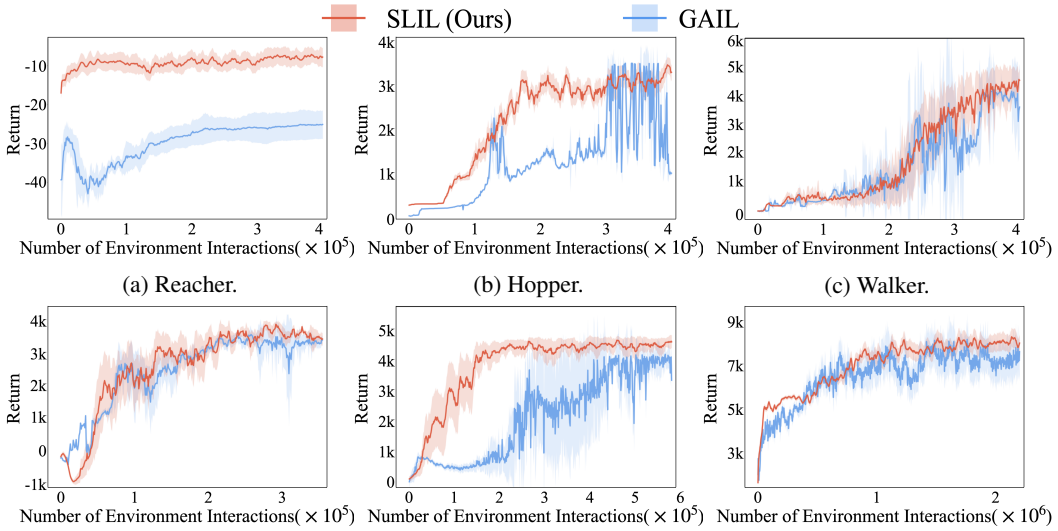


Figure 14: Learning curves of DCNF and SoftFlow in the Hopper task with 18 demonstration trajectories.

Tab. 4 shows the EMD (Ling & Okada, 2007), KL, and reverse KL (RKL) results between expert and learner final state distributions in the Reacher2, Reacher4 and HalfCheetah2 tasks. It reflects that SLIL is better at recovering expert modes than baseline methods.



(a) Reacher. (b) Hopper. (c) Walker.

(d) HalfCheetah. (e) Ant. (f) Humanoid.

Figure 16: Learning curve comparision between GAIL and SLIL (Ours). All tasks are shown 4 demonstrations. The y-axis is the obtained return (*i.e*, total reward).

Table 3: Learned policy performance.

Task	Datasize	BC	GAIL	GPRIL	DRIL	SLIL (Ours)
CartPole	1	59±27	200±0	53±16	200±0	**200±0**
	4	81±31	200±0	187±8	200±0	**200±0**
	7	137 ±27	200 ±0	200 ±0	200±0	**200±0**
	10	167± 30	200 ±0	200 ±0	200±0	**200±0**
Reacher	4	-10.27±2.14	-26.90±7.48	-12.55±3.54	**-9.13±.83**	-9.44 ±3.16
	11	-9.49±3.66	-12.77±8.90	-10.45±5.21	-7.11±2.23	**-6.34± 3.26**
	18	-8.89±3.83	-7.34±2.63	-9.96±5.01	-6.93±2.37	**-6.33±2.52**
	25	-9.63±3.84	-6.64±2.47	-11.87±4.71	-6.90±2.65	**-5.78± 2.50**
Hopper	4	2352± 894	**3394±37**	22± 1	898±132	3381± 183
	11	2589± 635	**3599± 4**	407± 202	3150±184	3500± 31
	18	3331± 66	3631± 3	1339± 1390	3611±3	**3686± 8**
	25	3589± 56	3476± 5	1406± 844	3580±8	**3595± 8**
Walker2d	4	1233± 969	4070 ±1010	557 ±357	555±148	**4230± 1108**
	11	3456± 863	5108± 410	1042± 75	4567±1231	**6537± 560**
	18	4477± 1329	6671± 39	1464± 637	**6886±202**	6873± 55
	25	5294± 1860	6815± 20	2254± 1006	6693±130	**6923± 58**
Ant	4	4204±289	4218±240	2722±36	3837±259	**4613±161**
	11	**4577±145**	4105±223	2510±27	4515±239	4540±169
	18	4736±75	4690±102	2755±183	4703±40	**4752±91**
	25	4682±89	4735±54	2656±85	4690±75	**4825±45**
HalfCheetah	4	2070±528	3254±133	558±148	359±266	**3687±416**
	11	3979±61	4015±344	2655±253	**4063±50**	4052±236
	18	3911±416	4393±212	2666±186	4185±30	**4531±65**
	25	4027±91	**4423±104**	3619±257	4227±26	4416±77
Humanoid	80	6145±1918	8268±1401	2048±1140	**8800±639**	8404±571
	160	6722±1126	**9994±1053**	6023±1006	9507±832	9771±835
	240	8834±998	**9430±906**	8091±878	9185±492	9294±385

Task	Approach	Measure		
		EMD	KL	RKL
Reacher2	BC	1.01	2.60	4.38
	GAIL	0.84	2.47	4.51
	DRIL	1.00	2.51	4.25
	SLIL	**0.47**	**2.39**	**3.25**
Reacher4	BC	0.81	4.23	6.33
	GAIL	0.41	4.49	6.22
	DRIL	0.55	5.16	6.16
	SLIL	**0.33**	**3.94**	**4.53**
HalfCheetah2	BC	1.81	6.74	12.78
	GAIL	1.75	4.52	18.13
	DRIL	1.83	6.90	12.76
	SLIL	**1.01**	**3.89**	**11.54**

Table 4: The EMD (Ling & Okada, 2007), KL and RKL between expert and learned policy state distribution.