Off-Policy Reinforcement Learning via Online Policy Mirror Descent (OPMD)

Author

April 17, 2025

We present some findings about off-policy reinforcement learning (with a focus on the bandit setting) through the lens of online policy mirror descent (OPMD). At the end of this technical report, we arrive at the surprising conclusion that the standard policy gradient, weighted by a coefficient and using the group mean reward as the baseline, can be a feasible direction for updating the policy even in off-policy settings (while the standard theory of policy gradient only holds for on-policy settings). This has been validated empirically in our exploratory experiments during the development of Trinity-RFT.

1 OPMD: Kimi's version

This section is a recap of the OPMD variant proposed in the technical report of Kimi k1.5 [2].

Analysis. For a specific task/query x and a reference policy π_{ref}, consider the following objective for training policy $\pi_{\boldsymbol{\theta}}$ at a particular iteration of the RL process:

$$\max_{\boldsymbol{\theta}} \quad J(\boldsymbol{\theta}; x, \pi_{\mathsf{ref}}) \coloneqq \mathbb{E}_{y \sim \pi_{\boldsymbol{\theta}}(\cdot|x)}[r(x,y)] - \tau \cdot D_{\mathsf{KL}}\big(\pi_{\boldsymbol{\theta}}(\cdot|x) \| \pi_{\mathsf{ref}}(\cdot|x)\big).$$

Note that π_{ref} can be changing during the RL process. In [2], π_{ref} is set to $\pi_{\boldsymbol{\theta}_t}$ at the t-th iteration, i.e., when updating the policy from $\boldsymbol{\theta}_t$ to $\boldsymbol{\theta}_{t+1}$.

The optimal policy $\pi^\star$ for this objective satisfies the following: for any response y,

$$\pi^\star(y|x) = \frac{\pi_{\mathsf{ref}}(y|x) e^{r(x,y)/\tau}}{Z} \propto \pi_{\mathsf{ref}}(y|x) e^{r(x,y)/\tau}, \tag{1}$$

$$\text{where} \quad Z \coloneqq Z(x, \pi_{\mathsf{ref}}) = \int \pi_{\mathsf{ref}}(y'|x) e^{r(x,y')/\tau} \, \mathrm{d}y' = \mathbb{E}_{y' \sim \pi_{\mathsf{ref}}(\cdot|x)}[e^{r(x,y')/\tau}]. \tag{2}$$

Taking logarithm of both sides of Eq. (1), we see that the optimal policy $\pi^\star$ must satisfy the following consistency condition:

$$r(x,y) - \tau \cdot \log Z - \tau \cdot \big(\log \pi^\star(y|x) - \log \pi_{\mathsf{ref}}(y|x)\big) = 0.$$

Algorithm. Based on the above analysis, [2] proposes the following OPMD variant. For a query x, first sample K rollouts $y_1, \ldots, y_K \sim \pi_{\mathsf{ref}}(\cdot|x)$ from the reference policy, then define a surrogate loss as follows:

$$\widehat{J}(\boldsymbol{\theta}; x, \pi_{\mathsf{ref}}) \coloneqq \sum_{i \in [K]} \Big(r(x, y_i) - \tau \cdot \log \widehat{Z} - \tau \cdot \big(\log \pi_{\boldsymbol{\theta}}(y_i|x) - \log \pi_{\mathsf{ref}}(y_i|x) \big) \Big)^2,$$

$$\text{where} \quad \tau \cdot \log \widehat{Z} \coloneqq \tau \cdot \log \Big(\frac{1}{K} \sum_{i \in [K]} e^{r(x,y_i)/\tau} \Big).$$

Although this is an off-policy method (since the rollout policy π_{ref} is different from the policy $\pi_{\boldsymbol{\theta}}$ being updated), it is still limited because the rollouts have to be sampled from the particular policy $\pi_{\mathsf{ref}} = \pi_{\boldsymbol{\theta}_t}$ for the t-th iteration of the RL process, as mentioned earlier. The reason for this limitation is the need of estimating $Z = Z(x, \pi_{\mathsf{ref}})$ using samples from $\pi_{\mathsf{ref}}(\cdot|x)$.

2 Pairwise OPMD

Analysis. To eliminate the Z term, we note that Eq. (1) is equivalent to the following:

$$\forall\ y_1 \text{ and } y_2, \quad \frac{\pi^\star(y_1|x)}{\pi^\star(y_2|x)} = \frac{\pi_{\mathsf{ref}}(y_1|x)}{\pi_{\mathsf{ref}}(y_2|x)} e^{\big(r(x,y_1)-r(x,y_2)\big)/\tau}.$$

Taking logarithm of both sides, we have

$$\log \pi^\star(y_1|x) - \log \pi^\star(y_2|x) = \log \pi_{\mathsf{ref}}(y_1|x) - \log \pi_{\mathsf{ref}}(y_2|x) + \frac{r(x,y_1)-r(x,y_2)}{\tau},$$

or equivalently,

$$r(x,y_1) - \tau \cdot \big(\log \pi^\star(y_1|x) - \log \pi_{\mathsf{ref}}(y_1|x)\big) = r(x,y_2) - \tau \cdot \big(\log \pi^\star(y_2|x) - \log \pi_{\mathsf{ref}}(y_2|x)\big).$$

Note that this holds true for a pair of arbitrary responses y_1 and y_2.

Algorithm. For a query x and K *arbitrary* responses $y_1, \dots, y_K$, we define the following surrogate loss:

$$\begin{aligned} &\widehat{J}(\boldsymbol{\theta}; x, \pi_{\mathsf{ref}}) \coloneqq \sum_{1 \le i < j \le K} \big(a_i - a_j\big)^2, \\ &\text{where} \quad a_i \coloneqq r(x,y_i) - \tau \cdot \big(\log \pi_{\boldsymbol{\theta}}(y_i|x) - \log \pi_{\mathsf{ref}}(y_i|x)\big), \quad i \in [K]. \end{aligned}$$

Here π_{ref} can be any reference policy for KL regularization, regardless of how $y_1, \dots, y_K$ were sampled. While this is a fully off-policy RL method, it has its own limitation: to run this algorithm, we should make sure that multiple (at least 2) rollouts for the same task are included within one micro-batch (whose size is typically much smaller that of a batch or mini-batch), which adds to infrastructure complexity.

Remark 1. In the special case of $K = 2$, the above method, termed as "pairwise OPMD", turns out to be the same as "contrastive policy gradient" proposed in [1], albeit with a simpler and more intuitive derivation.

3 OPMD: an embarrassingly simple variant

Analysis. Consider the t-th iteration of the RL process, i.e., updating from $\boldsymbol{\theta}_t$ to $\boldsymbol{\theta}_{t+1}$, and use $\pi_{\mathsf{ref}} = \pi_{\boldsymbol{\theta}_t}$ as the reference policy. For a specific task/query x, recall from Section 1 the original objective:

$$\max_{\boldsymbol{\theta}} \quad J(\boldsymbol{\theta}; x, \pi_{\boldsymbol{\theta}_t}) \coloneqq \mathbb{E}_{y \sim \pi_{\boldsymbol{\theta}}(\cdot|x)}[r(x,y)] - \tau \cdot D_{\mathsf{KL}}\big(\pi_{\boldsymbol{\theta}}(\cdot|x) \| \pi_{\boldsymbol{\theta}_t}(\cdot|x)\big).$$

We leverage the analysis in Section 2, and take a closer look at the following pairwise loss for a_i and a_j, normalized by $1/(1+\tau)^2$ to make the loss scale invariant to the hyperparameter τ:

$$\begin{aligned} \frac{(a_i - a_j)^2}{(1+\tau)^2} = \frac{1}{(1+\tau)^2} \Bigg[&\Big(r(x,y_i) - r(x,y_j)\Big) \\ &- \tau \cdot \Big(\big(\log \pi_{\boldsymbol{\theta}}(y_i|x) - \log \pi_{\boldsymbol{\theta}_t}(y_i|x)\big) - \big(\log \pi_{\boldsymbol{\theta}}(y_j|x) - \log \pi_{\boldsymbol{\theta}_t}(y_j|x)\big)\Big)\Bigg]^2. \end{aligned}$$

The trick here is that, if we only intend to take one gradient step of this loss at $\boldsymbol{\theta} = \boldsymbol{\theta}_t$, then the value of $(\log \pi_{\boldsymbol{\theta}}(y_i|x) - \log \pi_{\boldsymbol{\theta}_t}(y_i|x)) - (\log \pi_{\boldsymbol{\theta}}(y_j|x) - \log \pi_{\boldsymbol{\theta}_t}(y_j|x))$ is simply zero. As a result,

$$\nabla_{\boldsymbol{\theta}} \frac{(a_i - a_j)^2}{(1+\tau)^2}\Big|_{\boldsymbol{\theta}_t} = \frac{-2\tau}{(1+\tau)^2} \Big(r(x,y_i) - r(x,y_j)\Big) \Big(\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(y_i|x)|_{\boldsymbol{\theta}_t} - \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(y_j|x)|_{\boldsymbol{\theta}_t}\Big),$$

and thus

$$\nabla_{\boldsymbol{\theta}} \sum_{1 \le i < j \le K} \frac{(a_i - a_j)^2}{(1+\tau)^2}\Big|_{\boldsymbol{\theta}_t}$$

$$
\begin{aligned}
&= \sum_{1\le i<j\le K} \frac{-2\tau}{(1+\tau)^2}\Big(r(x,y_i)-r(x,y_j)\Big)\Big(\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(y_i|x)|_{\boldsymbol{\theta}_t}-\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(y_j|x)|_{\boldsymbol{\theta}_t}\Big)\\
&= \sum_{1\le i<j\le K} \frac{-2\tau}{(1+\tau)^2}\bigg(\big(r(x,y_i)-r(x,y_j)\big)\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(y_i|x)|_{\boldsymbol{\theta}_t}+\big(r(x,y_j)-r(x,y_i)\big)\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(y_j|x)|_{\boldsymbol{\theta}_t}\bigg)\\
&= \frac{-2\tau}{(1+\tau)^2}\sum_{1\le i\le K}\sum_{1\le j\le K}\big(r(x,y_i)-r(x,y_j)\big)\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(y_i|x)|_{\boldsymbol{\theta}_t}\\
&= \frac{-2\tau}{(1+\tau)^2}\sum_{1\le i\le K} K\cdot\big(r(x,y_i)-\overline{r}(x)\big)\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(y_i|x)|_{\boldsymbol{\theta}_t},
\end{aligned}
$$

where $\overline{r}(x) \coloneqq \frac{1}{K}\sum_{j\in[K]} r(x,y_j)$ in the last line.

Algorithm. To this end, we update from $\boldsymbol{\theta}_t$ to $\boldsymbol{\theta}_{t+1}$ by taking one gradient step of the following surrogate loss, where we simplify the constant factor from $2\tau/(1+\tau)^2$ to $1/(1+\tau)$ and also drop the K factor:

$$
\min_{\boldsymbol{\theta}} \quad \widehat{J}(\boldsymbol{\theta};x) \coloneqq -\frac{1}{1+\tau}\sum_{1\le i\le K}\big(r(x,y_i)-\overline{r}(x)\big)\log\pi_{\boldsymbol{\theta}}(y_i|x).
$$

This is simply the standard policy gradient using the group mean reward as the baseline, but derived differently and applicable to off-policy cases. The hyperparameter τ controls the size of each policy update.

As a heuristic, we simply add a regularization term (denoted by g) to the above objective when additional regularization with respect to a fixed policy, e.g., a SFT model π_{sft}, is desired:

$$
\min_{\boldsymbol{\theta}} \quad \widehat{J}(\boldsymbol{\theta};x) \coloneqq -\frac{1}{1+\tau}\sum_{1\le i\le K}\big(r(x,y_i)-\overline{r}(x)\big)\log\pi_{\boldsymbol{\theta}}(y_i|x)+\beta\cdot g\big(\pi_{\boldsymbol{\theta}},\pi_{\mathsf{sft}};x,y_1,\ldots,y_K\big).
$$

References

[1] Yannis Flet-Berliac, Nathan Grinsztajn, Florian Strub, Bill Wu, Eugene Choi, Chris Cremer, Arash Ahmadian, Yash Chandak, Mohammad Gheshlaghi Azar, Olivier Pietquin, and Matthieu Geist. Contrastive policy gradient: Aligning llms on sequence-level scores in a supervised-friendly fashion. In *EMNLP*, 2024.

[2] Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, Haiqing Guo, Han Zhu, Hao Ding, Hao Hu, Hao Yang, Hao Zhang, Haotian Yao, Haotian Zhao, Haoyu Lu, Haoze Li, Haozhen Yu, Hongcheng Gao, Huabin Zheng, Huan Yuan, Jia Chen, Jianhang Guo, Jianlin Su, Jianzhou Wang, Jie Zhao, Jin Zhang, Jingyuan Liu, Junjie Yan, Junyan Wu, Lidong Shi, Ling Ye, Longhui Yu, Mengnan Dong, Neo Zhang, Ningchen Ma, Qiwei Pan, Qucheng Gong, Shaowei Liu, Shengling Ma, Shupeng Wei, Sihan Cao, Siying Huang, Tao Jiang, Weihao Gao, Weimin Xiong, Weiran He, Weixiao Huang, Wenhao Wu, Wenyang He, Xianghui Wei, Xianqing Jia, Xingzhe Wu, Xinran Xu, Xinxing Zu, Xinyu Zhou, Xuehai Pan, Y. Charles, Yang Li, Yangyang Hu, Yangyang Liu, Yanru Chen, Yejie Wang, Yibo Liu, Yidao Qin, Yifeng Liu, Ying Yang, Yiping Bao, Yulun Du, Yuxin Wu, Yuzhi Wang, Zaida Zhou, Zhaoji Wang, Zhaowei Li, Zhen Zhu, Zheng Zhang, Zhexu Wang, Zhilin Yang, Zhiqi Huang, Zihao Huang, Ziyao Xu, and Zonghan Yang. Kimi k1.5: Scaling reinforcement learning with llms. *arXiv*, 2025.