Efficient Meta Learning via Minibatch Proximal Update

Pan Zhou* Xiao-Tong Yuan† Huan Xu‡ Shuicheng Yan△ Jiashi Feng*

* National University of Singapore, † Nanjing University of Information Science & Technology, ‡ Alibaba, △ YITU Tech (pzhou@u.nus.edu xtyuan@nuist.edu.cn Huan.xu@alibaba-inc.com {eleyans, elefjia}@nus.edu.sg)

homepage

Problem Setup

Problems: how to learn task-level knowledge from various observed tasks such that it could facilitate new task learning, e.g. few-shot learning and new policy learning in RL?

Proposed Efficient Meta Learning via Minibatch Proximal Update (Meta-MinibatchProx)

Meta-MinibatchProx: it learns a good prior model $\boldsymbol{w}$ from observed tasks that is close to the optimal models of new similar tasks, facilitating new task learning.

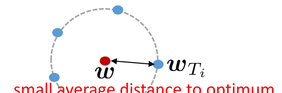
Training model: given a meta task distribution $\mathcal{T}$, we consider the meta-learning model:

$$\min_{\boldsymbol{w}} F(\boldsymbol{w}) := \frac{1}{n}\sum_{i=1}^{n} \min_{\boldsymbol{w}_{T_i}} \Big\{ \mathcal{L}_{D_{T_i}}(\boldsymbol{w}_{T_i}) + \frac{\lambda}{2}\|\boldsymbol{w}_{T_i} - \boldsymbol{w}\|_2^2 \Big\}. \tag{1}$$

where each task $T \sim \mathcal{T}$ only contains K training samples $D_{T_i} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}$; $\mathcal{L}_{D_T}(\boldsymbol{w}_T) = \frac{1}{K}\sum_{(\boldsymbol{x},\boldsymbol{y})\in D_T} \ell(f(\boldsymbol{w},\boldsymbol{x}),\boldsymbol{y})$ denotes the empirical loss on D_T with predictor f and loss ℓ.

Inner level of intra-task learning: it finds the task-specific optimal model parameter $\boldsymbol{w}_T$ of task T around the prior $\boldsymbol{w}$.

Outer level of inter-task learning: it uses optimal parameters $\boldsymbol{w}_T$ to tune $\boldsymbol{w}$ such that $\boldsymbol{w}$ is close to all $\boldsymbol{w}_T$ in average.



Test model: given a task $T \sim \mathcal{T}$ with K samples $D_T = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^K$, we adapt the prior model $\boldsymbol{w}$ to task T by minimizing

$$\min_{\boldsymbol{w}_T} \mathcal{L}_{D_T}(\boldsymbol{w}_T) + \frac{\lambda}{2}\|\boldsymbol{w}_T - \boldsymbol{w}\|_2^2, \tag{2}$$

Benefit: it requires a few data for adaptation, since prior model $\boldsymbol{w}$ is close to optimal model of T in expectation as existing and new tasks are from the same distribution and are similar.

Advantages over MAML

Higher computational efficiency. Meta-MinibatchProx only computes the gradient (see algorithm below) and is more efficient than MAML requiring Hessian computation.

More accurate solution to inner problem. MAML finds a good initialization $\boldsymbol{w}$ via

$$\boldsymbol{w}_T^* = \boldsymbol{w} - \eta\nabla\mathcal{L}_{D_T}(\boldsymbol{w}) = \operatorname*{argmin}_{\boldsymbol{w}_T} \langle \nabla\mathcal{L}_{D_T}(\boldsymbol{w}), \boldsymbol{w}_T - \boldsymbol{w}\rangle + \frac{1}{2\eta}\|\boldsymbol{w}_T - \boldsymbol{w}\|_2^2.$$

Meta-MinibatchProx finds the task-specific optimal model by minimizing

$$\boldsymbol{w}_T^* = \min_{\boldsymbol{w}_T} \mathcal{L}_{D_T}(\boldsymbol{w}_T) + \frac{\lambda}{2}\|\boldsymbol{w}_T - \boldsymbol{w}\|_2^2 = \min_{\boldsymbol{w}_T} \langle \nabla\mathcal{L}_{D_T}(\boldsymbol{w}), \boldsymbol{w}_T - \boldsymbol{w}\rangle + \frac{\lambda}{2}\|\boldsymbol{w}_T - \boldsymbol{w}\|_2^2 + \frac{1}{2}\langle \nabla^2\mathcal{L}_{D_T}(\boldsymbol{w})(\boldsymbol{w}_T - \boldsymbol{w}), (\boldsymbol{w}_T - \boldsymbol{w})\rangle + \frac{1}{6}\langle \nabla^3\mathcal{L}_{D_T}(\boldsymbol{w}), (\boldsymbol{w}_T - \boldsymbol{w})^{\otimes^3}\rangle + \cdots$$

So Meta-MinibatchProx uses higher-order information of $\mathcal{L}_{D_T}$ beyond gradient to search.

More flexible regularization. Meta-MinibatchProx can handle different possible structures in model parameter space by using proper $\ell_{p,q}$-regularizer $\|\boldsymbol{w}_{T_i} - \boldsymbol{w}\|_p^q$, while MAML cannot due to its fixed update rule. E.g., Meta-MinibatchProx could use $\ell_{2,1}$-norm to handle outlier-tasks.

Stochastic Gradient Meta-Optimization

$$\min_{\boldsymbol{w}} \frac{1}{n}\sum_{i=1}^{n} \phi_{D_T}(\boldsymbol{w}) \quad \text{where} \quad \phi_{D_T}(\boldsymbol{w}) = \min_{\boldsymbol{w}_T} \mathcal{L}_{D_T}(\boldsymbol{w}_T) + \frac{\lambda}{2}\|\boldsymbol{w}_T - \boldsymbol{w}\|_2^2. \tag{3}$$

The difficulty in applying SGD is how to compute the gradient of meta loss $\phi_{D_T}(\boldsymbol{w})$ w.r.t. $\boldsymbol{w}$.

Stochastic gradient computation of $\phi_{D_T}(\boldsymbol{w})$. Assume $\mathcal{L}_{D_T}$ is differentiable and $\boldsymbol{w}_T^* = \operatorname{argmin}_{\boldsymbol{w}_T} \mathcal{L}_{D_T}(\boldsymbol{w}_T) + \frac{\lambda}{2}\|\boldsymbol{w}_T - \boldsymbol{w}^*\|_2^2$. Then the gradient of the meta-loss $\phi_{D_T}(\boldsymbol{w})$ is given by $\nabla\phi_{D_T}(\boldsymbol{w}) = \lambda(\boldsymbol{w} - \boldsymbol{w}_T^*)$.

Stochastic Gradient Meta-Optimization

In practice, we only approximately solve the inner problem: $\boldsymbol{w}_T^* \approx \operatorname{argmin}_{\boldsymbol{w}_T} \mathcal{L}_{D_T}(\boldsymbol{w}_T)$.

Algorithm 1: SGD for Meta-MinibatchProx

Input : Initial point $\boldsymbol{w}_0$, learning rate $\{\eta_s\}$.
for $s = 0$ ***to*** $S-1$ **do**
  Randomly select a mini-batch of tasks $\{T_i\}$ of size b_s from the observed n tasks.
  for $T_i \in \{T_i\}$
    Compute an ϵ_s-approximate stable minimizer $\boldsymbol{w}_{T_i}^s$ to the within-meta-task problem $\min_{\boldsymbol{w}_{T_i}} g(\boldsymbol{w}_{T_i}) := \mathcal{L}_{D_{T_i}}(\boldsymbol{w}_{T_i}) + \frac{\lambda}{2}\|\boldsymbol{w}_{T_i} - \boldsymbol{w}^s\|_2^2$ such that $\|\nabla g(\boldsymbol{w}_{T_i}^s)\|_2^2 \le \epsilon_s$.
  end
  Update the meta parameter $\boldsymbol{w}^{s+1} = \boldsymbol{w}^s - \eta_s\lambda(\boldsymbol{w}^s - \frac{1}{b_s}\sum_{i=1}^{b_s} \boldsymbol{w}_{T_i}^s)$.
end
Output : the parameter initialization $\boldsymbol{w}^S$ of model f.

Convergence guarantees. Assume $\mathcal{L}_{D_T}(\boldsymbol{w}_T)$ is differentiable and for each task, its optimum $\boldsymbol{w}_T^* = \operatorname{argmin}_{\boldsymbol{w}_T} \mathcal{L}_{D_T}(\boldsymbol{w}_T) + \frac{\lambda}{2}\|\boldsymbol{w}_T - \boldsymbol{w}\|_2^2$ satisfies $\mathbb{E}[\|\boldsymbol{w}_T^* - \boldsymbol{w}\|_2^2] \le \sigma^2$. Let $\boldsymbol{w}^* = \operatorname{argmin}_{\boldsymbol{w}} F(\boldsymbol{w})$ and $\boldsymbol{w}_{T_i}^{*s} = \operatorname{argmin}_{\boldsymbol{w}_{T_i}} \mathcal{L}_{D_{T_i}}(\boldsymbol{w}_{T_i}) + \frac{\lambda}{2}\|\boldsymbol{w}_{T_i} - \boldsymbol{w}^s\|_2^2$.

(1) Convex setting. Assume $\mathcal{L}_{D_T}(\boldsymbol{w}_T)$ is convex. Then by setting $\eta_s = \frac{2}{s\lambda}$, $\epsilon_s = \frac{c}{S}$, $\alpha = \frac{8S\lambda^2\sigma^2}{S-1} + c(1 + \frac{8}{S-1})$ with a constant c, we have

$$\mathbb{E}[\|\boldsymbol{w}^S - \boldsymbol{w}^*\|_2^2] \le \frac{\alpha}{\lambda^2 S} \quad \text{and} \quad \mathbb{E}\Big[\Big\|\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{w}_{T_i}^{*S} - \boldsymbol{w}^S\Big\|_2^2\Big] \le \frac{L^2\alpha}{(\lambda + L)^2 S}.$$

(2) Non-convex setting. Assume $\mathcal{L}_{D_T}(\boldsymbol{w}_T)$ is L-smooth. Then by setting $\lambda > L$, $\eta_s = \sqrt{\frac{\Delta}{\gamma S}}$, $\epsilon_s = \frac{c}{\sqrt{S}}$ with $\gamma = \frac{\lambda^3 L}{(\lambda+L)}(\sigma^2 + \frac{c}{\sqrt{S}(\lambda-L)^2})$ and $\Delta = F(\boldsymbol{w}^0) - F(\boldsymbol{w}^*)$, we have

$$\min_s \mathbb{E}[\|\nabla F(\boldsymbol{w}^s)\|_2^2] = \lambda^2 \min_s \mathbb{E}\Big[\Big\|\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{w}_{T_i}^{*s} - \boldsymbol{w}^s\Big\|_2^2\Big] \le \frac{1}{\sqrt{S}}\Big[4\sqrt{\Delta\gamma} + \frac{2c\lambda^2}{(\lambda - L)^2}\Big].$$

Statistical Justification: Benefit of Prior Model in Meta Learning

Assume we have learned an optimal prior hypothesis $\boldsymbol{w}^* = \operatorname{argmin}_{\boldsymbol{w}} F(\boldsymbol{w})$. Here we view $\boldsymbol{w}^*$ as a deterministic hypothesis.
For any $T \sim \mathcal{T}$ and $D_T = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^K \sim T$, define the population solution as

$$\boldsymbol{w}_{T,E}^* \in \operatorname*{argmin}_{\boldsymbol{w}_T} \big\{ \mathcal{L}(\boldsymbol{w}_T) := \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim T}[\ell(f(\boldsymbol{w}_T, \boldsymbol{x}), \boldsymbol{y})] \big\}$$

and empirical solution in Meta-MinibatchProx as

$$\boldsymbol{w}_T^* = \operatorname*{argmin}_{\boldsymbol{w}_T} \mathcal{L}_{D_T}(\boldsymbol{w}_T) + \frac{\lambda}{2}\|\boldsymbol{w}_T - \boldsymbol{w}^*\|_2^2.$$

Benefit of prior $\boldsymbol{w}^*$ for new task learning. Suppose $\ell(f(\boldsymbol{w},\boldsymbol{x}),\boldsymbol{y})$ is G-Lipschitz continuous and convex w.r.t. $\boldsymbol{w}$. Then we have

$$\mathbb{E}_{T\sim\mathcal{T}}\mathbb{E}_{D_T\sim T}\big[\mathcal{L}(\boldsymbol{w}_T^*) - \mathcal{L}(\boldsymbol{w}_{T,E}^*)\big] \le \frac{\sqrt{2}G}{\sqrt{K}}\sqrt{\mathbb{E}_{T\sim\mathcal{T}}\Big[\|\boldsymbol{w}^* - \boldsymbol{w}_{T,E}^*\|^2\Big]}.$$

Remark. Training model guarantees $\boldsymbol{w}^*$ to be close to $\boldsymbol{w}_{T,E}^*$ in expectation.

First-order optimality. Suppose $\ell(f(\boldsymbol{w},\boldsymbol{x}),\boldsymbol{y})$ is G-Lipschitz continuous and L-smooth w.r.t. $\boldsymbol{w}$. Then for $\lambda > L$, it holds that

$$\mathbb{E}_{T\sim\mathcal{T}}\Big[\big\|\mathbb{E}_{D_T\sim T}[\nabla\mathcal{L}(\boldsymbol{w}_T^*)]\big\|^2\Big] \le \frac{32G^2L^2}{(\lambda-L)^2K^2} + \frac{8G^2}{(\lambda-L)\beta K} + \frac{2}{\beta}\mathbb{E}_{T\sim\mathcal{T}}[\mathcal{L}(\boldsymbol{w}^*) - \mathcal{L}(\boldsymbol{w}_T^*)],$$

where $\beta = \frac{1}{\lambda}\big[1 - \frac{L}{2\lambda}\big]$ is a constant.

Remark. $\boldsymbol{w}^*$ helps obtain small gradient.

Comparison with Existing Works

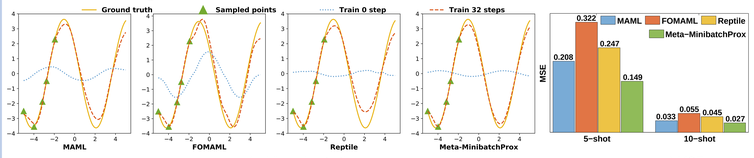
Recently, Denevi et al. (2018, 2019) and Khodak et al. (2019) also consider similar prior hypothesis biased regularized empirical risk minimization.

But Denevi et al. focus on convex linear models in contrast ours which is developed for both convex and non-convex learning problems.

Different from online convex meta-learning framework developed in (Khodak et al. 2019), we use a simple yet scalable paradigm within SGD for stochastic meta-optimization which is particularly friendly for computational and statistical complexity analysis in both convex and non-convex settings.

Experiments (code is available at `https://panzhous.github.io/`)

Few-shot regression.



(a) Visual illustration (b) MSE

Few-shot classification.

Table 1: Few-shot classification accuracy (%) on miniImageNet.

method	1-shot 5-way	5-shot 5-way	1-shot 20-way	5-shot 20-way
Matching Net	43.56 ± 0.84	55.31 ± 0.73	17.31 ± 0.22	22.69 ± 0.20
Meta-LSTM	43.33 ± 0.77	60.60 ± 0.71	16.70 ± 0.23	26.06 ± 0.25
MAML	46.21 ± 1.76	61.12 ± 1.01	16.01 ± 0.52	18.34 ± 0.33
FOMAML	45.53 ± 1.58	61.02 ± 1.12	15.21 ± 0.54	17.67 ± 0.47
Reptile	47.07 ± 0.26	62.74 ± 0.37	18.27 ± 0.16	28.71 ± 0.19
Meta-MinibatchProx	**48.51 ± 0.92**	**64.15 ± 0.92**	**20.50 ± 0.35**	**33.61 ± 0.41**
MAML + Transduction	48.70 ± 1.84	63.11 ± 0.92	16.49 ± 0.58	19.29 ± 0.29
FOMAML + Transduction	48.07 ± 1.75	63.15 ± 0.91	15.80 ± 0.61	18.15 ± 0.43
Reptile + Transduction	49.97 ± 0.32	65.99 ± 0.58	18.76 ± 0.17	29.15 ± 0.22
Meta-MinibatchProx + Transduction	**50.77 ± 0.90**	**67.43 ± 0.89**	**21.17 ± 0.38**	**34.30 ± 0.41**

Table 2: Few-shot classification accuracy (%) on tieredImageNet.

method	1-shot 5-way	5-shot 5-way	1-shot 10-way	5-shot 10-way
Matching Net	34.95 ± 0.89	43.95 ± 0.85	22.46 ± 0.34	31.19 ± 0.30
Meta-LSTM	33.71 ± 0.76	46.56 ± 0.79	22.09 ± 0.43	35.65 ± 0.39
MAML	49.60 ± 1.83	66.58 ± 1.78	33.18 ± 1.23	49.05 ± 1.32
FOMAML	48.01 ± 1.74	64.07 ± 1.72	30.31 ± 1.12	46.54 ± 1.24
Reptile	49.12 ± 0.43	65.99 ± 0.42	31.79 ± 0.28	47.82 ± 0.30
Meta-MinibatchProx	**50.14 ± 0.92**	**68.30 ± 0.91**	**33.68 ± 0.64**	**51.84 ± 0.65**
MAML + Transduction	51.67 ± 1.81	70.30 ± 1.75	34.44 ± 1.19	53.32 ± 1.33
FOMAML + Transduction	50.12 ± 1.82	67.43 ± 1.80	31.53 ± 1.08	49.99 ± 1.36
Reptile + Transduction	51.06 ± 0.45	69.94 ± 0.42	33.79 ± 0.29	51.27 ± 0.31
Meta-MinibatchProx + Transduction	**54.37 ± 0.93**	**71.45 ± 0.94**	**35.56 ± 0.60**	**54.50 ± 0.71**

Outlier-Corrupted Tasks.

Assume there are a few outlier-tasks $\mathcal{O} = \{T_o\}$ whose optima $\{\boldsymbol{w}_o\}$ are quite far from the optima $\{\boldsymbol{w}_s\}$ of inlier/normal-tasks $\mathcal{S} = \{T_s\}$.

Here we add 5% outlier images with zero pixels into each training class in miniImageNet. If a sampled task T contains these outlier images, it is an outlier-task.

Meta-MinibatchProx uses the robust $\ell_{2,1}$-norm $\frac{1}{n}\sum_{i=1}^{n}\|\boldsymbol{w}_{T_i} - \boldsymbol{w}\|_2$ that tolerates relatively large distances between $\boldsymbol{w}$ and $\{\boldsymbol{w}_o\}$.