Model-Agnostic Meta-Learning for Fast Adaption of Deep Networks

PMLR 2017

Chelsea Finn, Pieter Abeel, Sergey Levine

1. Introduction

✓**빠르게 학습**하는 것은 Human Intelligence의 특징

- 소량의 Example을 보고 물체(Object)를 인식
- 짧은 시간의 경험을 통해 새로운 기술(Skills)을 학습

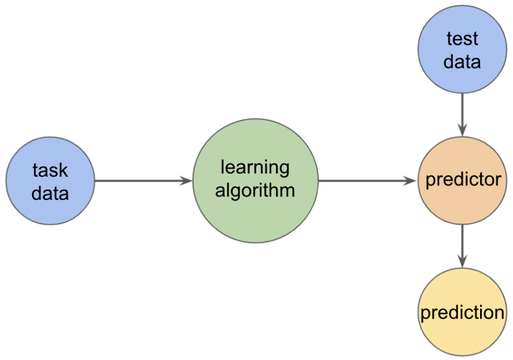
✓Artificial Agent도 소량의 Example을 통해 **빠르게 학습하고 적응하는 능력**이 필요함

- 하지만, 새로운 정보를 이전의 경험과 연관시키면서, 새로운 데이터에 대한 Overfitting 방지가 힘듦

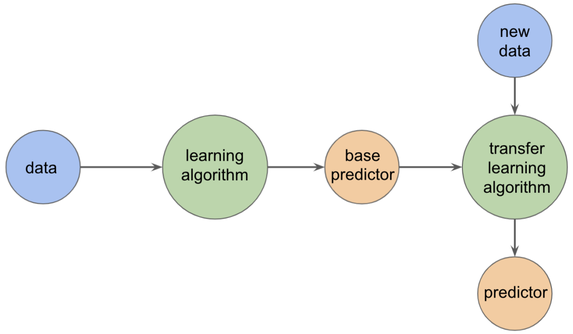
1. Introduction

Meta-Learning

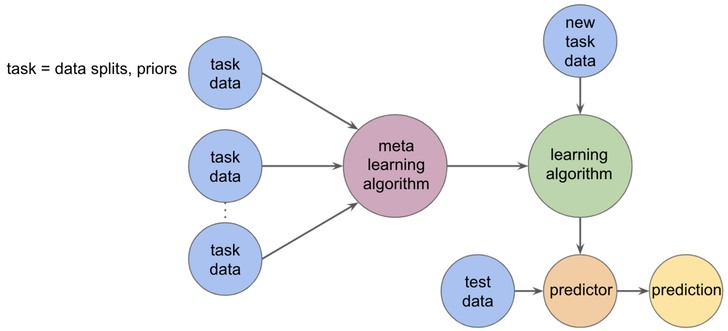
- 학습된 모델은 소량의 새로운 데이터를 통해 새로운 작업(Task)를 빠르게 학습하는 것이 가능해야 함
- Meta-Learner를 통해 학습되는 모델은 많은 다른 작업에서 학습이 가능해야 함



Supervised Learning

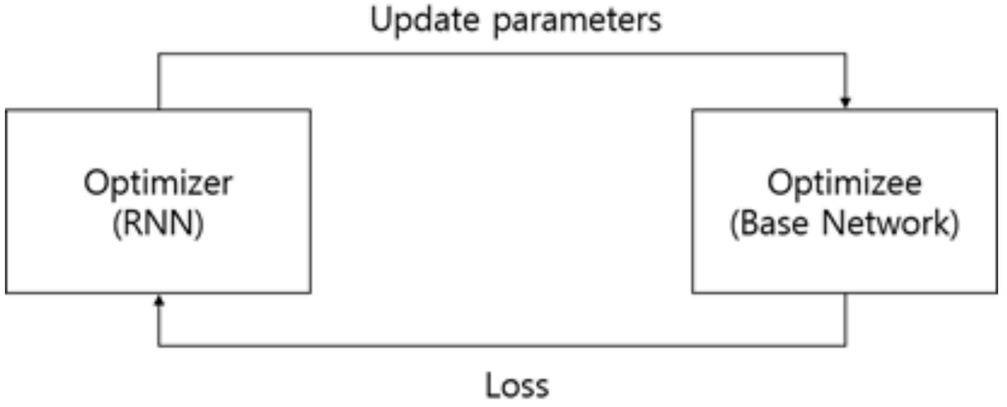


Transfer Learning



Meta Learning

1. Introduction



Model-Agnostic Meta-Learning (MAML)

-기존의 Meta-Learning 방법과 다르게 학습 가능한 Parameter 수 증가 및 Architecture에 대한 제한 X

-새로운 작업에 대한 소량의 데이터에서 A Few Gradient Step을 통해 Parameter를 업데이트

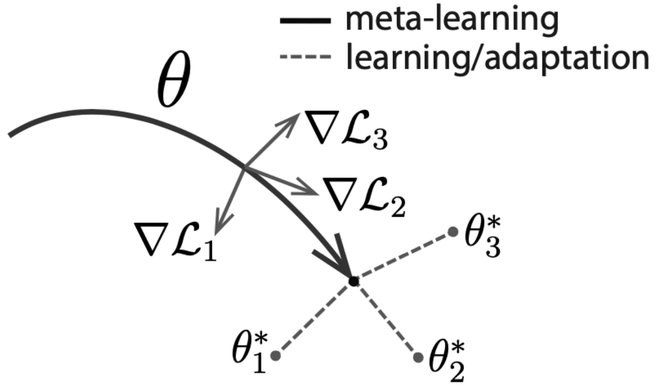
-많은 작업에 고르게 적합한 Internal Representation을 만드는 것으로 생각할 수 있음

-Internal Representation이 많은 작업에 적합하면, 약간의 모델 Fine-Turning은 좋은 결과를 얻을 수 있음

Model-Agnostic: 어떠한 학습 문제와 모델에도 직접적으로 적용 가능함

문제(Problem): Classification, Regression, Reinforcement Learning

모델(Model): Trained with a Gradient Descent Procedure



2. MAML - Meta Learning Problem Set-Up

Few-Shot Meta-Learning의 목표

- A Few Datapoint와 Training Iteration을 통해 새로운 작업에 모델이 빠르게 적응하도록 학습하는 것

Few-Shot Meta-Learning

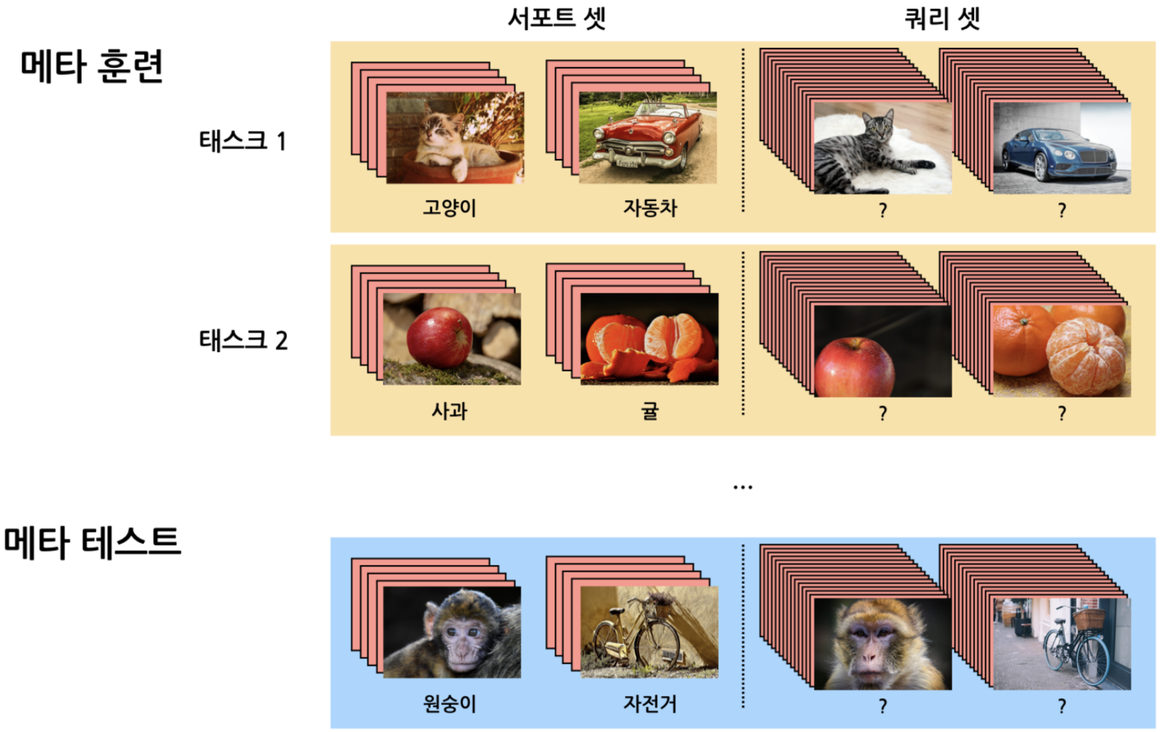
- Meta-Learning 동안, Model 또는 Learner는 일련의 작업을 통해 학습되어짐
- 학습된 모델은 소량의 Example 또는 Trial을 통해 빠르게 새로운 작업에 적응 가능해짐
- 사실상, Meta-Learning Problem은 전체 작업(Task)을 Training Example로써 다룸

2. MAML - Meta Learning Problem Set-Up

- Model f
- Observation x
- Outputs a
- Task

$$T = \{L(x_1, a_1, \ldots, x_H, a_H), q(x_1), q(x_{t+1} \mid x_t, a_t), H\}$$

- Loss Function L
- Initial Observations $q(x_1)$
- Transition Distribution $q(x_{t+1} \mid x_t, a_t)$
- Episode Length H

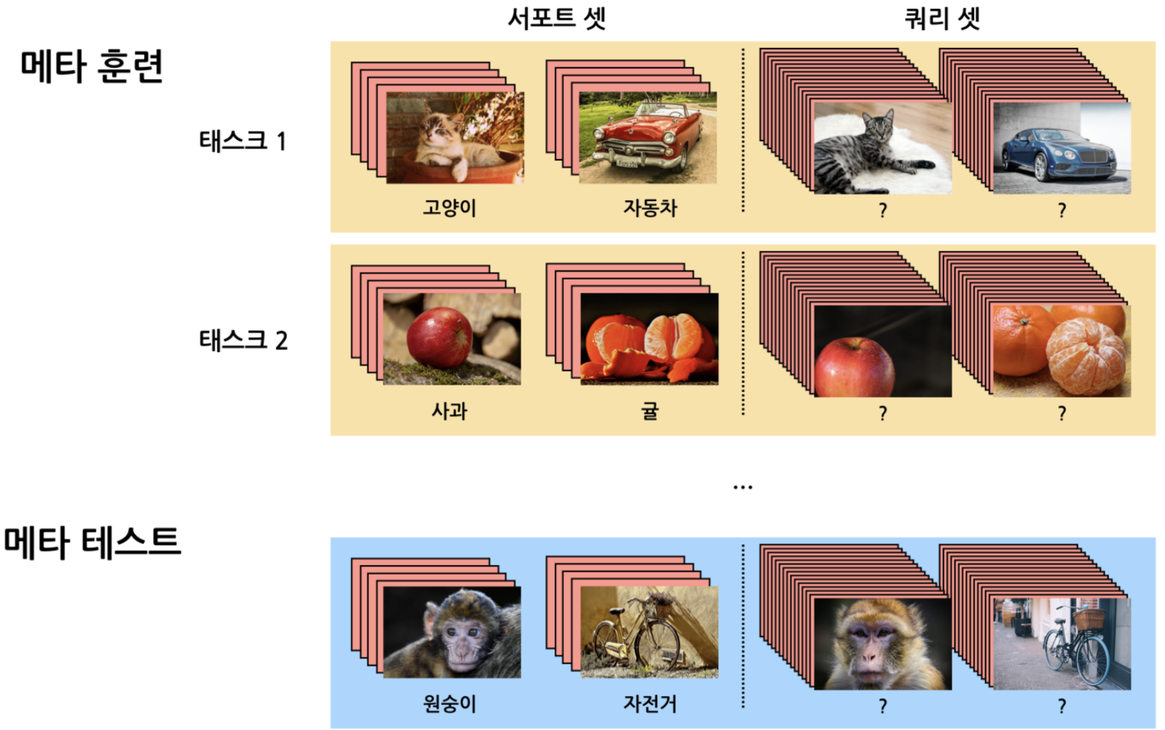


2. MAML - Meta Learning Problem Set-Up

✓모델이 작업들에 적응되기 위해
작업에 대한 분포 $p(T)$ 고려

✓K-Shot Learning 설정에서의 모델 학습

- $p(T)$에서 그려진 새로운 작업 T_i
- q_i에서 그려진 K개의 샘플
- T_i에서 생성된 Feedback L_T
- T_i에서 새로운 샘플로 테스트



2. MAML - Meta Learning Problem Set-Up

✓모델 f 는 Parameter에 대해서 새로운 샘플에 대한 테스트 에러를 고려하여 개선됨

- 사실상, 샘플된 작업 T_i의 테스트 에러는 Meta-Learning 과정의 학습 에러로서 고려되어짐
- Meta-Learning의 마지막에서, 새로운 작업은 $p(T)$에서 그려짐
- K개의 샘플로 학습된 이후, 모델 성능이 측정됨

2. MAML - Algorithm

- Parametr θ를 가진 Model f_θ
- 새로운 작업 T_i 에 모델이 적응됐을 때 $f_{\theta_i'}$

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

- Meta Objective

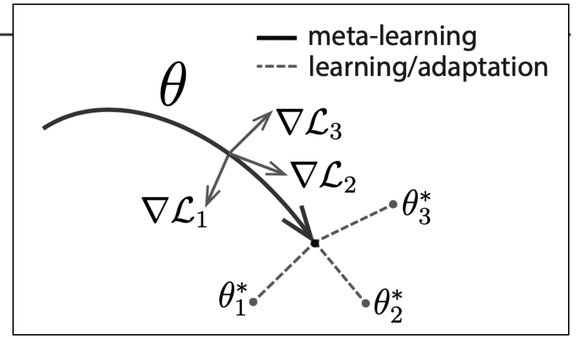
$$\min_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'}) = \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)})$$

- Meta Optimization: SGD

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

Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to K examples
6: Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
7: **end for**
8: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$
9: **end while**



2. MAML - Algorithm

Algorithm 2 MAML for Few-Shot Supervised Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K datapoints $\mathcal{D} = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ from $\mathcal{T}_i$
6: Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ using $\mathcal{D}$ and $\mathcal{L}_{\mathcal{T}_i}$ in Equation (2) or (3)
7: Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
8: Sample datapoints $\mathcal{D}_i' = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ from $\mathcal{T}_i$ for the meta-update
9: **end for**
10: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$ using each $\mathcal{D}_i'$ and $\mathcal{L}_{\mathcal{T}_i}$ in Equation 2 or 3
11: **end while**

Algorithm 3 MAML for Reinforcement Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K trajectories $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{a}_1, ...\mathbf{x}_H)\}$ using f_θ in $\mathcal{T}_i$
6: Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ using $\mathcal{D}$ and $\mathcal{L}_{\mathcal{T}_i}$ in Equation 4
7: Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
8: Sample trajectories $\mathcal{D}_i' = \{(\mathbf{x}_1, \mathbf{a}_1, ...\mathbf{x}_H)\}$ using $f_{\theta_i'}$ in $\mathcal{T}_i$
9: **end for**
10: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$ using each $\mathcal{D}_i'$ and $\mathcal{L}_{\mathcal{T}_i}$ in Equation 4
11: **end while**

3. Experimental Evaluation

✓MAML은 새로운 작업의 빠른 학습을 가능하게 하는가?

✓MAML은 여러 다른 도메인에 사용가능한 Meta-Learning 방법인가?

✓MAML을 통해 학습된 모델은 추가 Gradient Update and/or Example로 계속 개선이 가능한가?

3. Experimental Evaluation

Regression: Sine Wave

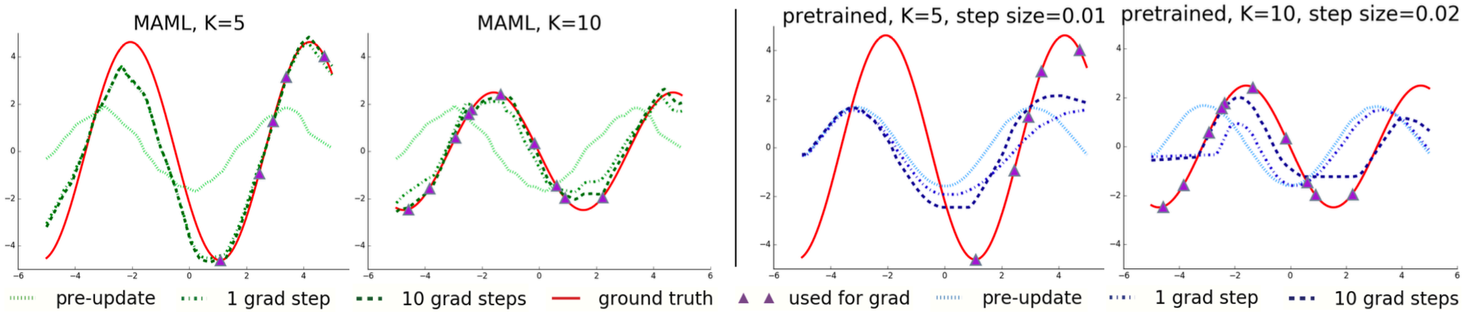


Figure 2. Few-shot adaptation for the simple regression task. Left: Note that MAML is able to estimate parts of the curve where there are no datapoints, indicating that the model has learned about the periodic structure of sine waves. Right: Fine-tuning of a model pretrained on the same distribution of tasks without MAML, with a tuned step size. Due to the often contradictory outputs on the pre-training tasks, this model is unable to recover a suitable representation and fails to extrapolate from the small number of test-time samples.

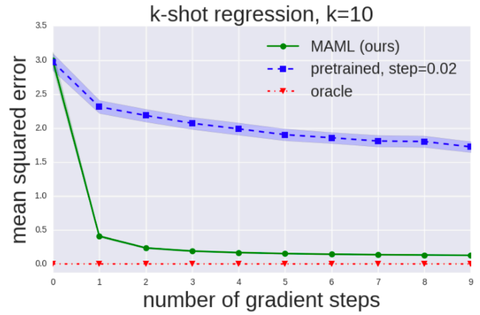


Figure 3. Quantitative sinusoid regression results showing the learning curve at meta test-time. Note that MAML continues to improve with additional gradient steps without overfitting to the extremely small dataset during meta-testing, achieving a loss that is substantially lower than the baseline fine-tuning approach.

3. Experimental Evaluation

Classification: Omniglot and MiniImagenet

Table 1. Few-shot classification on held-out Omniglot characters (top) and the MiniImagenet test set (bottom). MAML achieves results that are comparable to or outperform state-of-the-art convolutional and recurrent models. Siamese nets, matching nets, and the memory module approaches are all specific to classification, and are not directly applicable to regression or RL scenarios. The $\pm$ shows 95% confidence intervals over tasks. Note that the Omniglot results may not be strictly comparable since the train/test splits used in the prior work were not available. The MiniImagenet evaluation of baseline methods and matching networks is from Ravi & Larochelle (2017).

	5-way Accuracy		20-way Accuracy	
Omniglot (Lake et al., 2011)	1-shot	5-shot	1-shot	5-shot
MANN, no conv (Santoro et al., 2016)	82.8%	94.9%	–	–
MAML, no conv (ours)	**89.7 ± 1.1%**	**97.5 ± 0.6%**	–	–
Siamese nets (Koch, 2015)	97.3%	98.4%	88.2%	97.0%
matching nets (Vinyals et al., 2016)	98.1%	98.9%	93.8%	98.5%
neural statistician (Edwards & Storkey, 2017)	98.1%	99.5%	93.2%	98.1%
memory mod. (Kaiser et al., 2017)	98.4%	99.6%	95.0%	98.6%
MAML (ours)	**98.7 ± 0.4%**	**99.9 ± 0.1%**	**95.8 ± 0.3%**	**98.9 ± 0.2%**

	5-way Accuracy	
MiniImagenet (Ravi & Larochelle, 2017)	1-shot	5-shot
fine-tuning baseline	28.86 ± 0.54%	49.79 ± 0.79%
nearest neighbor baseline	41.08 ± 0.70%	51.04 ± 0.65%
matching nets (Vinyals et al., 2016)	43.56 ± 0.84%	55.31 ± 0.73%
meta-learner LSTM (Ravi & Larochelle, 2017)	43.44 ± 0.77%	60.60 ± 0.71%
MAML, first order approx. (ours)	**48.07 ± 1.75%**	**63.15 ± 0.91%**
MAML (ours)	**48.70 ± 1.84%**	**63.11 ± 0.92%**

3. Experimental Evaluation

Reinforcement Learning: 2D Navigation and Locomotion

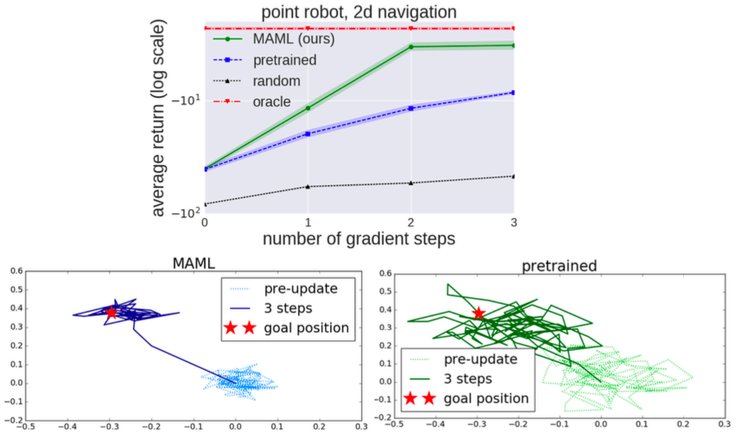


Figure 4. Top: quantitative results from 2D navigation task, Bottom: qualitative comparison between model learned with MAML and with fine-tuning from a pretrained network.

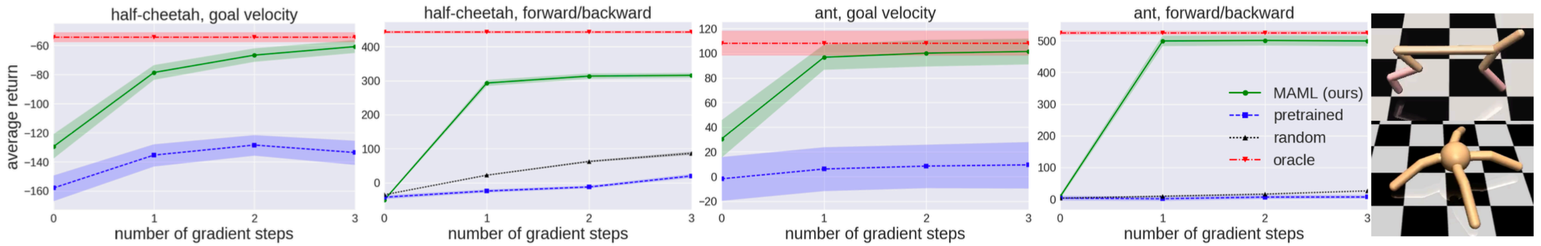


Figure 5. Reinforcement learning results for the half-cheetah and ant locomotion tasks, with the tasks shown on the far right. Each gradient step requires additional samples from the environment, unlike the supervised learning tasks. The results show that MAML can adapt to new goal velocities and directions substantially faster than conventional pretraining or random initialization, achieving good performs in just two or three gradient steps. We exclude the goal velocity, random baseline curves, since the returns are much worse (< -200 for cheetah and < -25 for ant).