Appendix: Variational Empowerment as Representation Learning for Goal-Based Reinforcement Learning

A Background: Mutual Information Maximization

We provide a detailed discussion about mutual information objectives as promised in Section 3.1.

State-predictive MI: Given a generative model of the form $p^\pi(z, s, s') = p(z)\rho^\pi(s|z)p^\pi(s'|s, z)$ where $p^\pi(s'|s, z) = \int \pi(a|s, z)p(s'|s, a)da$, we define the state-predictive MI as,

$$\begin{aligned}\mathcal{I}(s'; z \mid s) &= \mathcal{H}(s' \mid s) - \mathcal{H}(s' \mid z, s) && (8)\\ &= \mathbb{E}_{(z,s,s')\sim p^\pi(z,s,s')}\left[\log p^\pi(s' \mid s, z) - \log p^\pi(s' \mid s)\right] && (9)\end{aligned}$$

This is closer to the classic empowerment formulation as in (Klyubin et al., 2005; Jung et al., 2011). Variational bounds can be derived with respect to actions (Mohamed & Rezende, 2015; Gregor et al., 2017) or to future states (Sharma et al., 2020b). While this objective enables learning state-conditioned skills, we decide to focus on the other variant in this paper.

State-marginal MI: Similarly, given a generative model of the form $p^\pi(z, s) = p(z)\rho^\pi(s|z)$, the MI can be written as,

$$\begin{aligned}\mathcal{I}(s; z) &= \mathcal{H}(z) - \mathcal{H}(z|s) && (10)\\ &= \mathbb{E}_{z\sim p(z)}[-\log p(z)] + \mathbb{E}_{z,s\sim p^\pi(z,s)}[\log p(z \mid s)] && (11)\\ &= \mathbb{E}_{z\sim p(z),s\sim\pi(z)}[\log p(z \mid s) - \log p(z)] && (12)\\ &\geq \mathbb{E}_{z\sim p(z),s\sim\pi(z)}[\log q_\lambda(z \mid s) - \log p(z)], && (13)\end{aligned}$$

where Eq. 13 is a common variational bound for MI (Barber & Agakov, 2003) with a variational posterior $q_\lambda(z|s)$ approximating the intractable posterior $p^\pi(z|s)$. DIAYN (Eysenbach et al., 2019) optimizes for this state-marginal MI objective in a entropy-regularized RL setting, trained with the SAC algorithm (Haarnoja et al., 2018). We note that an alternative lower bound of $\mathcal{I}(s; z)$ (a "forward" form of MI, i.e., $\mathcal{H}(s) - \mathcal{H}(s|z)$) is also possible (Campos et al., 2020).

B Equivalence between GCRL and Gaussian VGCRL

Full Covariance Gaussian. The Gaussian discriminator (or the variational posterior) $q_\lambda(z|s)$ should take the following form:

$$\begin{aligned}q_\lambda(z|s) &= \mathcal{N}(z; \phi(s), \Sigma(s)) && (14)\\ &= \frac{1}{\sqrt{(2\pi)^{|\mathcal{G}|}|\Sigma|}} \exp\left(-\frac{1}{2}(z - \phi(s))^\top \Sigma^{-1}(z - \phi(s))\right) && (15)\end{aligned}$$

Diagonal-Covariance Gaussian. If we assume a diagonal covariance $\Sigma(s) = \text{diag}(\sigma^2(s))$, the discriminator will have the following form:

$$q_\lambda(z|s) = \frac{1}{\sqrt{(2\pi)^{|\mathcal{G}|}}\prod_i \sigma_i} \exp\left(-\sum_i \frac{1}{2\sigma_i^2}(z_i - \mu_i)^2\right), \quad \text{where } \mu_i = [\phi(s)]_i, \sigma_i = [\sigma(s)]_i \tag{16}$$

$$\log q_\lambda(z|s) = -|\mathcal{G}|\log(\sqrt{2\pi}) + \sum_i -\log(\sigma_i) + \sum_i \left(-\frac{1}{2\sigma_i^2}(z_i - \mu_i)^2\right) \tag{17}$$

As discussed in Section 4, the intrinsic reward function for training a goal-conditioned policy for a fixed goal z is given by $r(s) = \log q_\lambda(z|s) - \log p(z)$.

Equivalence to GCRL. It is straightforward to see that for a fixed value of σ_i (say $\sigma_i = 1.0$), Eq. (17) further reduces to

$$\log q_\lambda(z|s) = \text{Const} + \sum_i \left(-\frac{1}{2}(z_i - \mu_i)^2 \right) \tag{18}$$

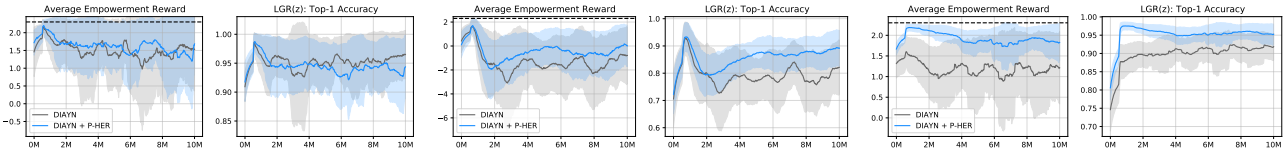
up to a constant factor. This can be interpreted as a smooth reward function for reaching a goal $z \in \mathcal{G}$, or the squared distance $\|z - \mu(s)\|_2^2$ between $\mu(s)$ and z in the goal space $\mathcal{G}$. A special case of this is when the goal space is set same as the state space ($\mathcal{G} = \mathcal{S}$) and a natural identity mapping $\mu(s) = s$ is used, where the smooth reward function in standard goal-conditioned RL (GCRL) is recovered.

C More Experimental Results

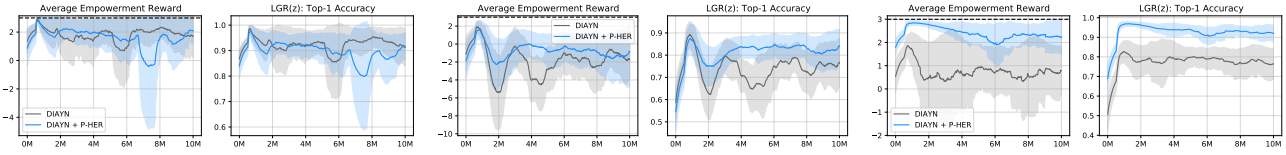
In this section, we present additional results for Section 6.3. Table 5 extends Table 4, showing the evaluation metrics for variants of VGCRL where continuous goal spaces of various dimensions are used. Figure 5 and Figure 6 show learning curve plots for the VGCRL variants with categorical and Gaussian posterior, respectively.

	$q_\lambda(z\|s)$	P-HER?	SN?	HalfCheetah $\mathcal{F}$	HalfCheetah LGR$_v(s)$	HalfCheetah LGR(z)	Ant $\mathcal{F}$	Ant LGR$_v(s)$	Ant LGR(z)	Humanoid $\mathcal{F}$	Humanoid LGR$_v(s)$	Humanoid LGR(z)
$\|\mathcal{G}\| = 2$	$\mathcal{N}(\mu(s), \text{fixed}^2)$	-	-	0.305	0.900	0.166	-0.128	0.398	0.242	-0.047	2.394	0.199
		✔	-	0.339	0.837	0.139	-0.110	0.678	0.306	-0.082	1.505	0.194
	$\mathcal{N}(\mu(s), \Sigma(s)^2)$	-	-	0.830	1.177	0.063	-4.669	0.478	0.265	0.677	0.393	0.080
		✔	-	0.403	0.720	0.079	-4.575	**0.289**	0.263	2.019	0.910	0.027
		-	✔	2.653	1.017	0.011	2.060	0.453	0.038	2.511	0.225	0.012
		✔	✔	**2.724**	1.074	**0.009**	**2.352**	0.511	**0.023**	**2.549**	**0.199**	**0.012**
	GMM ($K = 8$)	-	-	0.883	**0.707**	0.188	-4.344	0.640	0.360	1.141	1.637	0.072
		✔	-	1.183	2.032	0.181	-3.436	0.432	0.356	2.076	0.993	0.026
$\|\mathcal{G}\| = 5$	$\mathcal{N}(\mu(s), \text{fixed}^2)$	-	-	0.932	1.005	0.159	-0.590	1.005	0.382	0.239	1.461	0.202
		✔	-	-0.142	1.273	0.360	0.140	2.449	0.300	0.020	1.452	0.244
	$\mathcal{N}(\mu(s), \Sigma(s)^2)$	-	-	-0.731	1.251	0.172	-18.490	**0.306**	0.427	-3.597	0.538	0.147
		✔	-	-2.161	1.132	0.289	-0.108	2.423	0.303	1.207	0.206	0.074
		-	✔	**5.856**	**0.604**	0.019	2.548	0.925	0.091	4.509	0.460	0.040
		✔	✔	5.803	1.352	**0.017**	**4.349**	0.463	**0.039**	**5.203**	**0.203**	**0.026**
	GMM ($K = 8$)	-	-	-2.646	0.766	0.325	-16.196	0.367	0.486	-3.576	0.231	0.198
		✔	-	-3.091	1.065	0.404	-2.874	2.794	0.325	3.526	0.581	0.043
$\|\mathcal{G}\| = 10$	$\mathcal{N}(\mu(s), \text{fixed}^2)$	-	-	-0.145	0.855	0.346	-1.719	0.674	0.508	0.246	0.704	0.237
		✔	-	-0.825	0.866	0.407	-0.276	2.309	0.330	0.125	0.708	0.239
	$\mathcal{N}(\mu(s), \Sigma(s)^2)$	-	-	-3.688	1.381	0.384	-6.709	0.745	0.425	-3.399	0.313	0.221
		✔	-	-3.582	**0.640**	0.388	-0.190	3.989	0.324	-3.618	**0.244**	0.111
		-	✔	3.840	1.175	0.180	0.721	0.974	0.240	**8.134**	1.275	**0.061**
		✔	✔	**4.975**	0.874	**0.162**	**2.467**	0.674	**0.184**	6.349	0.262	0.072
	GMM ($K = 8$)	-	-	-5.137	1.250	0.404	-25.121	**0.307**	0.534	1.885	1.543	0.238
		✔	-	-6.162	0.835	0.399	-3.582	2.396	0.348	3.267	0.422	0.082
$\|\mathcal{G}\| = 20$	$\mathcal{N}(\mu(s), \text{fixed}^2)$	-	-	-2.024	0.901	0.438	-3.206	**0.486**	0.498	-0.656	0.622	0.320
		✔	-	-1.848	0.953	0.422	-0.527	3.038	0.331	-0.295	0.461	0.295
	$\mathcal{N}(\mu(s), \Sigma(s)^2)$	-	-	-3.754	**0.481**	0.377	-2.662	0.958	0.353	1.705	2.115	0.274
		✔	-	-4.704	1.648	0.385	-2.813	1.000	0.352	2.149	0.731	0.246
		-	✔	-0.176	1.054	0.318	-1.624	0.716	0.355	**7.176**	1.666	**0.191**
		✔	✔	**0.727**	1.066	**0.294**	**-0.503**	0.496	**0.320**	-0.350	**0.460**	0.340
	GMM ($K = 8$)	-	-	-6.294	1.254	0.394	-11.060	0.805	0.370	1.579	2.280	0.298
		✔	-	-10.647	1.725	0.397	-13.392	1.340	0.377	2.339	1.740	0.284

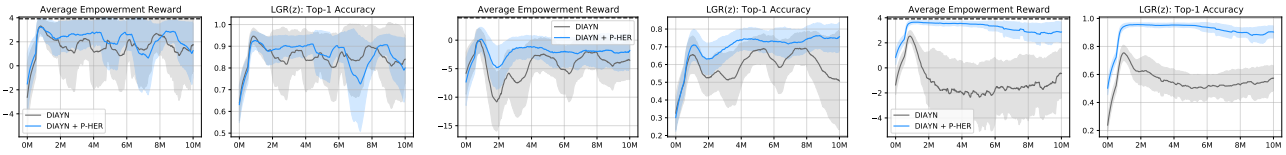
Table 5: An extended version of Table 4. We present a VGCRL-Gaussian variant where the variance is not learned but kept constant (fixed, e.g. $\log \sigma = 0$) and a variant where the variance is learned as a function of state s. VGCRL-GMM is when a Gaussian Mixture Model is used for the discriminator instead of a Gaussian distribution, where means, covariances, and mixture weights are learned through the neural network.



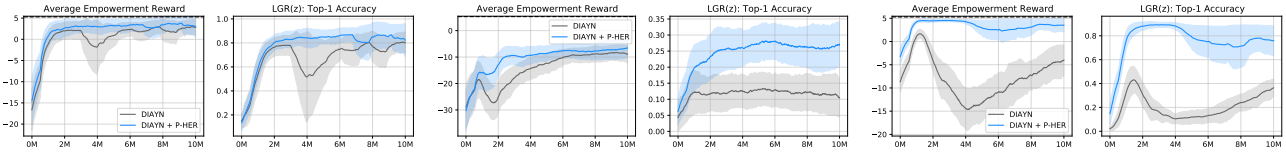
(a) HalfCheetah-v3 ($|\mathcal{G}| = 10$) (b) Ant-v3 ($|\mathcal{G}| = 10$) (c) Humanoid-v3 ($|\mathcal{G}| = 10$)



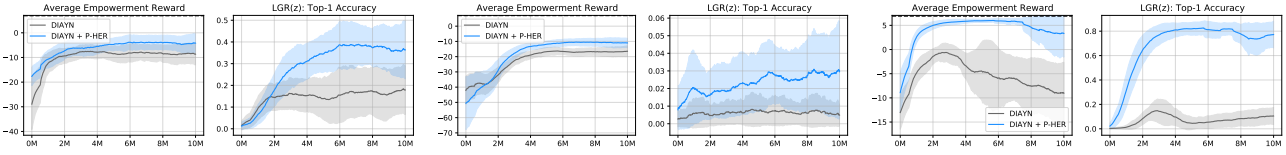
(d) HalfCheetah-v3 ($|\mathcal{G}| = 20$) (e) Ant-v3 ($|\mathcal{G}| = 20$) (f) Humanoid-v3 ($|\mathcal{G}| = 20$)



(g) HalfCheetah-v3 ($|\mathcal{G}| = 50$) (h) Ant-v3 ($|\mathcal{G}| = 50$) (i) Humanoid-v3 ($|\mathcal{G}| = 50$)

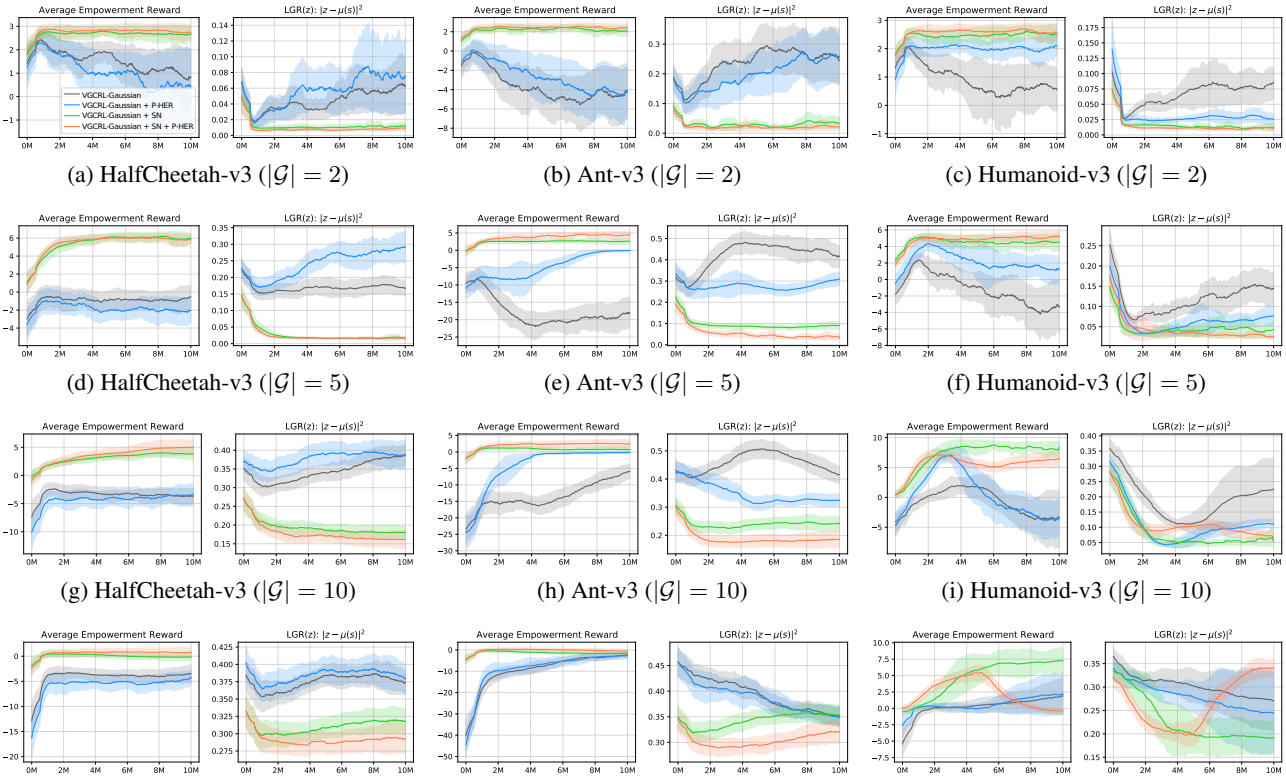


(j) HalfCheetah-v3 ($|\mathcal{G}| = 200$) (k) Ant-v3 ($|\mathcal{G}| = 200$) (l) Humanoid-v3 ($|\mathcal{G}| = 200$)



(m) HalfCheetah-v3 ($|\mathcal{G}| = 1000$) (n) Ant-v3 ($|\mathcal{G}| = 1000$) (o) Humanoid-v3 ($|\mathcal{G}| = 1000$)

Figure 5: Extension to Figure 4: Learning curves for VGCRL when discrete, categorical goal spaces are used. The dashed line denotes the maximum possible reward, achieved when the discriminator $q(z|s)$ is perfect at every time step. Overall, we can see P-HER improves the learning process of variational empowerment consistently across different environments and the dimensionality of the goal space.



(a) HalfCheetah-v3 ($|\mathcal{G}| = 2$) (b) Ant-v3 ($|\mathcal{G}| = 2$) (c) Humanoid-v3 ($|\mathcal{G}| = 2$)

(d) HalfCheetah-v3 ($|\mathcal{G}| = 5$) (e) Ant-v3 ($|\mathcal{G}| = 5$) (f) Humanoid-v3 ($|\mathcal{G}| = 5$)

(g) HalfCheetah-v3 ($|\mathcal{G}| = 10$) (h) Ant-v3 ($|\mathcal{G}| = 10$) (i) Humanoid-v3 ($|\mathcal{G}| = 10$)

(j) HalfCheetah-v3 ($|\mathcal{G}| = 20$) (k) Ant-v3 ($|\mathcal{G}| = 20$) (l) Humanoid-v3 ($|\mathcal{G}| = 20$)

Figure 6: Extension to Figure 4: Learning curves for VGCRL when continuous goal spaces and a family of Gaussian distribution is used for the variational posterior.

D Details of Environments

D.1 Windy PointMass

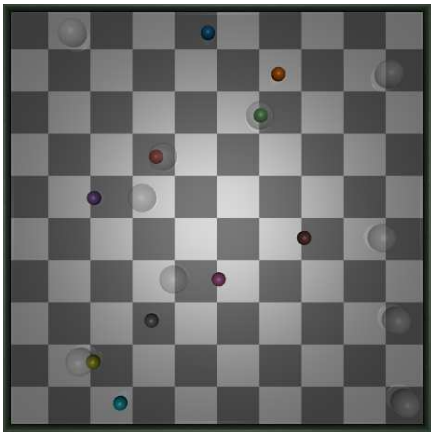


Figure 7: Windy PointMass (10-dimensional).

The (windy) point mass environment is a N-dimensional continuous control environment. The observation space is $2N$-dimensional, each of which describes the position and the velocity per dimension. Each point mass, one per dimension, can move left and right independently within the arena of range $(-1.5, 1.5)$. The action space is N-dimensional, each of which denoting the amount of velocity acceleration on each dimension. This generalizes common 2D (planar) point mass environments (Brockman et al., 2016; Tassa et al., 2018); indeed, it is exactly equivalent to the 2D point mass environments when $N = 2$. The positions of point masses are initialized randomly at each episode. Figure 7 shows a target goal location in overlaying transparent spheres (note that in the experiment we assumed the goal $\mathcal{G}$ to be a N-dimensional vector, same as the observation space) with $\mu(s) = s$.

For the windy point mass used in the experiment, we apply a random external force sampled from an uniform distribution $U(-R_i, R_i)$ to the point mass on dimension i, at every time step. The range of external force gets higher as the dimension index i increases; we use a profile of $R_i = 11 \times i$ for $N = 10$ (i.e., $R_0 = 0$ or no force on dimension 0, and $R_9 = 99$ for the last dimension $i = 9$) and $[R_0, R_1] = [0, 40]$ for $N = 2$. With such a large external force, the point mass on dimension $i = 9$ is almost uncontrollable, mostly bouncing around the external perturbation.

D.2 Expert State Generation

To generate target states $s^{1:N}$ in the latent goal reaching metric Section 6.2, we collected states (observations) randomly sampled from an expert policy's rollout trajectory. Expert policies are SAC agents successfully trained on the task with multiple target velocities rather than the standard task (i.e., only moving forward in HalfCheetah, Ant, Humanoid-v3, etc.). Similar to OpenAI gym's locomotion tasks (Brockman et al., 2016), we use a custom reward function $r_x = \mathrm{HuberLoss}(\text{target } x \text{ velocity} - \text{achieved } y \text{ velocity})$ and a similar one for r_y to let the robot move in some directions with the desired target velocities. The set of target velocities (v_x, v_y) were constructured from the choices of $(-2, -1, -0.5, 0, 0.5, 1, 2)$. We used the SAC implementation from (Guadarrama et al., 2018) with a default hyperparameter setting to train expert policies. We sample 6 random states from each expert policy, yielding a total of $7^2 \times 6 = 294$ (or $7 \times 6 = 42$ for HalfCheetah) target states for each environment. Altogether, this dataset provides a set of states where the agent is posing or moving in diverse direction.

E Implementation Details

For training the goal-conditioned policy, we used Soft Actor-Critic (SAC) (Haarnoja et al., 2018) algorithm with the default hyperparameter setting. To represent the discriminator $q(z|s)$ with a neural network, we simply used a 2-layer MLP with (256, 256) hidden units and ReLU activations. The heads $\mu(s)$ and $\log \sigma(s)$ are obtained through a linear layer on top of the last hidden layer. For Gaussian VGCRLs, we employed an uniform prior $p(z) = [-1, 1]^{|\mathcal{G}|}$ and also applied $\tanh$ bijections to the variational posterior distribution $q_\lambda(z|s)$ to make the domain of z fit $[-1, 1]^{|\mathcal{G}|}$. We also clipped the output of $\log \sigma(s)$ with the clip range $[\log(0.3), \log(10.0)]$ for the sake of numerical stability, so that the magnitude of posterior evaluations (and hence the reward) does not get too large.

For spectral normalization, we swept hyperparameters σ that control the Lipschitz constant over a range of $[0, 0.5, 0.95, 2.0, 5.0, 7.0]$, and chose a single value $\sigma = 2.0$ that worked best in most cases. The number of mixtures used in Gaussian Mixture Models is $K = 8$. The heads $\mu(s)$, $\log \sigma(s)$, mixture weights $\alpha(s)$ are obtained through a linear layer on top of the last hidden layer.