Summary of Model Agnostic Meta Learning

Ashutosh Kumar Jha, ME14B148
Mechanical Engineering, IIT Madras
Topics in RL Course

***Abstract*—In this technical report, we summarize the contents and ideas presented in the technical paper titled *Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks*[1]. We also present the possible shortcomings of the work.**

***Keywords*—Deep Reinforcement Learning, Meta-Learning, Few-Shot Learning**

I. Summary

Deep RL algorithms use Convolutional neural networks with classic RL algorithms like Actor-Critic methods [3] and Q-Learning [2] to learn good control policies in high dimensional state spaces. However, all these algorithms are heavily data-intensive and are known to suffer from long training times. This introduces the field of few shot-learning where the goal is to train a agent sufficiently well enough with the help of a small number of examples.

The primary contribution of this work is a simple model and task-agnostic algorithm for meta-learning that learns an initialization of the models parameters such that a small number of gradient updates will lead to fast learning on a new task.

The problem setup is as follows: Given a model f that maps observations **x** to output actions **a**, f is trained to be able to adapt to a large number of tasks. A task $\mathcal{T}$ is defined as $\mathcal{T} = \{\mathcal{L}(x_1, a_1, \cdots, x_H, a_H), q(x_1), q(x_{t+1}|x_t, a_t), H\}$ where $\mathcal{L}$ is a loss function, $q(x_1)$ is the initial distribution over the states, $q(x_{t+1}|x_t, a_t)$ is a transition distribution and H is the episode length.
We consider a distribution over tasks $p(\mathcal{T})$ that we want our model to be able to adapt to. In the K-shot learning setting, the model is trained to learn a new task $\mathcal{T}_i$ drawn from $p(\mathcal{T})$ from only K samples. The meta learner is then trained by using the **test** error on the extra samples.

A. The Intuition of the Algorithm

The basic intuition of the algorithm is to learn features which are more transferable in the transfer-learning setup across different tasks. The algorithm tries to do this by learning model parameters which are **sensitive** to the changes in tasks. This would effectively result in large improvements on the loss of any task.

Mathematically speaking, the adapted parameters for task $\mathcal{T}_i$, θ_i' are given by:

$$\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$$

The meta objective would then become (to maximize sensitivity):

$$\min_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$$

The model parameters θ are thus updated as follows:

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$$

The MAML meta-gradient update involves a gradient through a gradient. Computationally, this requires an additional backward pass through the computational graph to compute Hessian-vector products.

B. Species of MAML

1) Supervised Learning: We define the horizon H = 1 and drop the timestep subscript on x_t , since the model accepts a single input and produces a single output, rather than a sequence. The task $\mathcal{T}_i$ generates K i.i.d. observations **x** from q_i, and the task loss is represented by the error between the models predictions and the corresponding target values **y**. According to the conventional terminology, K-shot classification tasks use K input/output pairs from each class.

2) Reinforcement Learning: Each RL task $\mathcal{T})_i$ contains an initial state distribution $q_i(\mathbf{x}_1)$ and a transition distribution $q_i(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{a}_t)$, and the loss $\mathcal{L}_{\mathcal{T}_i}$ corresponds to the negative of the reward function R. Mathematically the loss for task i is given by:

$$\mathcal{L}_{\mathcal{T}_i}(f_\phi) = -\mathbb{E}_{\mathbf{x}_t, \mathbf{a}_t \sim f_\phi, q_{\mathcal{T}_i}} \left[\sum_{t=1}^{H} R_i(\mathbf{x}_t, \mathbf{a}_t) \right]$$

In K-shot reinforcement learning, K rollouts from f_θ and task $\mathcal{T}_i$, and the corresponding rewards $R(\mathbf{x}_t, \mathbf{a}_t)$, are used for adaptation on a new task $\mathcal{T}_i$.

C. Experiments and Results

1) Regression: Each task involves regressing from the input to the output of a sine wave, where the amplitude and phase of the sinusoid are varied between tasks. The regressor is a neural network model with 2 hidden layers of size 40 with ReLU nonlinearities. K is set to be 10. The first baseline used is a network trained to regress random sinusoid functions and then, at test-time, fine-tuning with gradient descent on the K pro- vided points. The second baseline is an oracle which receives the true amplitude and phase as input.

The learned model is able to quickly adapt with only 5 data points whereas the model that is pretrained using standard supervised learning on all tasks is unable to adequately adapt with so few data points without catastrophic overfitting. More interestingly, the MAML model is able to learn the periodic nature even when all the points in the new task are in one half of the sine-wave.

2) Classification: The problem of N -way classification is set up as follows: select N unseen classes, provide the model with K different instances of each of the N classes, and evaluate the models ability to classify new instances within the N classes. The paper evaluates its performance on the standard Omniglot and MiniImageNet datasets. The MAML model outperforms the current SOTA models by a narrow margin on 1 and 5 shot classification tasks on both the datasets.

The authors also show that upon using a first-order approximation in the algorithm (which originally requires computing Hessian products), the performance is not affected significantly, suggesting that most of the improvement comes from the gradients of the objective at the post-update parameter values, rather than the second order updates from differentiating through the gradient update.

3) Reinforcement Learning: The paper evaluates the MAML algorithm on tasks from rllab. In all of the domains, the model trained by MAML is a neural network policy with two hidden layers of size 100, with ReLU nonlinearities. The gradient updates are computed using vanilla policy gradient and the meta-optimizer is selected as TRPO. Three baselines are considered:

1) Pretraining one policy on all of the tasks and then fine-tuning.
2) Training a policy from randomly initialized weights.
3) An oracle policy which receives the parameters of the task as input.

The various tasks here correspond to different goals in **the same environment**.

In the 2D navigation environment, where the goal is a position, the MAML algorithm beats all baselines except the oracle(obviously).

In the 3D Locomotion environment from MuJoCo, the goal is a velocity that has to be obtained by the agent. Again, the MAML algorithm beats all the baselines except the oracle.

The interesting part of the results is the sheer number of updates it takes for MAML to adapt to new tasks. It is seen that MAML learns a model that can quickly adapt its velocity and direction with even just a single gradient update, and continues to improve with more gradient steps.

II. Shortcomings

In this section we make a note of possible short comings of this paper:

- The loss for the task in the RL setting assumes that the discount factor is 1 which is a strong assumption to make in the case the horizon is infinite.
- While the task specific update intuitively makes sense, the fact that the algorithm again samples tasks upon making the global update doesn't really make sense.
- I don't really like the definition of ”tasks” in the RL setting. I would rather see results in true multitasking case where the tasks would correspond to two different environments.

III. Conclusion

The paper presents an interesting model-independent framework for few shot learning and fast adaptation. The paper is also the current SOTA on few shot classification, thus strengthening its prowess. In case of Reinforcement Learning, while the experimental results on multiple tasks with similar environments is convincing, further work with networks which can generalize across different environments would strengthen the paper.

References

[1] Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks, Chelsea Finn, Pieter Abbeel, Sergey Levine. In International Conference in Machine Learning (ICML), 2017.

[2] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, Demis Hassabis. ”Human-level control through deep reinforcement learning”, in Nature 518, 529533 , 2015.

[3] Volodymyr Mnih, Adri Puigdomnech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, Koray Kavukcuoglu.”Asynchronous Methods for Deep Reinforcement Learning”, in ICML, 2016.