Appendices

A. Discussion on the choice of Proximal Policy Optimization as a baseline

A general learning process of RL can be described using policy iteration, which consists of two iterative phases: policy evaluation and policy improvement (Sutton & Barto, 1998). In policy iteration, the value function is assumed to be exact, meaning that given policy, the value function is learned until convergence for the entire state space, which results in a strong bound on the rate of convergence to the optimal value and policy (Puterman, 1994).

But the exact value method is often infeasible from resource limitation since it requires multiple sweeps over the entire state space. Therefore, in practice, the value function is approximated, i.e. it is not trained until convergence nor across the entire state space on each iteration. The approximate version of the exact value function method, also known as *asynchronous value iteration*, still converges to the unique optimal solution of the Bellman optimality operator. However, the Bellman optimality only describes the limit convergence, and thus the best we can practically consider is to measure the improvement on each update step.

Bertsekas & Tsitsiklis (1996) have shown that, when we approximate the value function V_π of some policy π with $\tilde{V}$, the lower bound of a greedy policy π' is given by

$$V_{\pi'}(x) \geq V_\pi(x) - \frac{2\gamma\varepsilon}{1-\gamma}, \tag{22}$$

where $\varepsilon = \max_x |\tilde{V}(x) - V_\pi(x)|$ is the L_∞ error of value approximation $\tilde{V}$. This means a greedy policy from an approximate value function guarantees that its exact value function will not degrade more than $\frac{2\gamma\varepsilon}{1-\gamma}$. However, there is no guarantee on the improvement, i.e. $V_{\pi'}(x) > V_\pi(x)$ (Kakade & Langford, 2002).

As a solution to this issue, Kakade & Langford (2002) have proposed a policy updating scheme named *conservative policy iteration*,

$$\pi_{new}(a|x) = (1-\alpha)\pi_{old}(a|x) + \alpha\pi'(a|x), \tag{23}$$

which has an explicit lower bound on the improvement

$$\eta(\pi_{new}) \geq L_{\pi_{old}}(\pi_{new}) - \frac{2\epsilon\gamma}{(1-\gamma)^2}\alpha^2, \tag{24}$$

where $\epsilon = \max_x |\mathbb{E}_{\pi'}[A_\pi(x,a)]|$, $A_\pi(x,a) = Q(x,a) - V(x)$ is the advantage function, $\eta(\pi)$ denotes the expected sum of reward under the policy π,

$$\eta(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(x_t, a_t)\right], \tag{25}$$

and $L_{\pi_{old}}$ is the local approximation of η with the state visitation frequency under the old policy.

From the definition of distributional Bellman optimality operator in (6), one can see that the lower bound in (24) also holds when π' is greedy with respect to the expectation of the value distribution, i.e., $\mathbb{E}_{x'\sim P}[Z(x', a')]$. Thus the improvement of the distributional Bellman update is guaranteed in expectation under conservative policy iteration, and the value functions are guaranteed to converge in distribution to a fixed point by γ-contraction.

Schulman et al. (2015) takes this further, suggesting an algorithm called trust region policy optimization (TRPO), which extends conservative policy iteration to a general stochastic policy by replacing α with Kullback-Leibler (KL) divergence between two policies,

$$D_{KL}^{max}(\pi, \tilde{\pi}) = \max_x D_{KL}\left(\pi(\cdot|x) \| \tilde{\pi}(\cdot|x)\right). \tag{26}$$

Then, the newly formed objective is to maximize the following, which is a form of constraint optimization with penalty:

$$\hat{\mathbb{E}}_t\left[\frac{\pi(a_t|x_t)}{\tilde{\pi}(a_t|x_t)}\hat{A}_t - \beta D_{KL}(\pi(\cdot|x_t), \tilde{\pi}(\cdot|x_t))\right] = \hat{\mathbb{E}}_t\left[r_t(\pi)\hat{A}_t - \beta D_{KL}(\pi(\cdot|x_t), \tilde{\pi}(\cdot|x_t))\right]. \tag{27}$$

where $r(\pi)$ refers to the ratio $r(\pi) = \frac{\pi(a_t|x_t)}{\tilde{\pi}(a_t|x_t)}$. However, in practice, choosing a fixed penalty coefficient β is difficult and thus Schulman et al. (2015) uses hard constraint instead of the penalty.

$$\max_{\theta} \hat{\mathbb{E}}_t \left[r_t(\pi) \hat{A}_t \right] \tag{28}$$

$$\text{s.t. } D_{KL}(\pi(\cdot|x_t), \tilde{\pi}(\cdot|x_t)) \leq \delta \tag{29}$$

Schulman et al. (2017) simplifies the loss function even further in proximal policy optimization (PPO) by replacing KL divergence with ratio clipping between the old and the new policy with the following:

$$L^{CLIP} = \hat{\mathbb{E}}_t \left[\min \left(r_t(\pi)\hat{A}_t, \text{clip}(r_t(\pi), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right]. \tag{30}$$

Thus, by using PPO as the baseline, we aim to optimize the value function via unique point convergence of distributional Bellman operator for a policy being approximately updated under the principle of conservative policy.

B. Expectation value of $Z_t^{(\lambda)}$

Continuing from (13), let us define a random variable that has a cumulative distribution function of $\mathbb{E}[\tilde{F}_Z]$ as $Z_t^{(\lambda)}$. Then, its cumulative distribution function is given by

$$F_{Z_t^{(\lambda)}} = (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} F_{Z_t^{(n)}}. \tag{31}$$

If we assume that the support of $Z_t^{(\lambda)}$ is defined in the extended real line $[-\infty, \infty]$,

$$\mathbb{E}[Z_t^{(\lambda)}] = \int_0^{\infty} \left(1 - F_{Z_t^{(\lambda)}}\right) dz - \int_{-\infty}^{0} F_{Z_t^{(\lambda)}} dz \tag{32}$$

$$= \int_0^{\infty} \left(1 - (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} F_{Z_t^{(n)}}\right) dz - \int_{-\infty}^{0} (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} F_{Z_t^{(n)}} dz \tag{33}$$

$$= (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} \left[\int_0^{\infty} \left(1 - F_{Z_t^{(n)}}\right) dz - \int_{-\infty}^{0} F_{Z_t^{(n)}} dz \right] \tag{34}$$

$$= (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)} = G_t^{(\lambda)}. \tag{35}$$

Thus we can arrive at the desired expression of $\mathbb{E}[Z_t^{(\lambda)}] = G_t^{(\lambda)}$.

C. Distributional Bellman operator as a contraction in Cramér metric space

The Cramér distance possesses the following characteristics (detailed derivation of each can be found in (Bellemare et al., 2017b)):

$$l_p(A+X, A+Y) \leq l_p(X,Y), \qquad l_p(cX, cY) \leq |c|^{1/p} l_p(X,Y). \tag{36}$$

Using the above characteristics, the Bellman operator in l_p divergence is

$$\begin{aligned} l_p\left(\mathcal{T}^{\pi} Z_1(x,a), \mathcal{T}^{\pi} Z_2(x,a)\right) &= l_p(R(x,a) + \gamma P^{\pi} Z_1(x,a), R(x,a) + \gamma P^{\pi} Z_2(x,a)) \\ &\leq |\gamma|^{1/p} l_p(P^{\pi} Z_1(x,a), P^{\pi} Z_2(x,a)) \\ &\leq |\gamma|^{1/p} \sup_{x',a'} l_p(Z_1(x',a'), Z_2(x',a')). \end{aligned} \tag{37}$$

Substituting the result into the definition of the maximal form of the Cramér distance yields

$$\begin{aligned}\bar{l}_p(\mathcal{T}^\pi Z_1, \mathcal{T}^\pi Z_2) &= \sup_{x,a} l_p(\mathcal{T}^\pi Z_1(x,a), \mathcal{T}^\pi Z_2(x,a)) \\ &\leq |\gamma|^{1/p} \sup_{x',a'} l_p(Z_1(x',a'), Z_2(x',a')) \\ &= |\gamma|^{1/p} \bar{l}_p(Z_1, Z_2).\end{aligned} \tag{38}$$

Thus the distributional Bellman operator is a $|\gamma|^{1/p}$-contraction mapping in the Cramér metric space, which was also proven in Rowland et al. (2019).

Similar characteristics as in (36) can be derived for the energy distance

$$\mathcal{E}(A+X, A+Y) \leq \mathcal{E}(X,Y), \qquad \mathcal{E}(cX, cY) = c\mathcal{E}(X,Y), \tag{39}$$

showing that the distributional Bellman operator is a γ-contraction in energy distance

$$\mathcal{E}(\mathcal{T}^\pi Z_1, \mathcal{T}^\pi Z_2) \leq \gamma \mathcal{E}(Z_1, Z_2). \tag{40}$$

D. Loss Functions

As in other policy gradient methods, our value distribution approximator models the distribution of the value, $V(x_t)$, not the state-action value $Q(x_t, a_t)$, and denote it as $Z_\theta(x_t)$ parametrized with θ, whose cumulative distribution function is defined as

$$F_{Z_\theta(x_t)} = \sum_{a \in \mathcal{A}} \pi(a, x_t) F_{Z(x_t, a)}. \tag{41}$$

Below, we provide the complete loss function of value distribution approximation for each of the cases used in experiments (Section 5).

D.1. Implicit Quantile + Huber quantile (IQAC)

For the value loss of IQAC, we follow the general flow of Huber quantile loss described in Dabney et al. (2018b). For two random samples $\tau, \tau' \sim U([0,1])$,

$$\delta_t^{\tau,\tau'} = Z_t^{(\lambda)}(x_t, a_t; \tau') - Z_\theta(x_t; \tau) \tag{42}$$

where $Z_t^{(\lambda)}$ is generated via SR(λ) and $Z(x;\tau) = F_Z^{-1}(\tau)$ is realization of $Z(X)$ given $X = x$ and τ. Then, the full loss function of value distribution is given by

$$L_{Z_\theta} = \frac{1}{N'} \sum_{i=1}^{N} \sum_{j=1}^{N'} \rho_{\tau_i}^{\kappa} \left(\delta_t^{\tau_i, \tau_j'} \right) \tag{43}$$

where N and N' are number of samples of τ, τ', respectively, and ρ is the Huber quantile loss

$$\rho_\tau^\kappa(\delta_{ij}) = |\tau - \mathbb{I}\{\delta_{ij} < 0\}| \frac{L_\kappa(\delta_{ij})}{\kappa}, \quad \text{with} \tag{44}$$

$$L_\kappa(\delta_{ij}) = \begin{cases} \frac{1}{2}\delta_{ij}^2, & \text{if} |\delta_{ij}| \leq \kappa \\ \kappa(|\delta_{ij}| - \frac{1}{2}\kappa), & \text{otherwise.} \end{cases} \tag{45}$$

D.2. Implicit Quantile + Energy Distance (IQAC-E)

Here, we replace the Huber quantile loss in (43) with sample-based approximation of energy distance defined in (20).

$$L_{Z_\theta} = \frac{2}{NN'} \sum_{i=1}^{N} \sum_{j=1}^{N'} \left| \delta_t^{\tau_i, \tau_j'} \right| - \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \left| \delta_t^{\tau_i, \tau_{i'}} \right| - \frac{1}{N'^2} \sum_{j=1}^{N'} \sum_{j'=1}^{N'} \left| \delta_t^{\tau_{j'}', \tau_j'} \right| \tag{46}$$

D.3. Gaussian Mixture + Energy Distance (GMAC)

Unlike the two previous losses, which use samples at τ generated by the implicit quantile network $Z_\theta(x_t;\tau)$, here we discuss a case in which the distribution is k-component Gaussian mixture parameterized with (μ_k, σ_k^2, w_k).

Using the expectation of a folded normal distribution, we define δ between two Gaussian distributions as

$$\delta(\mu_i,\sigma_i^2,\mu_j,\sigma_j^2)=\sqrt{\frac{2}{\pi}}\sqrt{\sigma_i^2+\sigma_j^2}\exp\left(-\frac{(\mu_i-\mu_j)^2}{2(\sigma_i^2+\sigma_j^2)}\right)+(\mu_i-\mu_j)\left[1-2\Phi\left(\frac{(\mu_i-\mu_j)}{\sqrt{2}}\right)\right]. \tag{47}$$

Let $Z_\theta(x)$ and $Z_t^{(\lambda)}$ be Gaussian mixtures parameterized with $(\mu_{\theta i},\sigma_{\theta i}^2,w_{\theta i}),(\mu_{\lambda j},\sigma_{\lambda j}^2,w_{\lambda j})$, respectively. Then, the loss function for the value head is given by

$$\begin{aligned}L_{Z_\theta}=&\frac{2}{NN'}\sum_{i=1}^{N}\sum_{j=1}^{N'}w_{\theta i}w_{\lambda j}\delta(\mu_{\theta i},\sigma_{\theta i}^2,\mu_{\lambda j},\sigma_{\lambda j}^2)\\&-\frac{1}{N^2}\sum_{i=1}^{N}\sum_{i'=1}^{N}w_{\theta i}w_{\theta i'}\delta(\mu_{\theta i},\sigma_{\theta i}^2,\mu_{\theta i'},\sigma_{\theta i'}^2)\\&-\frac{1}{N'^2}\sum_{j=1}^{N'}\sum_{j'=1}^{N'}w_{\lambda j}w_{\lambda j'}\delta(\mu_{\lambda j},\sigma_{\lambda j}^2,\mu_{\lambda j'},\sigma_{\lambda j'}^2).\end{aligned} \tag{48}$$

E. Pseudocode of GMAC

Algorithm 2 Pseudocode of GMAC

Input: Initial policy parameters θ_0, initial value function parameters ϕ_0, length of trajectory N, number of environments E, clipping factor ϵ, discount factor γ, weight parameter λ
repeat
  for $e = 1$ **to** E **do**
    Collect samples of discounted sum of rewards $\{Z_1, \dots, Z_N\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment
    Compute the parameters (μ_i, σ_i, w_i) for each of the λ-returns $\{Z_1^{(\lambda)}, \dots, Z_{N-1}^{(\lambda)}\}$ by SR(λ) (Algorithm 1)
    Compute advantage estimates $\hat{A}_t$ using GAE (Schulman et al., 2016), based on the current value function V_{ϕ_k}
  end for
  Gather the data from E environments
  Update policy using the clipped surrogate loss:

$$\theta_{k+1} = \arg\max_{\theta} \mathbb{E}\left[\min\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}\hat{A}_t,\ g(\epsilon, \hat{A}_t)\right)\right]$$

  via stochastic gradient ascent.
  Update value function using the energy distance between Gaussian mixtures (Equation 20):

$$\phi_{k+1} = \arg\min_{\phi} \mathbb{E}\left[\mathcal{E}\left(V_\phi(s_t),\ Z_t^{(\lambda)}\right)\right]$$

  via stochastic gradient descent.
until Final update step

The clipping function $g(\epsilon, A)$ shown in the algorithm is defined as follows:

$$g(\epsilon, A) = \begin{cases} (1+\epsilon)A & \text{if } A \geq 0 \\ (1-\epsilon)A & \text{if } A < 0 \end{cases}$$

Note that expectation of each loss is taken over the collection of trajectories and environments.

F. Implementation Details

For producing a categorical distribution, a softmax layer was added to the output of the network. For producing a Gaussian mixture distribution, the mean of each Gaussian is simply the output of the network, the variance is kept positive by running the output through a softplus layer, and the weights of each Gaussian is produced through the softmax layer. Since

Table 2: Network architecture for GMAC on atari

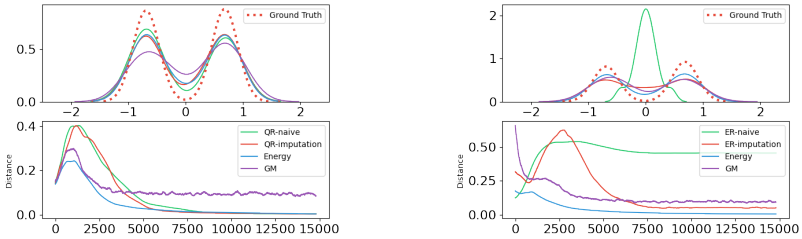
Layer Type	Specifications		Filter size, stride
Input	84 x 84 x 4		
Conv1	20 x 20 x 32		8 x 8 x 32, 4
Conv2	9 x 9 x 64		4 x 4 x 64, 2
Conv3	7 x 7 x 32		3 x 3 x 32, 1
FC1	512		
Heads	Policy	Value	
(FC)	action dim	# of modes (= 5)	

our proposed method takes an architecture which only changes the value head of the original PPO network, we base our hyperparameter settings from the original paper (Schulman et al., 2017). We performed a hyperparameter search on a subset of variables: optimizers={Adam, RMSprop}, learning rate={2.5e-4, 1.0e-4}, number of epochs={4, 10}, batch size={256, 512}, and number of environments={16, 32, 64, 128} over 3 atari tasks of Breakout, Gravitar, and Seaquest, for which there was no degrade in the performance of PPO.

Table 3: Parameter settings for training Atari games and PyBullet tasks

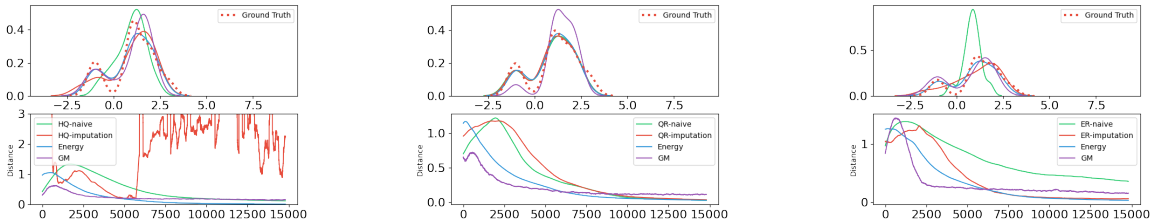
Task	Atari				PyBullet			
Parameter	PPO	IQ	IQAC-E	GMAC	PPO	IQ	IQAC-E	GMAC
Learning rate	2.5e-4				1e-4			
Optimizer	Adam				Adam			
Total frames	2e8				5e7			
Rollout steps	128				512			
Skip frame	4				1			
Environments	64				64			
Minibatch size	512				2048			
Epoch	4				10			
γ	0.99				0.99			
λ	0.95				0.95			
Dirac samples	-		64	-	-		64	-
Mixtures	-	-	-	5	-	-	-	5

G. More Experimental Results



(a) Quantile Regression (b) Expectile Regression

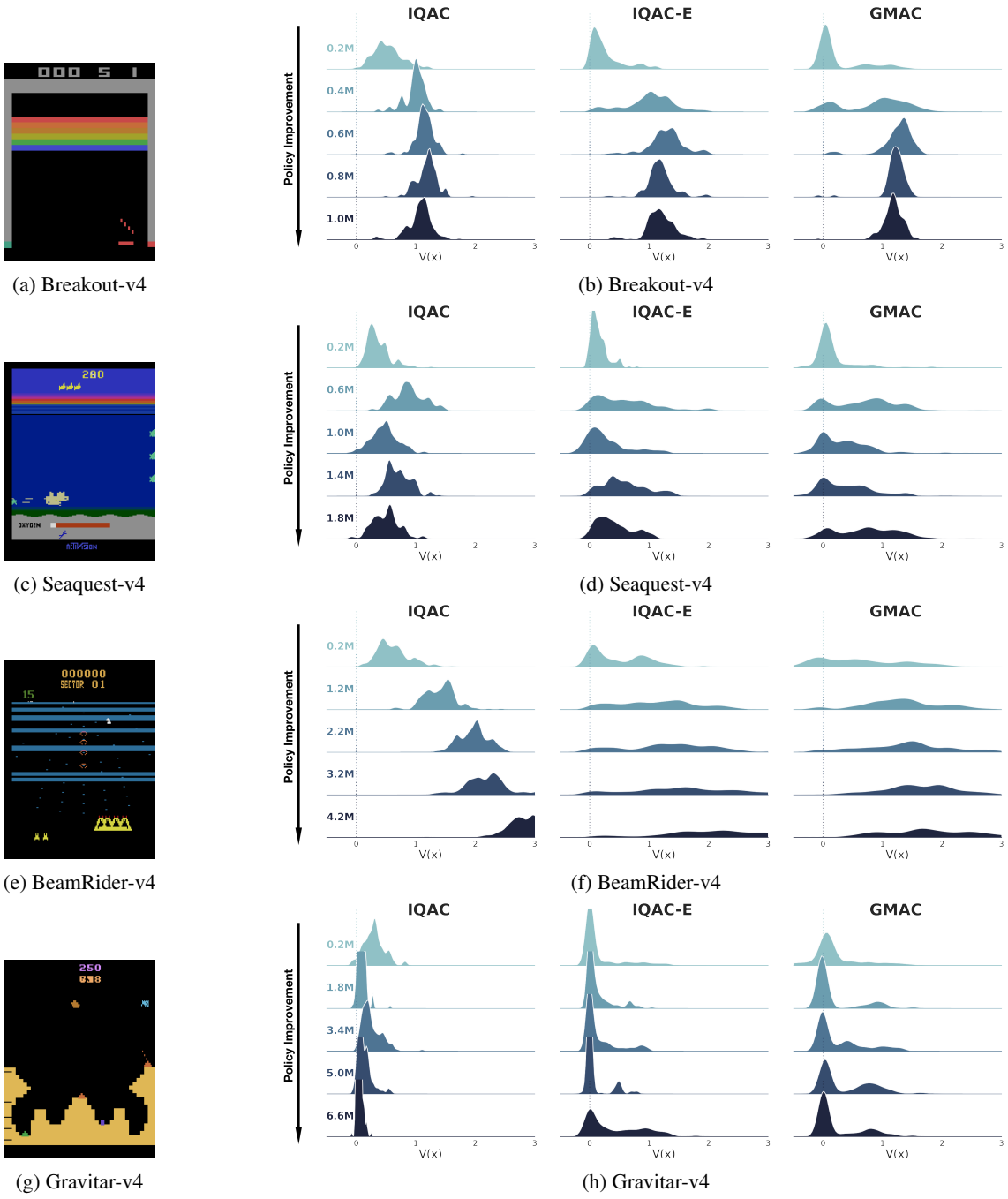
Figure 6: In addition to Figure 3, quantile and expectile regressions are also evaluated in the 5-state MDP against IQE and GMM with their respective loss functions.



(a) Huber Quantile Regression (b) Quantile Regression (c) Expectile Regression

Figure 7: Evaluation of the five-state MDP under tabular setting on a asymmetric reward distribution). Huber quantile($\kappa = 1$), quantile, and expectile regressions are compared to the energy distance minimization between samples and Gaussian mixtures.

Here we provide more details on the five-state MDP presented in Figure 3. For each cases in the figure, 15 diracs are used for quantile based methods and 5 mixtures are used for GMM to balance the total number of parameters required to represent a distribution. For the cases with the label "naive", the network outputs (quantiles, expectiles, etc.) are used to create the plot. On the other hand, the cases with "imputation" labels apply appropriate imputation strategy to the statistics to produce samples which are then used to plot the distribution. Sample based energy-distance was used to calculate the distance from the true distribution for all cases.



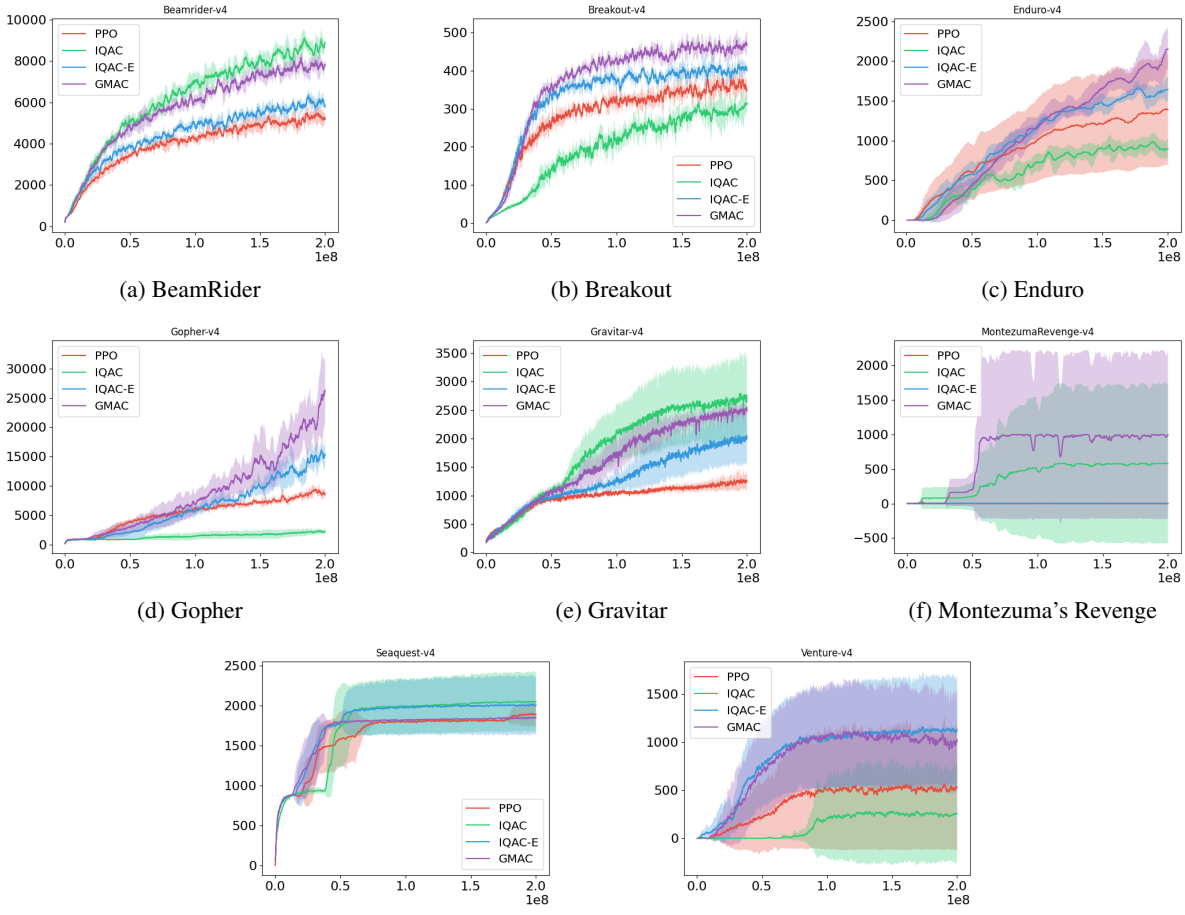
(a) Breakout-v4 (b) Breakout-v4

(c) Seaquest-v4 (d) Seaquest-v4

(e) BeamRider-v4 (f) BeamRider-v4

(g) Gravitar-v4 (h) Gravitar-v4

Figure 8: More value distributions of different tasks. All states are chosen such that the agent is in place of near-death or near positive score. Thus, when the policy is not fully trained, such as in a very early stage, the value distribution should include a notion of death indicated by a mode positioned at zero. In all games, IQN + Huber quantile (IQAC) fails to correctly capture a mode positioned at zero while the other two methods, IQN + energy distance (IQAC-E) and GMM + energy distance (GMAC) captures the mode in the early stage of policy improvement. Again, the visual representation is *maxpool* of the 4 frame stacks in the given state.

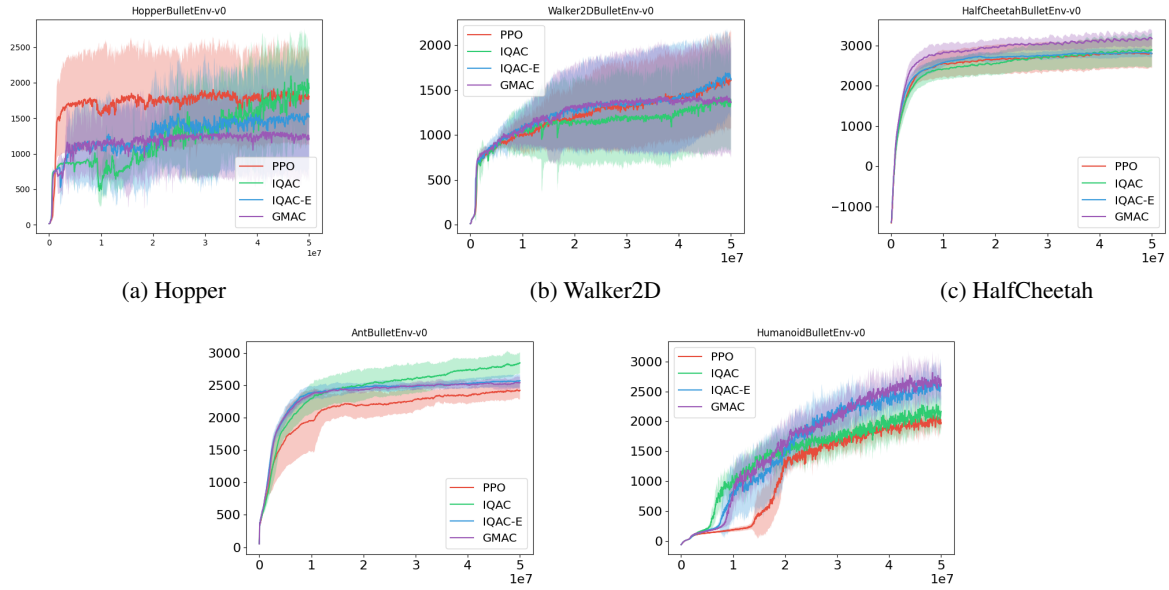


(a) BeamRider (b) Breakout (c) Enduro

(d) Gopher (e) Gravitar (f) Montezuma's Revenge

(g) Seaquest (h) Venture

Figure 9: Raw learning curves over 5 random seeds for 8 selected Atari games. The y-axis is raw score and x-axis is in frames. The tasks are selected considering the stochasticity in games, score gap between the previous scalar and distributional method (DQN (Mnih et al., 2015) vs. IQN (Dabney et al., 2018a)), and complexity of the game in terms of exploration.



(a) Hopper (b) Walker2D (c) HalfCheetah

(d) Ant (e) Humanoid

Figure 10: Raw learning curves over 5 random seeds for 5 selected PyBullet continuous control tasks. The y-axis is in raw score and x-axis is in environment steps.

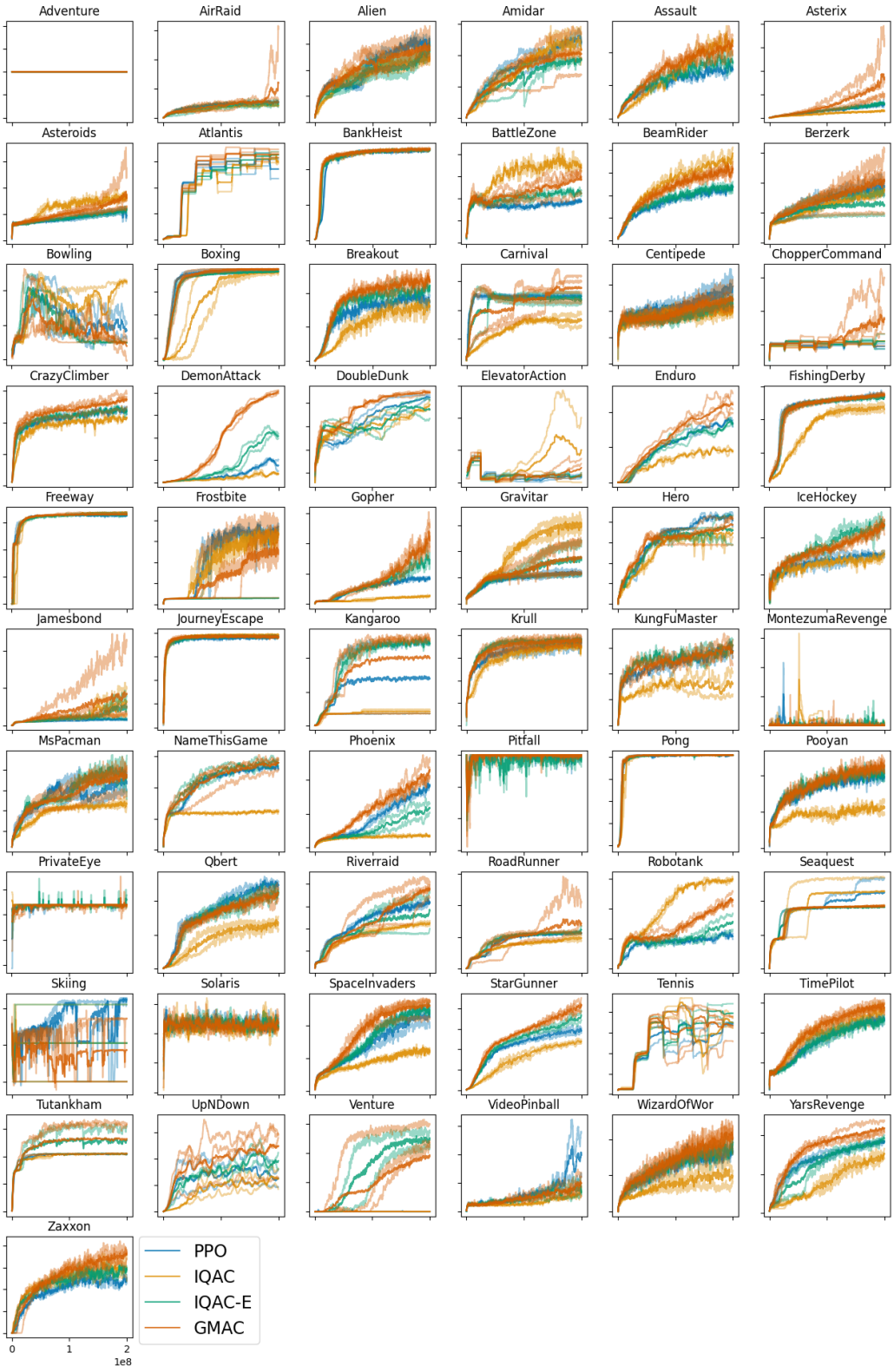


Figure 11: Full learning curves of 61 atari games from ALE

Table 4: Average score over last 100 episodes in 200M frame collected for training 61 atari games. The algorithms are trained using same single seed and hyperparameters. Random and Human scores are taken from Wang et al.

GAMES	RANDOM	HUMAN	PPO	IQAC	IQAC-E	GMAC
Adventure	NA	NA	**0.00**	**0.0**	**0.00**	**0.00**
AirRaid	NA	NA	10,205.75	8,304.50	7,589.50	**62,328.75**
Alien	227.80	**7,127.70**	2,918.60	2,505.80	2,704.20	3,687.10
Amidar	5.80	**1,719.50**	1,244.12	1,210.40	932.11	1,363.72
Assault	222.40	742.00	7,508.03	**12,053.03**	8,589.55	10,281.73
Asterix	210.00	8,503.30	13,367.00	6,868,00	15,426.00	**22,650.00**
Asteroids	719.10	**47,388.70**	2,088.10	3,428.10	2,332.00	2,597.50
Atlantis	12,850.00	29,028.10	3,073,796.00	2,916,292.00	**3,373,635.00**	3,141,534.00
BankHeist	14.20	753.10	1,263.80	1,265.80	**1,286.60**	1,274.30
BattleZone	2,360.00	**37,187.50**	18,540.00	35,160.00	21,310.00	32,490.00
BeamRider	363.90	**16,926.50**	5,913.84	8,968.58	6,507.68	8,718.72
Berzerk	123.70	2,630.40	1,748.10	1,682.70	887.50	**3,081.20**
Bowling	23.10	**160.70**	33.54	65.81	30.00	19.39
Boxing	0.10	12.10	96.79	97.84	97.10	**99.89**
Breakout	1.70	30.50	384.29	296.91	445.64	**462.68**
Carnival	NA	NA	5,079.20	2,865.40	4,401.00	**6,344.20**
Centipede	2,090.90	**12,017.00**	5,205.25	4,085.38	4,864.69	4,303.10
ChopperCommand	811.00	**7,387.90**	872.00	1,096.00	1,314.00	1,795.00
CrazyClimber	10,780.50	35,829.40	112,640.00	107,375.00	121,550.00	**125,143.00**
DemonAttack	152.10	1,971.00	50,590.65	40,369.90	236,839.85	**411,118.85**
DoubleDunk	-18.60	-16.40	-3.26	-6.30	-8.28	**-2.72**
ElevatorAction	NA	NA	10,449.00	50.00	8,516.00	**14,254.00**
Enduro	0.00	860.50	1,588.68	861.65	1,612.17	**2,092.65**
FishingDerby	-91.70	-38.70	37.01	9.12	33.13	**37.52**
Freeway	0.00	29.60	32.53	32.96	**33.68**	32.84
Frostbite	62.50	**4,334.70**	3,571.50	3,550.10	307.10	3,392.40
Gopher	257.60	2,412.50	8,199.80	2,932.20	16,934.60	**25,266.80**
Gravitar	173.00	**3,351.40**	1,151.50	2,798.00	2,178.50	2,401.00
Hero	1,027.00	30,826.40	37,725.55	32,568.50	**43,065.95**	41,509.05
IceHockey	-11.20	0.90	-1.90	-1.98	**2.13**	0.34
Jamesbond	29.00	302.80	642.50	**4,913.50**	961.00	1,512.00
JourneyEscape	NA	NA	-607.00	**-339.00**	-840.00	-680.00
Kangaroo	52.00	3,035.00	1,742.00	2,368.00	12,208.00	**12,909.00**
Krull	1,598.00	2,665.50	**9,605.51**	8,643.09	9,514.03	9,127.63
KungFuMaster	258.50	22,736.50	26,846.00	12,006.00	33,378.00	**31,025.00**
MontezumaRevenge	0.00	**4,753.30**	0.00	3.00	0.00	0.00
MsPacman	307.30	**6,951.60**	3,674.20	2,450.70	4,699.00	3,884.40
NameThisGame	2,292.30	8,049.00	13,229.10	6,027.80	13,454.00	**14,031.30**
Phoenix	761.40	7,242.60	37,263.70	6,366.20	26,154.00	**42,664.00**
Pitfall	-229.40	**6,463.70**	0.00	0.00	-18.86	-3.36
Pong	-20.70	14.60	20.87	20.68	20.88	**20.97**
Pooyan	NA	NA	4,018.95	1,819.85	3,674.70	**4,178.65**
PrivateEye	24.90	**69,571.30**	100.00	71.51	196.30	100.00
Qbert	163.90	13,455.00	**25,519.25**	11,728.25	21,599.50	23,176.25
Riverraid	1,338.50	17,118.00	15,983.00	10,840.80	18,073.40	**19,761.30**
RoadRunner	11.50	7,845.00	56,321.00	44,685.00	56,121.00	**68,272.00**
Robotank	2.20	11.90	23.45	**60.79**	36.69	45.82
Seaquest	68.40	**42,054.70**	1,832.00	2,704.40	1,814.60	1,838.40
Skiing	-17,098.10	**-4,336.90**	-7,958.81	-8,987.12	-29,971.02	-29,975.52
Solaris	1,236.30	**12,326.70**	2,452.80	2,342.60	2,204.80	2,579.20
SpaceInvaders	148.00	1,668.70	**2,544.10**	1,177.65	2,410.90	2,228.30
StarGunner	664.00	10,250.00	74,848.00	57,053.00	97,450.00	**104,188.00**
Tennis	-23.80	-8.30	-8.16	-6.17	-7.54	**-5.90**
TimePilot	3,568.00	5,229.20	12,157.00	**14,746.00**	11,704.00	13,227.00
Tutankham	11,40	167.60	206.32	**210.66**	208.72	209.82
UpNDown	533.40	11,693.20	158,629.50	84,962.70	**161,328.40**	129,243.70
Venture	0.00	1,187.50	0.00	0.00	**1,339.00**	1,181.00
VideoPinball	16,256.90	17,667.90	**279,504.81**	55,113.30	59,988.90	55,272.82
WizardOfWar	563.50	4,756.50	8,749.00	5,688.00	9,165.00	**11,388.00**
YarsRevenge	3,092.90	54,576.90	92,709.94	83,136.68	100,082.55	**103,895.05**
Zaxxon	32.50	9,173.00	13,336.00	11,886.00	14,882.00	**18,436.00**