Review of Continuous Adaptation Via Meta-Learning in Nonstationary and Competitive Environments

Guannan Hu

April 18, 2018

1 A Probabilistic View of Model-Agnostic Meta-LEarning (MAML)

Assume that we are given a distribution over tasks, $\mathcal{D}(T)$, where each task, T, is a tuple:

$$T := (L_T, P_T(\mathbf{x}), P_T(\mathbf{x}_{t+1}|\mathbf{x}, \mathbf{a}_t), H) \tag{1}$$

L_T is a task specific loss function that maps a trajectory, $\boldsymbol{\tau} := (\mathbf{x}_0, \mathbf{a}_1, \mathbf{x}_1, R_1, ..., \mathbf{a}_H, \mathbf{x}_H, R_H) \in \mathcal{T}$, to a loss value, i.e., $L_T : \mathcal{T} \to \mathbb{R}$; $P_T\mathbf{x}$ and $P_T(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{a}_t)$ define the Markovian dynamics of the environment in task T; H denotes the horizon; observations, $\mathbf{x}_t$, and actions, $\mathbf{a}_t$, are elements (typically, vectors) of the observation space, $\mathcal{X}$, and action space, $\mathcal{A}$, respectively. The loss of a trajectory, $\boldsymbol{\tau}$, is the negative cumulative reward, $L_T(\boldsymbol{\tau}) := -\sum_{t=1}^{H} R_t$.

The goal of meta-learning is to find a procedure which, given access to limited experience on a task sampled from $\mathcal{D}(T)$, can produce a good policy for solving it. More formally, after querying K trajectories from a task $T \sim \mathcal{D}(T)$ under policy π_θ, denoted $\boldsymbol{\tau}_\theta^{1:K}$, we would like to construct a new, task-specific policy, π_ϕ, that would minimize the expected subsequent loss on the task T. In particular, MAML constructs parameters of the task-specific policy, ϕ, using gradient of L_T w.r.t. θ:

$$\phi := \theta - \alpha \nabla_\theta L_T(\boldsymbol{\tau}_\theta^{1:K}), \text{ where } L_T(\boldsymbol{\tau}_\theta^{1:K}) := \frac{1}{K}\sum_{k=1}^{K} L_T(\boldsymbol{\tau}_\theta^{k}), \text{ and } \boldsymbol{\tau}_\theta^{k} \sim P_T(\boldsymbol{\tau}|\theta) \tag{2}$$

We call (2) the *adaptation update* with a step α. The adaptation update is parametrized by θ, which we optimize by minimizing the expected loss over the distribution of tasks, $D(T)$—the *meta-loss*:

$$\min_\theta \mathbb{E}_{T \sim D(T)}[\mathcal{L}_T(\theta)], \text{ where } \mathcal{L}_T(\theta) := \mathbb{E}_{\boldsymbol{\tau}_\theta^{1:K} \sim P_T(\boldsymbol{\tau}|\theta)}[\mathbb{E}_{\boldsymbol{\tau}_\phi \sim P_T(\boldsymbol{\tau}|\phi)}[L_T(\boldsymbol{\tau}_\phi)|\boldsymbol{\tau}_\theta^{1:K}, \theta]] \tag{3}$$

where $\boldsymbol{\tau}_\theta$ and $\boldsymbol{\tau}_\phi$ are trajectories obtained under π_θ and π_ϕ, respectively.

In general, we can think of the task, trajectories, and policies, as random variables, where ϕ is generated from some conditional distribution $P_T(\phi|\theta, \boldsymbol{\tau}_{1:k}) := \delta(\theta - \alpha\nabla_\theta \frac{1}{k}\sum_{k=1}^{K} L_T(\boldsymbol{\tau}_k))$. To optimize, we can use the policy gradient method, where the gradient of $\mathcal{L}_T$ is as follows:

$$\nabla_\theta \mathcal{L}_T(\theta) = \mathbb{E}_{\substack{\boldsymbol{\tau}_\theta^{1:K} \sim P_T(\boldsymbol{\tau}|\theta) \\ \boldsymbol{\tau}_\phi \sim P_T(\boldsymbol{\tau}|\phi)}} \left[L_T(\boldsymbol{\tau}_\phi) \left[\nabla_\theta \log \pi_\phi(\boldsymbol{\tau}_\phi) + \nabla_\theta \sum_{k=1}^{K} \log \pi_\theta(\boldsymbol{\tau}_\theta^k) \right] \right] \tag{4}$$

The expected loss on a task, $\mathcal{L}_T$, can be optimized with trust-region policy (TRPO) or proximal policy (PPO) optimization methods.

Algorithm 1 Meta-learning at training time

Input: Distribution over pairs of tasks, $\mathcal{P}(T_i, T_{i+1})$, learning rate, β.

```
Randomly initialize θ and α.
repeat
    Sample a batch of task pairs, {(T_i, T_{i+1})}_{i=1}^n.
    for All task pairs T_i, T_{i+1} in the batch do.
        Sample traj. τ_θ^{1:K} from T_i using π_θ.
        Compute φ = φ(τ_θ^{1:K}, θ, α) as given in (7).
        Sample traj. τ_φ from T_{i+1} using π_φ.
    end for
    Compute ∇_θ L_{T_i,T_{i+1}} and ∇_α L_{T_i,T_{i+1}} using τ_θ^{1:K} and τ_φ as given in (8).
    Update θ ← θ + β∇_θ L_T(θ, α).
    Update α ← α + β∇_α L_T(θ, α).
until Convergence
```

Output: Optimal θ^* and α^*.

Algorithm 2 Adaptation at execution time.

Input: A Stream of tasks, $T_1, T_2, T_3, \ldots.$

```
Initialize φ = θ.
while do there are new incoming tasks
    Get a new task, T_i, from the stream.
    Solve T_i using π_φ policy.
    While solving T_i, collect trajectories, τ_{i,φ}^{1:K}.
    Update φ ← φ(τ_{i,φ}^{1:K}, θ*, α*) using importance-corrected meta-update as in (9).
end while
```

$$\min_{\theta} \mathbb{E}_{\mathcal{P}(T_0),\mathcal{P}(T_{i+1}|T_i)} \left[\sum_{i=1}^{L} \mathcal{L}_{T_i,T_{i+1}}(\theta) \right] \tag{5}$$

$$\mathcal{L}_{T_i,T_{i+1}}(\theta) := \mathbb{E}_{\boldsymbol{\tau}_{i,\theta}^{1:K} \sim P_{T_i}(\boldsymbol{\tau}|\theta)} \left[\mathbb{E}_{\boldsymbol{\tau}_{i+1,\phi} \sim P_{T_{i+1}}(\boldsymbol{\tau}|\phi)} [L_{T_{i+1}}(\boldsymbol{\tau}_{i+1,\phi}) | \boldsymbol{\tau}_{i,\theta}^{1:K}, \theta] \right] \tag{6}$$

$$\begin{aligned} \phi_i^0 &:= \theta, \qquad \boldsymbol{\tau}_\theta^{1:K} \sim P_{T_i}(\boldsymbol{\tau}|\theta), \\ \phi_i^m &:= \phi_i^{m-1} - \alpha_m \nabla_{\phi_i^{m-1}} L_{T_i}\left(\boldsymbol{\tau}_{i,\phi_i^{m-1}}^{1:K}\right), m = 1, ..., M-1, \\ \phi_{i+1} &:= \phi_i^{M-1} - \alpha_M \nabla_{\phi_i^{M-1}} L_{T_i}\left(\boldsymbol{\tau}_{i,\phi_i^{M-1}}^{1:K}\right) \end{aligned} \tag{7}$$

$$\nabla_{\theta,\alpha}\mathcal{L}_{T_i,T_{i+1}}(\theta,\alpha) = \mathbb{E}_{\substack{\boldsymbol{\tau}_{i,\theta}^{1:K} \sim P_{T_i}(\boldsymbol{\tau}|\theta) \\ \boldsymbol{\tau}_{i+1,\phi} \sim P_{T_{i+1}}(\boldsymbol{\tau}|\phi)}} \left[L_{T_{i+1}}(\boldsymbol{\tau}_{i+1,\phi}) \left[\nabla_{\theta,\alpha} \log \pi_\theta(\boldsymbol{\tau}_{i+1},\phi) + \nabla_\theta \sum_{k=1}^{K} \log \pi_\theta(\boldsymbol{\tau}_{i,\theta}^k) \right] \right] \tag{8}$$

$$\phi_i := \theta - \alpha \frac{1}{K} \sum_{k=1}^{K} \left(\frac{\pi_\theta(\boldsymbol{\tau}^k)}{\pi_{\phi_{i-1}}(\boldsymbol{\tau}^k)} \right) \nabla_\theta L_{T_{i-1}}(\boldsymbol{\tau}^k), \qquad \boldsymbol{\tau}^{1:K} \sim P_{T_{i-1}}(\boldsymbol{\tau}|\phi_{i-1}) \tag{9}$$