

## A PROOF OF EQUATION (11)

Given the initial state-action pair  $(s_0, a_0)$ , the bellman expectation equation (Sutton, 2018) is written as:

$$\begin{aligned} Q^\pi(s_0, a_0) &= \mathbb{E}_{s_1 \sim P(\cdot | s_0, a_0)} [r(s_0, a_0) + \gamma V^\pi(s_1)] \\ &= \mathbb{E}_{\substack{s_1, \dots, s_H \sim P \\ a_1, \dots, a_{H-1} \sim \pi}} \left[ \sum_{t=0}^{H-1} \gamma^t r(s_t, a_t) + \gamma^H V^\pi(s_H) \right] \end{aligned} \quad (15)$$

According to the definition of optimal Q-value:  $Q^*(s_0, a_0) = \max_\pi Q^\pi(s_0, a_0)$ , we have:

$$\begin{aligned} Q^*(s_0, a_0) &= \max_\pi \mathbb{E}_{\substack{s_1, \dots, s_H \sim P \\ a_1, \dots, a_{H-1} \sim \pi}} \left[ \sum_{t=0}^{H-1} \gamma^t r(s_t, a_t) + \gamma^H V^\pi(s_H) \right] \\ &= \max_\pi \mathbb{E}_{\substack{s_1, \dots, s_H \sim P \\ a_1, \dots, a_{H-1} \sim \pi}} \left[ \sum_{t=0}^{H-1} \gamma^t r(s_t, a_t) + \gamma^H \max_\pi V^\pi(s_H) \right] \\ &= \max_\pi \mathbb{E}_{\substack{s_1, \dots, s_H \sim P \\ a_1, \dots, a_{H-1} \sim \pi}} \left[ \sum_{t=0}^{H-1} \gamma^t r(s_t, a_t) + \gamma^H V^*(s_H) \right] \\ &= \max_\pi \mathbb{E}_{\substack{s_1, \dots, s_H \sim P \\ a_1, \dots, a_{H-1} \sim \pi}} \left[ \sum_{t=0}^{H-1} \gamma^t r(s_t, a_t) + \gamma^H \max_{a_H} Q^*(s_H, a_H) \right] \end{aligned} \quad (16)$$

For Equation (16), we derive the optimal policy  $\pi$  is the greedy policy selecting actions with the greatest  $Q^*$ -value as follows:

$$\begin{aligned} Q^*(s_0, a_0) &= \max_\pi \mathbb{E}_{\substack{s_1, \dots, s_H \sim P \\ a_1, \dots, a_{H-1} \sim \pi}} \left[ \sum_{t=0}^{H-1} \gamma^t r(s_t, a_t) + \gamma^H \max_{a_H} Q^*(s_H, a_H) \right] \\ &= \max_\pi \mathbb{E}_{\substack{s_1, \dots, s_{H-1} \sim P \\ a_1, \dots, a_{H-2} \sim \pi}} \left[ \sum_{t=0}^{H-2} \gamma^t r(s_t, a_t) + \gamma^{H-1} \max_{a_{H-1}} \mathbb{E}_{s_H \sim P} \left[ r(s_{H-1}, a_{H-1}) + \gamma \max_{a_H} Q^*(s_H, a_H) \right] \right] \\ &= \max_\pi \mathbb{E}_{\substack{s_1, \dots, s_{H-1} \sim P \\ a_1, \dots, a_{H-2} \sim \pi}} \left[ \sum_{t=0}^{H-2} \gamma^t r(s_t, a_t) + \gamma^{H-1} \max_{a_{H-1}} Q^*(s_{H-1}, a_{H-1}) \right] \end{aligned} \quad (17)$$

Therefore, we have  $a_{H-1} = \arg \max_a Q^*(s_{H-1}, a)$ . Similarly, continuing to use dynamic programming on Equation (17), we finally get:  $a_i = \arg \max_a Q^*(s_i, a)$ ,  $(i = 1, 2, \dots, H-1)$ . Thus we claim that the optimal policy is induced by the optimal value and use the symbol  $\pi_{Q^*}$  instead of  $\pi$  in Equation (16) to obtain Equation (11).

## B DERIVATION FROM EQUATION (11) TO EQUATION (12)

Given the learned dynamic predictor  $\hat{P}$ , reward predictor  $\hat{r}$ , and estimated optimal Q-value  $\hat{Q}^*$ , we first substitute these three functions for the ground-truth functions in the right side of Equation (11):

$$\max_{\pi_{\hat{Q}^*}} \mathbb{E}_{\substack{s_1, \dots, s_H \sim \hat{P} \\ a_1, \dots, a_{H-1} \sim \pi_{\hat{Q}^*}}} \left[ \sum_{t=0}^{H-1} \gamma^t \hat{r}(s_t, a_t) + \gamma^H \max_{a_H} \hat{Q}^*(s_H, a_H) \right] \quad (18)$$

However, due to the estimation error between learned functions and ground-truth functions,  $\hat{Q}^*(s_t, a_t)$  is typically not equal to  $\mathbb{E}_{s_{t+1} \sim \hat{P}} [\hat{r}(s_t, a_t) + \gamma \max_a \hat{Q}^*(s_{t+1}, a)]$ . Therefore, instead of using dynamic programming to derive that the optimal policy is the greedy policy as in Proof A, we have to use search to solve formula (18).

However, we can use the conclusion derived with ground-truth functions to make assumptions to reduce the policy space for search. We assume that the optimal policy for formula (18) maps states to actions with top-K highest Q-values. Denote the constrained policy space as  $\Pi_{\hat{Q}^*}$ , where  $\forall \pi \in \Pi_{\hat{Q}^*}, \forall s \in \mathcal{S}, \forall a \notin \text{top-K}(\hat{Q}^*(s, \cdot))$ , we have  $\pi(a|s) = 0$ . we make the following assumption: the optimal policy of formula (18) is in the constrained policy space, i.e.,  $\pi_{\hat{Q}^*}^* \in \Pi_{\hat{Q}^*}$ . Under this assumption, formula (18) is equivalent to:

$$\max_{\pi \in \Pi_{\hat{Q}^*}} \mathbb{E}_{\substack{s_1, \dots, s_H \sim \hat{P} \\ a_1, \dots, a_{H-1} \sim \pi}} \left[ \sum_{t=0}^{H-1} \gamma^t \hat{r}(s_t, a_t) + \gamma^H \max_{a_H} \hat{Q}^*(s_H, a_H) \right] \quad (19)$$

Then we maximize formula (19) over  $a_0$  to find the optimal initial action. Considering the above assumption,  $a_0 \in \text{top-K}(\hat{Q}^*(s_0, \cdot))$ . Thus we have the objective for search as:

$$\max_{\pi \in \Pi_{\hat{Q}^*}} \mathbb{E}_{\substack{s_1, \dots, s_H \sim \hat{P} \\ a_0, \dots, a_{H-1} \sim \pi}} \left[ \sum_{t=0}^{H-1} \gamma^t \hat{r}(s_t, a_t) + \gamma^H \max_{a_H} \hat{Q}^*(s_H, a_H) \right] \quad (20)$$

## C UPPER BOUND OF SEARCH-BASED Q-VALUE ESTIMATION

Let function  $f(s_0, a_0)$  be equal to formula (19) and let  $\pi_{\hat{Q}^*}^*$  be the optimal policy of formula (19). We have:

$$f(s_0, a_0) = \mathbb{E}_{\substack{s_1, \dots, s_H \sim \hat{P} \\ a_1, \dots, a_{H-1} \sim \pi_{\hat{Q}^*}^*}} \left[ \sum_{t=0}^{H-1} \gamma^t \hat{r}(s_t, a_t) + \gamma^H \max_{a_H} \hat{Q}^*(s_H, a_H) \right] \quad (21)$$

We make the following assumption similar to EfficientZero-v2 (Wang et al., 2024):

**Assumption C.1.** Assume the state transition, reward, and Q-value estimations error are upper bounded by  $\epsilon_s, \epsilon_r, \epsilon_Q$  respectively. The error bound of each estimation is formulated as:

$$\max_{n \in [N], t \in [H(n)]} \mathbb{E} [\|\hat{s}_t - s_t\|] \leq \epsilon_s \quad (22)$$

$$\max_{n \in [N], t \in [H(n)]} \mathbb{E} [\|\hat{r}(s_t) - r(s_t)\|] \leq \epsilon_r \quad (23)$$

$$\max_{n \in [N], t \in [H(n)]} \mathbb{E} [\|\hat{Q}^*(s_t) - Q^*(s_t)\|] \leq \epsilon_Q \quad (24)$$

**Theorem C.2.** Define  $s_t, a_t$  to be the states and actions resulting from current policy using ground-truth dynamics  $P$  and reward function  $r$  and similarly define  $s'_t, a'_t$  using learned functions  $\hat{P}$  and  $\hat{r}$ . Assume the learned reward function  $\hat{r}$  to be  $L_r$ -Lipschitz and the estimated optimal Q-function  $\hat{Q}^*$  to be  $L_Q$ -Lipschitz. Assume the estimation errors of learned functions are bounded as in Assumption C.1. Then we have the error between search-based Q-value estimation  $f(s_0, a_0)$  and ground-truth Q-value  $Q^*(s_0, a_0)$  bounded as:

$$\|f(s_0, a_0) - Q^*(s_0, a_0)\| \leq \frac{1 - \gamma^H}{1 - \gamma} \epsilon_r + \left( \frac{\gamma - \gamma^H}{1 - \gamma} \epsilon_r + \gamma^H \epsilon_Q \right) \epsilon_s + \gamma^H \epsilon_Q \quad (25)$$

*Proof.*

$$\begin{aligned}
& \|f(s_0, a_0) - Q^*(s_0, a_0)\| \\
&= \left\| \mathbb{E} \left[ \sum_{t=0}^{H-1} \gamma^t \hat{r}(s'_t, a'_t) + \gamma^H \max_{a_H} \hat{Q}^*(s'_H, a_H) \right] - \mathbb{E} \left[ \sum_{t=0}^{H-1} \gamma^t r(s_t, a_t) + \gamma^H \max_{a_H} Q^*(s_H, a_H) \right] \right\| \\
&\leq \mathbb{E} \left[ \left\| \hat{r}(s_0, a_0) - r(s_0, a_0) + \sum_{t=1}^{H-1} \gamma^t \left( \hat{r}(s'_t, a'_t) - r(s_t, a_t) \right) + \gamma^H \left( \max_{a_H} \hat{Q}^*(s'_H, a_H) - \max_{a_H} Q^*(s_H, a_H) \right) \right\| \right] \\
&\leq \mathbb{E} [\|\hat{r}(s_0, a_0) - r(s_0, a_0)\|] + \sum_{t=1}^{H-1} \gamma^t \mathbb{E} [\|\hat{r}(s'_t, a'_t) - r(s_t, a_t)\|] + \gamma^H \mathbb{E} \left[ \left\| \max_{a_H} \hat{Q}^*(s'_H, a_H) - \max_{a_H} Q^*(s_H, a_H) \right\| \right] \\
&\leq \epsilon_r + \sum_{t=1}^{H-1} \gamma^t \mathbb{E} [\|\hat{r}(s'_t, a'_t) - r(s_t, a_t)\|] + \gamma^H \mathbb{E} \left[ \left\| \max_{a_H} \hat{Q}^*(s'_H, a_H) - \max_{a_H} Q^*(s_H, a_H) \right\| \right] \quad (26)
\end{aligned}$$

For the second term in inequality (26):

$$\begin{aligned}
& \mathbb{E} [\|\hat{r}(s'_t, a'_t) - r(s_t, a_t)\|] \\
&= \mathbb{E} [\|\hat{r}(s'_t, a'_t) - \hat{r}(s_t, a_t) + \hat{r}(s_t, a_t) - r(s_t, a_t)\|] \\
&\leq \mathbb{E} [\|\hat{r}(s'_t, a'_t) - \hat{r}(s_t, a_t)\|] + \mathbb{E} [\|\hat{r}(s_t, a_t) - r(s_t, a_t)\|] \\
&\leq L_r \|s'_t - s_t\| + \epsilon_r \\
&\leq L_r \epsilon_s + \epsilon_r \quad (27)
\end{aligned}$$

For the third term in inequality (26), let  $a_H^1 = \arg \max_{a_H} \hat{Q}^*(s'_H, a_H)$  and  $a_H^2 = \arg \max_{a_H} Q^*(s_H, a_H)$ . Then we have:

$$\begin{aligned}
& \mathbb{E} \left[ \left\| \max_{a_H} \hat{Q}^*(s'_H, a_H) - \max_{a_H} Q^*(s_H, a_H) \right\| \right] \\
&= \mathbb{E} \left[ \left\| \hat{Q}^*(s'_H, a_H^1) - Q^*(s_H, a_H^2) \right\| \right] \\
&= \mathbb{E} \left[ \left\| \hat{Q}^*(s'_H, a_H^1) - \hat{Q}^*(s_H, a_H^2) + \hat{Q}^*(s_H, a_H^2) - Q^*(s_H, a_H^2) \right\| \right] \\
&\leq \mathbb{E} \left[ \left\| \hat{Q}^*(s'_H, a_H^1) - \hat{Q}^*(s_H, a_H^2) \right\| \right] + \mathbb{E} \left[ \left\| \hat{Q}^*(s_H, a_H^2) - Q^*(s_H, a_H^2) \right\| \right] \\
&\leq L_Q \|s'_H - s_H\| + \epsilon_Q \\
&\leq L_Q \epsilon_s + \epsilon_Q \quad (28)
\end{aligned}$$

Substitute inequalities (27) and (28) into (26):

$$\begin{aligned}
& \|f(s_0, a_0) - Q^*(s_0, a_0)\| \\
&\leq \epsilon_r + \sum_{t=1}^{H-1} \gamma^t (L_r \epsilon_s + \epsilon_r) + \gamma^H (L_Q \epsilon_s + \epsilon_Q) \\
&= \frac{1 - \gamma^H}{1 - \gamma} \epsilon_r + \left( \frac{\gamma - \gamma^H}{1 - \gamma} \epsilon_r + \gamma^H \epsilon_Q \right) \epsilon_s + \gamma^H \epsilon_Q \quad (29)
\end{aligned}$$

□

**Analysis.** We expect the search-based Q-values  $f(s, a)$  have an upper error bound no greater than the estimated Q-values  $\hat{Q}^*(s, a)$ , which is formulated as the following inequality:

$$\frac{1 - \gamma^H}{1 - \gamma} \epsilon_r + \left( \frac{\gamma - \gamma^H}{1 - \gamma} \epsilon_r + \gamma^H \epsilon_Q \right) \epsilon_s + \gamma^H \epsilon_Q \leq \epsilon_Q \quad (30)$$

Based on inequality (30), we derive the following condition:

$$\frac{1}{1-\gamma}\epsilon_r + \left( \frac{\gamma}{1-\gamma} \frac{L_r - \gamma^H L_Q}{1-\gamma^H} - \frac{L_r - L_Q}{1-\gamma} \frac{\gamma^H}{1-\gamma^H} \right) \epsilon_s \leq \epsilon_Q \quad (31)$$

The inequality (31) means that if the weighted sum of rewards estimation error  $\epsilon_r$  and state transition estimation error  $\epsilon_s$  are less than or equal to the Q-values estimation error  $\epsilon_Q$ , then the search-based optimal Q-values have a lower upper-bound of error than estimated optimal Q-values.

## D GAMES

We select 20 Atari games maintaining the difficulty distribution of full Atari 2600 games defined by Gulcehre et al. (2020), which includes 9 easy games, 9 medium games, and 2 hard games. We use 15 out of 20 games for training and the remaining 5 for OOD generalization experiments. The 15 training games are: Phoenix, Centipede, SpaceInvaders, Carnival, NameThisGame, Assault, Atlantis, DemonAttack, BeamRider, ChopperCommand, Seaquest, TimePilot, StarGunner, Berzerk, Zaxxon. The 5 held-out games are: Pong, Robotank, YarsRevenge, Gravitar, MsPacman. Details about the size of action spaces and game difficulties are shown in Table 6.

Table 6: Atari Games: Name, Game difficulty, Action Space, and Type.

Game	Difficulty	Action Space	Type
Assault	Medium	7	Train
Atlantis	Hard	18	Train
BeamRider	Medium	9	Train
Berzerk	Hard	18	Train
Carnival	Medium	6	Train
Centipede	Medium	18	Train
ChopperCommand	Easy	18	Train
DemonAttack	Easy	6	Train
Gravitar	Easy	18	Fine-tune
MsPacman	Medium	9	Fine-tune
NameThisGame	Easy	6	Train
Phoenix	Easy	8	Train
Pong	Medium	6	Fine-tune
Robotank	Medium	18	Fine-tune
Seaquest	Easy	18	Train
SpaceInvaders	Easy	6	Train
StarGunner	Medium	18	Train
TimePilot	Easy	10	Train
YarsRevenge	Medium	18	Fine-tune
Zaxxon	Easy	18	Train

## E EXPERIMENTAL DETAILS

### E.1 IMPLEMENT DETAILS

#### E.1.1 JOWA

We implement JOWA based on the codes of IRIS (Micheli et al., 2022)<sup>1</sup>. We train the tokenizer VQ-VAE using the following loss function:

$$\mathcal{L}(E, D, \mathcal{E}) = \|x - D(z)\|_1 + \|\text{sg}(E(x)) - \mathcal{E}(z)\|_2^2 + \|\text{sg}(\mathcal{E}(z)) - E(x)\|_2^2 + \mathcal{L}_{\text{perceptual}}(x, D(z))$$

where  $E, D, \mathcal{E}$  are encoder, decoder, and embedding table respectively.  $\text{sg}(\cdot)$  is the stop-gradient operator. The last term is the perceptual loss (Johnson et al., 2016). We list the hyperparameters of VQ-VAE in Table 7 and 8. After the first stage of pretraining, the VQ-VAE is frozen.

<sup>1</sup><https://github.com/eloialonso/iris>

Table 7: Encoder / Decoder hyperparameters. We list the hyperparameters for the encoder, the same ones apply for the decoder.

Hyperparameter	Value
Frame dimensions (h, w)	$84 \times 84$
Layers	3
Residual blocks per layer	2
Channels in convolutions	64
Self-attention layers at resolution	6 / 12

Table 8: Embedding table hyperparameters.

Hyperparameter	Value
Vocabulary size	2048
Tokens per frame (K)	36
Token embedding dimension	512

In addition to the vocabulary embedding and position embedding, we add a learnable task embedding for observation tokens and action tokens respectively. Our transformer are based on minGPT<sup>2</sup> with FlashAttention (Dao et al., 2022) for acceleration. The hyperparameters of JOWA’s transformer backbone are listed in Table 9 and 10.

Table 9: Same hyperparameters of transformer for 3 JOWA variants.

Hyperparameter	Value
max sequence tokens	296
dropout rate	0.1

Table 10: Different hyperparameters of transformer for 3 JOWA variants.

Model	Layers	Hidden size	Heads
JOWA-40M	4	512	8
JOWA-70M	6	768	12
JOWA-150M	12	768	12

Table 11: Hyperparameters of Q-heads for 3 JOWA variants.

Model	Layers	MLP Hidden dimension	Number of heads	Dropout
JOWA-40M	3	768	1	0.01
JOWA-70M	3	1024	1	0.01
JOWA-150M	3	1792	3	0.01

The observation predictor, reward predictor, and terminal predictor are 2-layers MLP. The Q-heads are MLP with dropout, layer normalization, and Mish activations from Hansen et al. (2024). The hyperparameters of Q-heads for JOWA are shown in Table 11. The training hyperparameters of JOWA are shown in Table 12.

### E.1.2 MTBC

We implement MTBC based on JOWA. We remove the observation predictor, reward predictor, and terminal predictor. We change the output dimension of Q-heads to 18 and train the heads as a 18-class classification problem. All hyperparameters are kept the same as JOWA.

### E.1.3 EDT

We use the official code for EDT<sup>3</sup>. We implement EDT-200M based on the architecture configuration of MGDT-200M, which is shown in Table 13. We change the batch size to 512 and keep other hyperparameters the same as its original configuration. We enable data augmentation (random cropping and random rotation) for EDT.

### E.1.4 MGDT

We implement MGDT based on the codes of EDT. We use  $\{o_{t+i}, R_{t+i}, a_{t+i}, r_{t+i}\}_{i=0}^3$  as the input sequences, remove the expectile regression loss  $\mathcal{L}_{\max}$  and observation prediction loss  $\mathcal{L}_{\text{observation}}$ , and

<sup>2</sup><https://github.com/karpathy/minGPT>

<sup>3</sup><https://github.com/kristery/Elastic-DT>

Table 12: Training hyperparameters of JOWA.

Hyperparameter	Value
Optimizer (VQ-VAE)	Adam
Optimizer (except VQ-VAE)	AdamW
Learning rate (VQ-VAE)	0.0001
Learning rate (except VQ-VAE, stage 1)	0.0001
Learning rate (except VQ-VAE, stage 2)	0.00005
Batch size (VQ-VAE)	2048
Batch size (except VQ-VAE)	512
Weight decay (except VQ-VAE)	0.01
Gradient clip	1.0
Discount factor ( $\gamma$ )	0.99
Target Q update frequency	1000
Distributional Q	$[-10, 30]$
Number of atoms	51
Coefficient of CQL ( $\alpha$ )	0.1
Coefficient of $\mathcal{L}_{\text{world}}$ ( $\beta$ )	0.1

Table 13: Hyperparameters of transformer for 2 MGDT variants.

Model	Layers	Hidden size	Heads
MGDT-40M	6	768	12
MGDT-200M	10	1280	20

add the reward prediction loss to rewrite the codes of EDT into MGDT. We enable data augmentation (random cropping and random rotation) for MGDT. The hyperparameters of transformer for two MGDT variants are listed in Table 13.

### E.1.5 SCALED-QL

We implement a pytorch version of Scaled-QL from scratch, referring to the jax version of its official preliminary codes<sup>4</sup>. We use ResNet-101 as the representation backbone, followed by 3-layers MLP with 1024 hidden neurons and an output layer. We replace the batch normalization in ResNet with group normalization and use a learnable spatial embeddings to aggregate the outputs of the ResNet instead of global mean pooling. Before the output layer, we normalize the feature  $e$  as  $\frac{e}{\|e\|^2}$ . The training hyperparameters of Scaled-QL are listed in Table 14.

For fair comparison, all methods are trained with the same batch size of 512 for 1.75M gradient steps. For reporting results, we report the performance of the agent at the end of pretraining.

## E.2 FINE-TUNING PROTOCOL

We uniformly draw 5k transitions from expert-level DQN-Replay (Agarwal et al., 2020) (last 20% of the original dataset) for each held-out game. Each game was fine-tuned separately to measure the model’s transfer performance for a fixed game. we fine-tuned all methods using a batch size of 32 and learning rate of 0.00005 for 10k gradient steps. For reporting results, we report the performance of the agent snapshot that obtain the highest score during fine-tuning.

For JOWA in the second fine-tuning stage, we set both the planning horizon and the beam width to 2 for all fine-tuning experiments. Thus we sample batch of 6-steps segments, using planning algorithm to synthesis the last 2 steps. Then we update JOWA with 3/4 batch (24) of real data and 1/4 batch (8) of synthetic data using COMBO loss rather than CQL loss as the  $\mathcal{L}_{\text{action}}$ . For other baselines, we enable random cropping and random rotation for data augmentation.

<sup>4</sup><https://tinyurl.com/scaled-ql-code>

Table 14: Training hyperparameters of Scaled-QL.

Hyperparameter	Value
Optimizer	Adam
Learning rate	0.0002
Batch size	512
Gradient clip	1.0
Discount factor ( $\gamma$ )	0.99
Target Q update frequency	2000
Distributional Q	$[-20, 20]$
Number of atoms	51
Coefficient of CQL ( $\alpha$ )	0.05
n-step returns	3

Table 15: Evaluation settings of Atari.

Hyperparameter	Value
Sticky actions	No
Grey-scaling	True
Observation down-sampling	(84, 84)
Frames stacked	4
Frame skip (Action repetitions)	4
Terminal condition	Game Over
Max frames per episode	108K
Evaluation noise $\epsilon_{\text{eval}}$	0.001

### E.3 EVALUATION PROTOCOL

For all methods, each game score is calculated by averaging over 16 model rollout episode trials. To reduce inter-trial variability, we do not use sticky actions during evaluation following Lee et al. (2022); Kumar et al. (2023). Following standard protocols on Atari, we evaluate a noised version of the policy with an epsilon-greedy scheme, with  $\epsilon_{\text{eval}} = 0.001$ . The evaluation settings of Atari are shown in Table 15.

For the expert action inference of MGD and EDT, we set the inverse temperature  $\kappa$  to 10. For the planning of JOWA, we set the planning horizon  $H$  to 2 for all games. The beam width are set according to the size of valid action space of each game. Specifically, we set beam width  $K$  to 2 if the valid action space size is less than 10, otherwise we set  $K$  in  $\{3, 4\}$ . The planning hyperparameters for each game are shown in Table 16.

Table 16: Planning hyperparameters of JOWA during evaluation.

Game	planning horizon	beam width	Action space
Assault	2	2	7
Atlantis	2	3	18
BeamRider	2	2	9
Berzerk	2	4	18
Carnival	2	2	6
Centipede	2	4	18
ChopperCommand	2	4	18
DemonAttack	2	2	6
NameThisGame	2	2	6
Phoenix	2	2	8
Seaquest	2	3	18
SpaceInvaders	2	2	6
StarGunner	2	3	18
TimePilot	2	3	10
Zaxxon	2	3	18

## F RAW SCORES

We summarize the raw scores of fine-tuning experiments and ablation studies in Table 17 and 18 respectively.

Table 17: Offline fine-tuning performance on unseen games using 5k transitions, measured in terms of DQN-normalized score, following Lee et al. (2022); Kumar et al. (2023).

Game	Random	DQN	MTBC 120M	MGDT 200M	EDT 200M	SQL 80M	JOWA 150M	JOWA-150M (scratch)
Gravitar	173.0	473.0	35.7	250.0	253.3	137.5	273.3	83.3
MsPacman	307.3	3085.6	905.0	1290.3	1210.7	1040.2	2016.7	786.7
Pong	-20.7	19.5	5.8	9.7	11.3	13.7	17.7	8.8
Robotank	2.2	63.9	6.8	16.0	15.5	19.7	25.0	11.0
YarsRevenge	3092.9	18089.9	7987.5	10886.3	11276.9	10838.5	17506.2	6507.0
Mean	0.000	1.000	0.164	0.422	0.430	0.360	0.647	0.196
Median	0.000	1.000	0.215	0.354	0.325	0.284	0.615	0.173
IQM	0.000	1.000	0.205	0.377	0.380	0.355	0.647	0.181

Table 18: Raw scores on the 6 Atari games for various training choices. The mean, median, and IQM human-normalized score are shown in the last 3 rows and the best scores are marked in bold.

Game	Origin	Different training losses				Q-heads ensemble		No task embedding	Synthetic data in pretraining
		No CQL	No $\mathcal{L}_{\text{world}}$	sg( $\mathcal{L}_{\text{action}}$ )	MSE	Equal	Random		
Assault	1423.5	857.9	1650	452.7	637.6	1628.8	1258.6	1428.5	655.7
Carnival	5560	4144.6	5560	2120	620	5154.3	4160	5974.2	3492.9
Centipede	5018.9	1494.1	8146	5568.7	4097.3	5592.5	6450	3725.4	3253
NameThisGame	12208.1	9307.1	4407.3	2108.8	1315	12148.6	9420	7700	5080
Phoenix	4740	140	2036.7	1920	1190	4610	4941.7	4020	193.3
SpaceInvaders	1201.7	605	323.4	539.3	260	958.4	1283.3	575.4	786.3
Mean HNS	1.183	0.613	0.917	0.293	0.189	<b>1.209</b>	1.026	0.964	0.448
Median HNS	<b>1.078</b>	0.696	0.489	0.304	0.118	0.975	0.921	0.721	0.452
IQM HNS	<b>1.123</b>	0.637	0.659	0.307	0.126	1.049	0.931	0.824	0.464