Revisiting Gradient-based Meta-learning Optimization via Stochastic Composition Optimization

Huaxiu Yao

College of Information Science and Technology

28^{th} Nov 2018

Introduction: Background (Meta-learning)

Supervised Learning

Input: $\mathbf{x}$;Output: $\mathbf{y}$; Data: $(\mathbf{x}, \mathbf{y})_i$
Relation: $\mathbf{y} = f(\mathbf{x}; \theta)$

Meta-Supervised Learning

Input: $\mathcal{D}_{train}$, $\mathbf{x}_{test}$; Output: $\mathbf{y}_{test}$; Data: $\mathcal{D}_i = (\mathbf{x}, \mathbf{y})_j$
Relation: $\mathbf{y} = f(\mathcal{D}_{train}, \mathbf{x}_{test}; \theta)$

Why it is so important?

Reduces the problem to the design & optimization of f.

Applications

1. **Learning to optimization**: learn how to design the hyperparameters (e.g., step size)
2. **Few-shot Learning**: learning how to classify images with a few samples.

Gradient-based Meta-learning

Key Idea: Train over many tasks, to learn parameter vector θ that transfers [Finn et al. 2017]

θ: parameter vector being meta-learned (i.e., the learnable parameters of f)

$\phi_{\mathbf{i}}^*$: optimal parameter vector for task i

Loss Function (one gradient as exemplarity):

$$\min_{\theta} \sum_{task\ i} \mathcal{L}_{test}^{i}(\theta - \alpha \nabla_{\theta} \mathcal{L}_{train}^{i}(\theta))$$

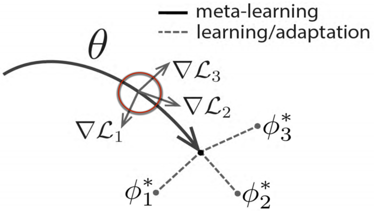


Figure 1: Meta-learning

How to solve this optimization problem?

Stochastic Gradient Descent (e.g., Adam)?

Simply using stochastic gradient descent may introduce biased estimation.

Challenge: How to Optimize the Loss Function

Loss Function:

$$\min_{\theta} \sum_{task\ i} \mathcal{L}^{i}_{test}(\theta - \alpha \nabla_{\theta} \mathcal{L}^{i}_{train}(\theta))$$

The loss function can be regarded as a nested function, which can be revised as:

$$\min_{\theta} \sum_{task\ i} \mathcal{L}^{i}_{test}(\theta - \alpha \nabla_{\theta} \mathcal{L}^{i}_{train}(\theta)) \Rightarrow \mathbb{E}[g_1(\mathbb{E}[g_2(\theta)])]$$

$$g_1(\theta) = \mathcal{L}_{test}(\cdot);\ g_2(\theta) = \theta - \alpha \nabla_{\theta} \mathcal{L}_{train}(\theta)$$

where $\mathcal{L}_{train}(\theta)$ is a complex function and the variable is θ.
The gradient of loss function is:

$$\nabla G(\theta) = \nabla g_2(\theta) \nabla g_1(g_2(\theta))$$

Challenge: $g_2(\theta)$ is unknown with finite sample oracles (SO).

Solution: Stochastic Composition Optimization

Optimality condition of problem (assuming that the problem is convex) is [Wang et al. 2017]:

$$\nabla G(\theta^*)'(\theta - \theta^*) \geq 0$$

The unbiased sample of ∇G is difficult to obtain. We introduce a new function z and the optimality condition can be:

$$\begin{aligned}
&(\nabla g_2(\theta)\nabla g_1(z))'(\theta - \theta^*) \geq 0 \\
&z = g_2(\theta)
\end{aligned}$$

For a given (θ, z), we can get unbiased results.
Multi-level Optimization: we can also easily extend to multi-level optimization, which is (assuming there are T steps):

$$\begin{aligned}
&(\nabla g_T(\theta)\nabla g_{T-1}(z_{T-1})\cdots\nabla g_1(z_1))'(\theta - \theta^*) \geq 0 \\
&z_{T-1} = g_T(\theta) \\
&z_1 = g_2(z_2)
\end{aligned}$$

Solution: Stochastic Composition Optimization

Stochastic Composition Optimization The formulation is:

$$\min_{\theta} \mathbb{E}[g_1(\mathbb{E}[g_2(\theta)])]$$

Input: SO, K, stepsize $\{\alpha_k\}_{k=1}^{K}$, $\{\beta_k\}_{k=1}^{K}$, **data,** z_0

1: **for** $k = 1$ to K **do**

2: Query the SO to obtain $\nabla g_2(\theta_k)$, $g_2(\theta_k)$, $g_1(y_{k+1})$

Update $y_{k+1} = (1 - \beta_k) y_k + \beta_k g_2(\theta_k)$

Update $x_{k+1} = x_k - \alpha \nabla g_2(\theta_k) \nabla g_1(y_{k+1})$

3: **end for**

Algorithm 1: Pseudocode for Stochastic Composition Gradient Descent

Experiments: Synthetic Dataset

Dataset Description: We generate the task from a familiy functions. The base function is:

$$y = A_1 sin(wx + b_1) + A_2 sin(wx + b_2)$$

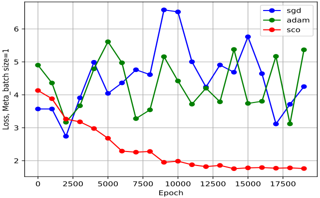
$$A_1 \sim U[1.0, 5.0], A_2 \sim U[-1.0, 2.0], b_1 \sim U[0, 2\pi], w \sim U[0.5, 2.0]$$

Each composition represents one task. For each task, 5 samples are used for training and 15 samples for testing.

Compared Methods: We compare the results with stochastic gradient descent (SGD) and Adam.

Base model: Multiple Layer Perception with two hidden layers (each layer has 40 neurons).

Results:



Experiments: Real-world Dataset

Dataset Description: We selected 100 class from Imagenet, 64, 16, 20 classes are used for training, validation and testing. For each task, we randomly select 5 classes (5-way), each class has 1 or 5 (1-shot or 5-shot) sample for training and 15 samples for testing.
Compared Methods: Adam.
Base model: Four layers convolutional neural network.
Results: The accuracy and training loss are shown in Table 1 and Figure 2.

Table 1: Classification Accuracy

Model	5-way 1-shot	5-way 5-shot
Adam	48.70±1.68	63.11±0.92
SCO	**50.01±1.65**	**64.02±0.83**

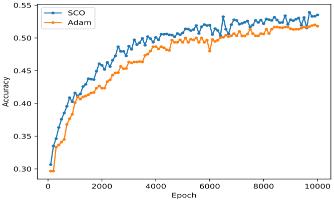


Figure 3: Accuracy of 1-shot

Discussion & Conclusion

Theoretical Analysis: Since the meta-learning problem is non-convex case, the convergence error bound of stochastic gradient descent algorithm is $\mathcal{O}(n^{-4/(7+T)})$.

Weakness:

1. The computational cost of this algorithm is still high, usually take several hours to train.
2. Empirically, sometimes the algorithms is a little unstable.

Conclusion & Discussion:

1. By using stochastic composition gradient descent on gradient-based meta-learning problem, we can alleviate the effect of biased SO. Empirically, we achieve better performance on synthetic and real-world datasets.
2. In the future, we can apply the stochastic composition optimization to more meta-learning applications, especially reinforcement learning case. In addition, we can apply to more meta-learning frameworks (e.g., recurrent meta-learning)