SMIX(λ): Enhancing Centralized Value Functions for Cooperative Multi-Agent Reinforcement Learning (Supplementary)

Chao Wen, Xinghu Yao,Yuhui Wang, Xiaoyang Tan

A. Proof of Theorem 1

Theorem 1. *For QMIX, if $\frac{\partial Q_{tot}}{\partial Q^i} \geq 0$ for $i \in \{1, 2, \cdots, N\}$, then we have*

$$\max_{\mathbf{a}} Q_{tot}(\boldsymbol{\tau}, \mathbf{a}) = Q_{tot}\left(\boldsymbol{\tau}, \text{argmax}_{a^1} Q^1(\tau^1, a^1), \cdots, \text{argmax}_{a^N} Q^N(\tau^N, a^N)\right) \tag{1}$$

Proof. For QMIX, we have

$$Q_{tot}(\boldsymbol{\tau}, \mathbf{a}) := \hat{Q}\left(Q_1(\tau^1, a^1), \ldots, Q^N(\tau^N, a^N)\right),$$

where $\hat{Q}$ is the *mixing network* and $\mathbf{a} = (a_1, \ldots, a_N)$. Similarly, we have $\frac{\partial \hat{Q}}{\partial Q^i} \geq 0$ and

$$Q_{tot}\left(\boldsymbol{\tau}, \text{argmax}_{a^1} Q^1(\tau^1, a^1), \cdots, \text{argmax}_{a^N} Q^N(\tau^N, a^N)\right) := \hat{Q}\left(\max_{a^1} Q^1(\tau^1, a^1), \ldots, \max_{a^n} Q^N(\tau^N, a^N)\right).$$

Since $\frac{\partial \hat{Q}}{\partial Q^i} \geq 0$, given $(\bar{a}^2, \ldots, \bar{a}^n)$, we have

$$\hat{Q}\left(Q^1(\tau^1, a^1), Q^2(\tau^2, \bar{a}^2), \ldots, Q^N(\tau^N, \bar{a}^N)\right) \leq \hat{Q}\left(\max_{a^1} Q^1(\tau^1, a^1), Q^2(\tau^2, \bar{a}^2), \ldots, Q^N(\tau^N, \bar{a}^N)\right) \text{ for any } a^1.$$

Similarly, given $(\bar{a}^1, \ldots, \bar{a}^{k-1}, \bar{a}^{k+1}, \ldots, \bar{a}^N)$, we have

$$\hat{Q}\left(Q_1(\tau^1, \bar{a}^1), \ldots, Q^k(\tau^k, a^k), \ldots, Q^N(\tau^N, \bar{a}^N)\right) \leq \hat{Q}\left(Q^1(\tau^1, \bar{a}^1), \ldots, \max_{a^k} Q^k(\tau^k, a^k), \ldots, Q^N(\tau^N, \bar{a}^N)\right) \text{ for any } a^k.$$

Finally, for any $(a^1, \ldots, a^N)$, we have

$$\begin{aligned} & \hat{Q}\left(Q^1(\tau^1, a^1), \ldots, Q^N(\tau^N, a^N)\right) \\ \leq\ & \hat{Q}\left(\max_{a^1} Q_1(\tau^1, a^1), \max_{a^2} Q^2(\tau^2, a^2), Q^3(\tau^3, a^3), \ldots, Q^N(\tau^N, a^N)\right) \\ \leq\ & \hat{Q}\left(\max_{a^1} Q^1(\tau^1, a^1), \max_{a^2} Q^2(\tau^2, a^2), \ldots, \max_{a^N} Q^N(\tau^N, a^N)\right). \end{aligned}$$

Therefore, we obtain

$$\max_{a^1, \ldots, a^N} \hat{Q}\left(Q^1(\tau^1, a^1), \ldots, Q^N(\tau^n, a^n)\right) = Q\left(\max_{a^1} Q^1(\tau^1, a^1), \max_{a^2} Q^2(\tau^2, a^2), \ldots, \max_{a^N} Q_n(\tau^N, a^N)\right),$$

which is the specific form of (1) for QMIX. □

B. Proof of Theorem 2

Theorem 2. *Suppose we update the value function from $Q_n^{SMIX}(\boldsymbol{\tau}_t, \mathbf{a}_t) = Q_n^{Q(\lambda)}(\boldsymbol{\tau}_t, \mathbf{a}_t)$, where n represents the n-th update. Let $\epsilon = \max_{\boldsymbol{\tau}} \|\boldsymbol{\pi}(\cdot|\boldsymbol{\tau}) - \boldsymbol{\mu}(\cdot|\boldsymbol{\tau})\|_1$, $M = \max_{\boldsymbol{\tau}, \mathbf{a}} |Q_n^{Q(\lambda)}(\boldsymbol{\tau}, \mathbf{a})|$. Then, the error between $Q_{n+1}^{SMIX}(\boldsymbol{\tau}_t, \mathbf{a}_t)$ and $Q_{n+1}^{Q(\lambda)}(\boldsymbol{\tau}_t, \mathbf{a}_t)$ can be bounded by the expression:*

$$|Q_{n+1}^{SMIX}(\boldsymbol{\tau}_t, \mathbf{a}_t) - Q_{n+1}^{Q(\lambda)}(\boldsymbol{\tau}_t, \mathbf{a}_t)| \leq \frac{\epsilon\gamma}{1 - \lambda\gamma} M. \tag{2}$$

Proof. First, we have,

$$\begin{aligned} |\delta_t^{\boldsymbol{\pi}} - \hat{\delta}_t^{\boldsymbol{\pi}}| &= |\gamma \mathbb{E}_{\boldsymbol{\mu}} Q_n^{\text{SMIX}}(\boldsymbol{\tau}_{t+1}, \cdot) - \gamma \mathbb{E}_{\boldsymbol{\pi}} Q_n^{Q(\lambda)}(\boldsymbol{\tau}_{t+1}, \cdot)| \\ &= \gamma |\sum_{\mathbf{a}} \boldsymbol{\mu}(\mathbf{a}|\boldsymbol{\tau}_{t+1}) Q_n^{\text{SMIX}}(\boldsymbol{\tau}_{t+1}, \cdot) - \sum_{\mathbf{a}} \boldsymbol{\pi}(\mathbf{a}|\boldsymbol{\tau}_{t+1}) Q_n^{Q(\lambda)}(\boldsymbol{\tau}_{t+1}, \cdot)| \leq \gamma\epsilon M. \end{aligned}$$

Thus,

$$
\begin{aligned}
|Q_{n+1}^{\text{SMIX}}\left(\boldsymbol{\tau}_t, \mathbf{a}_t\right) - Q_{n+1}^{Q(\lambda)}\left(\boldsymbol{\tau}_t, \mathbf{a}_t\right)| &= \left| \mathbb{E}_{\boldsymbol{\mu}} \left[\sum_{k=t}^{\infty} \left(\prod_{i=t+1}^{k} \lambda\gamma \right) \delta_k^{\boldsymbol{\pi}} \right] - \mathbb{E}_{\boldsymbol{\mu}} \left[\sum_{k=t}^{\infty} \left(\prod_{i=t+1}^{k} \lambda\gamma \right) \hat{\delta}_k^{\boldsymbol{\pi}} \right] \right| \\
&= \left| \mathbb{E}_{\boldsymbol{\mu}} \left[\sum_{k=t}^{\infty} \left(\prod_{i=t+1}^{k} \lambda\gamma \right) \left(\delta_k^{\boldsymbol{\pi}} - \hat{\delta}_k^{\boldsymbol{\pi}} \right) \right] \right| \leq \left| \mathbb{E}_{\boldsymbol{\mu}} \left[\sum_{k=t}^{\infty} \left(\prod_{i=t+1}^{k} \lambda\gamma \right) (\gamma\epsilon M) \right] \right| \\
&\leq \mathbb{E}_{\boldsymbol{\mu}} \left[\frac{1}{1-\lambda\gamma} (\gamma\epsilon M) \right] = \frac{\epsilon\gamma}{1-\lambda\gamma} M.
\end{aligned}
$$

Therefore, the expression (2) holds. □

C. Algorithm

Algorithm 1 Training Procedure for SMIX(λ)

Initialize the behavior network with parameters θ, the target network with parameters θ^-, empty replay buffer $\mathcal{D}$ to capacity $N_{\mathcal{D}}$, training batch size b
for each training episode **do**
  for each episode **do**
    for $t = 1$ to $T - 1$ **do**
      Obtain the partial observation $\mathbf{o}_t = \{o_t^1, \cdots, o_t^N\}$ for all agents and global state s_t
      Select action a_t^i according to ϵ-greedy policy w.r.t agent i's decentralized value function Q^i for $i = 1, \cdots, N$
      Execute joint action $\mathbf{a}_t = \{a^1, a^2, \cdots, a^N\}$ in the environment
      Obtain the global reward r_{t+1}, the next partial observation o_{t+1}^i for each agent i and next global state s_{t+1}
    end for
    Store the episode in $\mathcal{D}$, replacing the oldest episode if $|\mathcal{D}| \geq N_{\mathcal{D}}$
  end for
  Sample a batch of b episodes $\sim$ Uniform($\mathcal{D}$)
  Calculate λ-return targets $y_i^{tot} = (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$, where $G_t^{(n)} = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^n \mathbb{E}_{\boldsymbol{\pi}} Q\left(\boldsymbol{\tau}_{t+n}, \mathbf{a}_{t+n}; \theta^-\right)$
  Update θ by minimizing $\sum_{t=1}^{T-1} \sum_{i=1}^{b} \left[(y_i^{tot} - Q_{tot}^{\boldsymbol{\pi}}(\boldsymbol{\tau}, \mathbf{a}; \theta))^2 \right]$
  Replace target parameters $\theta^- \leftarrow \theta$ every C episodes
end for

D. Implementation Details

The agent network consists of a 64-dimensional GRU (Chung et al. 2014). One 64-dimensional fully connected layer with ReLU activation function before GRU is applied for processing the input. The layer after GRU is a fully connected layer of 64 units, which outputs the decentralized state-action values $Q^i(\tau^i, \cdot)$ of agent i. All agent networks share parameters for reducing the number of parameters to be learned. Thus the agent's one-hot index i is concatenated onto each agent's observations. Agent's previous action is also concatenated to the input. The target network is updated after every 200 training episodes. We use RMSprop optimizer with learning rate 0.0005 and $\alpha = 0.99$ without weight decay or momentum during training. We perform independent ϵ-greedy for exploration. ϵ is annealed linearly from 1.0 to 0.05 across the first 50k timesteps for all experiments. The discount factor is set to $\gamma = 0.99$. The architecture of the mixing network is the same as in Rashid et al. (2018). Actually, all the settings of SMIX(λ)'s parameters are kept the same as those of QMIX for fair comparison, except that SMIX(λ) has an extra parameter λ and uses smaller buffer size.

E. Additional Results

We provide quantitative comparisons of our methods and their counterparts in Table 1.

Algorithms	3m		8m		2s3z		3s5z		2s_vs_1sc		3s_vs_3z	
	mean ± *std*	median	mean ± *std*	median	mean ± *std*	median	mean ± *std*	median	mean ± *std*	median	mean ± *std*	median
SMIX(λ)	**99** ($\pm$0)	**99**	**91** ($\pm$3)	**90**	**90** ($\pm$4)	**91**	**61** ($\pm$11)	**62**	**94** ($\pm$5)	**96**	**84** ($\pm$14)	**88**
QMIX	95 ($\pm$3)	95	90 ($\pm$3)	89	81 ($\pm$7)	81	16 ($\pm$12)	11	39 ($\pm$19)	45	15 ($\pm$20)	9
SMIX(λ)-COMA	**93** ($\pm$8)	**97**	**92** ($\pm$2)	**93**	**44** ($\pm$18)	**47**	**0** ($\pm$0)	**0**	**97** ($\pm$4)	**100**	**0** ($\pm$0)	**0**
COMA	92 ($\pm$2)	93	90 ($\pm$2)	91	24 ($\pm$6)	24	**0** ($\pm$0)	**0**	77 ($\pm$11)	78	**0** ($\pm$0)	**0**
SMIX(λ)-VDN	**98** ($\pm$0)	**98**	**94** ($\pm$3)	**93**	**78** ($\pm$14)	**79**	**29** ($\pm$12)	**26**	**96** ($\pm$2)	**97**	**67** ($\pm$25)	**83**
VDN	95 ($\pm$2)	95	86 ($\pm$5)	87	64 ($\pm$16)	71	1 ($\pm$2)	0	86 ($\pm$8)	88	27 ($\pm$9)	27
SMIX(λ)-IQL	**91** ($\pm$4)	**94**	**80** ($\pm$5)	**79**	**32** ($\pm$8)	**31**	**0** ($\pm$0)	**0**	**92** ($\pm$6)	**94**	**35** ($\pm$21)	**31**
IQL	83 ($\pm$9)	86	59 ($\pm$15)	58	14 ($\pm$10)	13	**0** ($\pm$0)	**0**	51 ($\pm$22)	54	5 ($\pm$4)	6

Table 1: Mean, standard deviation, and median of test win rate percentages after training for 1 million timesteps in six different scenarios. The highest results are in bold. Our methods (SMIX(λ), SMIX(λ)-COMA, SMIX(λ)-VDN, SMIX(λ)-IQL) perform on par or significantly better than their counterparts in terms of the learning speed across all scenarios.

References

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *Advances in Neural Information Processing Systems*.

Rashid, T.; Samvelyan, M.; Witt, C. S.; Farquhar, G.; Foerster, J.; and Whiteson, S. 2018. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, 4292–4301.