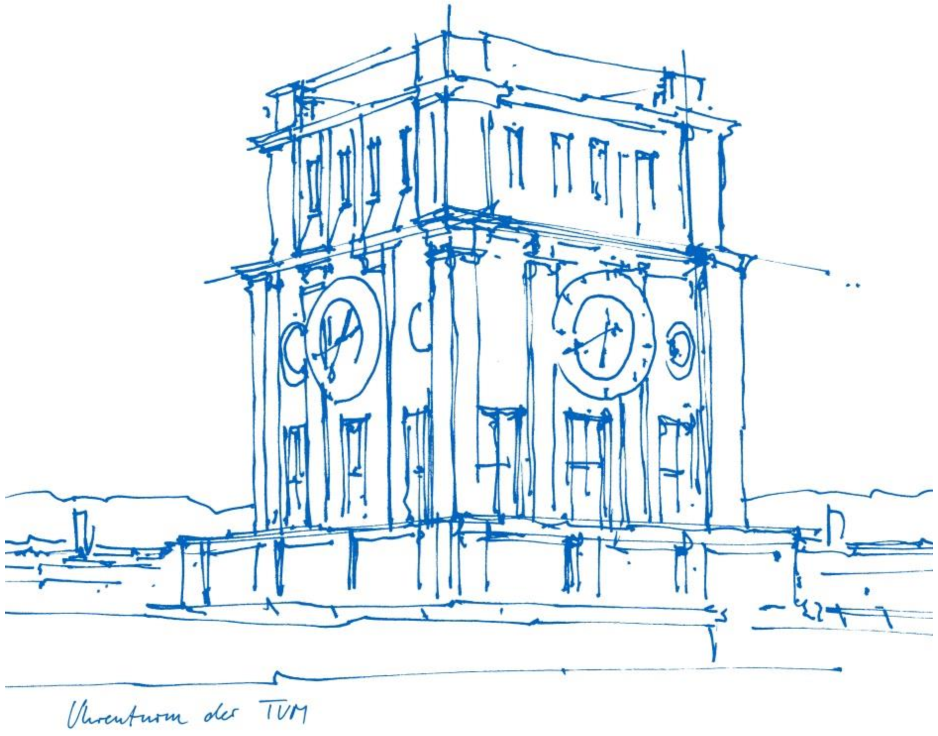
Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks

Chelsea Finn, Pieter Abbeel, and Sergey Levine. *International Conference on Machine Learning*. PMLR, 2017.

Orcun Cetintas
Recent Trends in Automated Machine Learning
Technical University of Munich
July 7th, 2021

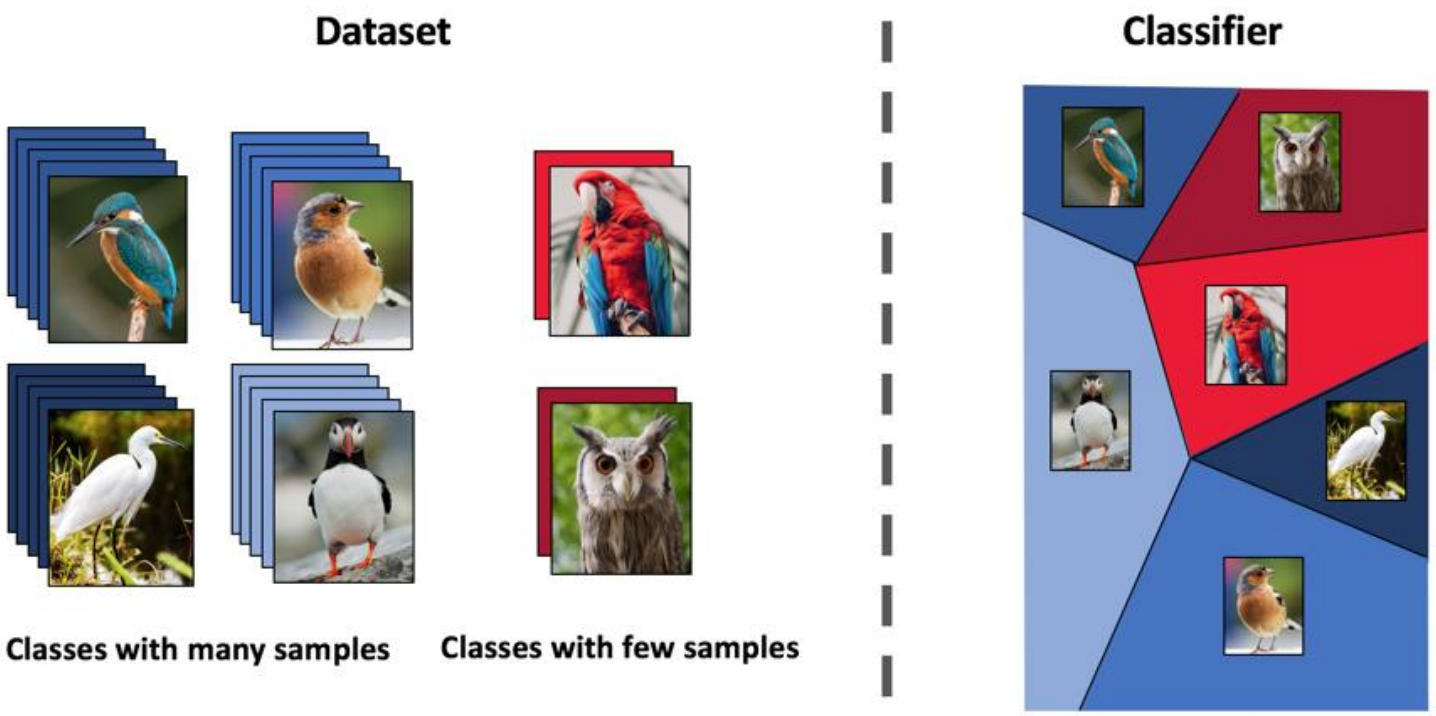


Meta-Learning

Problem: Deep learning is successful with a large amount of data, but often data is scarce

Solution: Use data from other tasks to **learn how to learn** → **Rapid adaptation** on the new task

Few-Shot Learning



https://medium.com/sap-machine-learning-research/deep-few-shot-learning-a1caa289f18

Generalizing to a new task using "few" samples and prior knowledge

One-Shot Video Object Segmentation [1]

First frame

Ground truth

Test frame

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

Chelsea Finn, Pieter Abbeel, and Sergey Levine. *International Conference on Machine Learning*. PMLR, 2017.

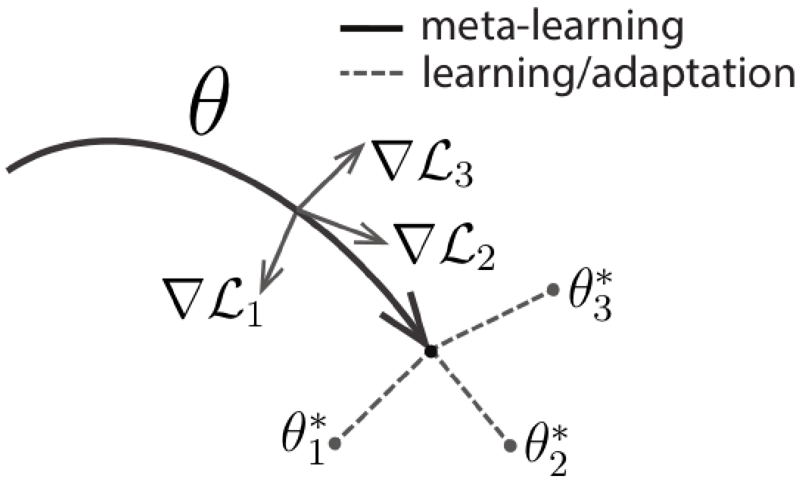
Paper's aim: learning a ***model initialization*** that can achieve ***rapid adaptation***

Paper proposes: an algorithm for meta-learning

- model-agnostic
- applicable to different learning problems

Model-Agnostic Meta-Learning (MAML) - Overview

Find model parameters that are **sensitive to changes** in the task



Model-Agnostic Meta-Learning (MAML) - Overview

Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters

randomly initialize θ
while not done **do**
    Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
    for all $\mathcal{T}_i$ **do**
        Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to K examples
        Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
    end for
    Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$
end while

Outer Loop

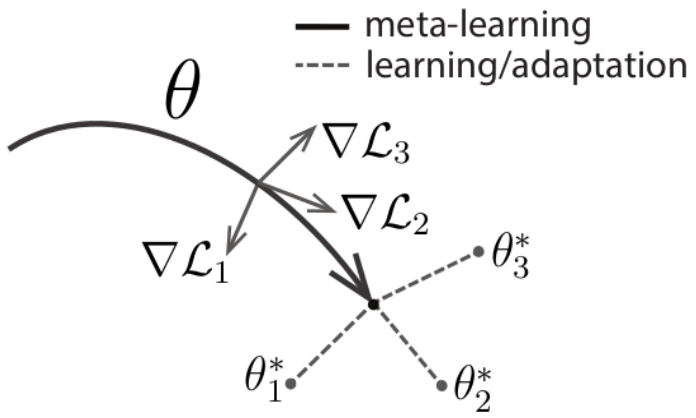
Inner Loop

MAML - Overview

Inner Loop: Update the model for a task from an initialization

Outer Loop: Optimize for the performance of all **inner loop** models on **all tasks**

Intuition: We want achieve a low loss after only a few updates on a task



MAML - Notation

- model f_θ with parameters θ
- distribution over tasks $p(\mathcal{T})$
- sampled task $\mathcal{T}_i$
- task loss $\mathcal{L}_{\mathcal{T}_i}$

MAML - Algorithm

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 $\mathcal{T}_i \sim p(\mathcal{T})$

MAML - Inner Loop

Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters

randomly initialize θ
while not done **do**
  Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
  for all $\mathcal{T}_i$ **do**
    Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to K examples
    Compute adapted parameters with gradient descent: $\theta'_i = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
  end for

Inner Loop: Update the model for a task from an initialization

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

Simple gradient update on the sampled task

MAML – Outer Loop

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**

Outer Loop: Optimize for the performance of all inner loop models on **all tasks**

MAML – Outer Loop

Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters

```
randomly initialize θ
while not done do
    Sample batch of tasks T_i ∼ p(T)
    for all T_i do
        Evaluate ∇_θ L_{T_i}(f_θ) with respect to K examples
        Compute adapted parameters with gradient descent: θ'_i = θ − α∇_θ L_{T_i}(f_θ)
    end for
    Update θ ← θ − β∇_θ Σ_{T_i∼p(T)} L_{T_i}(f_{θ'_i})
end while
```

Meta-objective:

$$\underbrace{\min_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta'_i})}_{\text{Total loss of all updated models}} = \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)})$$

Meta-update:

$$\theta \leftarrow \theta - \beta \nabla_\theta \underbrace{\sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta'_i})}_{\text{Total loss of all updated models}}$$

MAML for Few-Shot Supervised Learning

Regression: predict the outputs of a function from **only K datapoints** sampled from that function, after training on many functions with similar statistical properties

Classification: learn to classify an object **only from K examples**, after training on many other types of objects

How to use MAML?

- Simply use the general framework with appropriate **loss functions**!

MAML for Few-Shot Supervised Learning

Algorithm 2 MAML for Few-Shot Supervised Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters

randomly initialize θ
while not done **do**
  Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
  for all $\mathcal{T}_i$ **do**
    Sample K datapoints $\mathcal{D} = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ from $\mathcal{T}_i$
    Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ using $\mathcal{D}$ and $\mathcal{L}_{\mathcal{T}_i}$ in Equation (2) or (3)
    Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
    Sample datapoints $\mathcal{D}_i' = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ from $\mathcal{T}_i$ for the meta-update
  end for
  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
end while

Regression: Mean-squared error (MSE)

$$\mathcal{L}_{\mathcal{T}_i}(f_\phi) = \sum_{\mathbf{x}^{(j)},\mathbf{y}^{(j)} \sim \mathcal{T}_i} \| f_\phi(\mathbf{x}^{(j)}) - \mathbf{y}^{(j)} \|_2^2, \qquad (2)$$

Classification: Cross-entropy loss

$$\mathcal{L}_{\mathcal{T}_i}(f_\phi) = \sum_{\mathbf{x}^{(j)},\mathbf{y}^{(j)} \sim \mathcal{T}_i} \mathbf{y}^{(j)} \log f_\phi(\mathbf{x}^{(j)}) + (1 - \mathbf{y}^{(j)}) \log(1 - f_\phi(\mathbf{x}^{(j)})) \qquad (3)$$

MAML for Reinforcement Learning

Goal: enable an agent to **quickly acquire a new task policy** using only a small amount of experience

How to use MAML?

- Use **policy gradient method** for a differentiable framework
- **Sample new examples with the new policy**

MAML for Reinforcement Learning

Algorithm 3 MAML for Reinforcement Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters

```
randomly initialize θ
while not done do
    Sample batch of tasks T_i ∼ p(T)
    for all T_i do
        Sample K trajectories D = {(x_1, a_1, ...x_H)} using f_θ in T_i
        Evaluate ∇_θ L_{T_i}(f_θ) using D and L_{T_i} in Equation 4
        Compute adapted parameters with gradient descent: θ'_i = θ − α∇_θ L_{T_i}(f_θ)
        Sample trajectories D'_i = {(x_1, a_1, ...x_H)} using f_{θ'_i} in T_i
    end for
    Update θ ← θ − β∇_θ Σ_{T_i∼p(T)} L_{T_i}(f_{θ'_i}) using each D'_i and L_{T_i} in Equation 4
end while
```

Loss: Negative expected reward

$$\mathcal{L}_{\mathcal{T}_i}(f_\phi) = -\mathbb{E}_{\mathbf{x}_t, \mathbf{a}_t \sim f_\phi, q_{\mathcal{T}_i}} \left[\sum_{t=1}^{H} R_i(\mathbf{x}_t, \mathbf{a}_t) \right]. \quad (4)$$

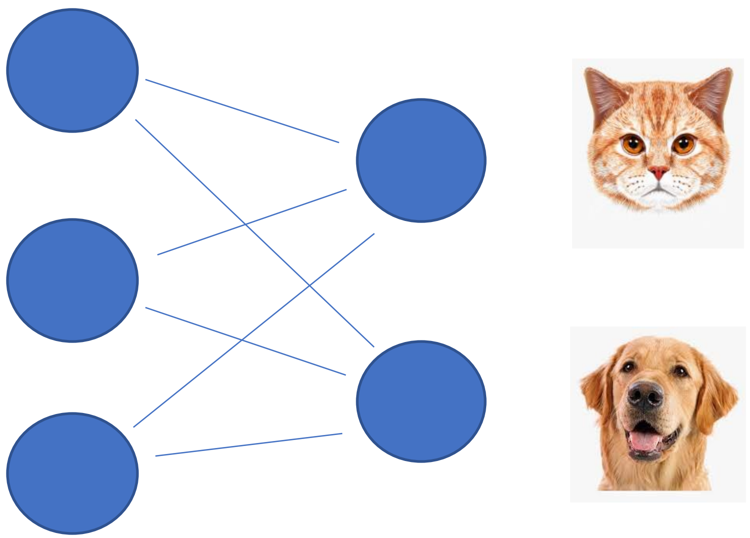
MAML – Task Overfitting and Memorization

Task overfitting: Model aligns too closely to a task and fails to generalize

Memorization problem: Meta-learner **memorizes the meta-training tasks** rather than **learning to adapt**

Example: Instead of ***learning to classify cats***, we want to ***learn to rapidly adapting to classify cats***

Solution: Per-task random assignment of image classes to N-way classification labels



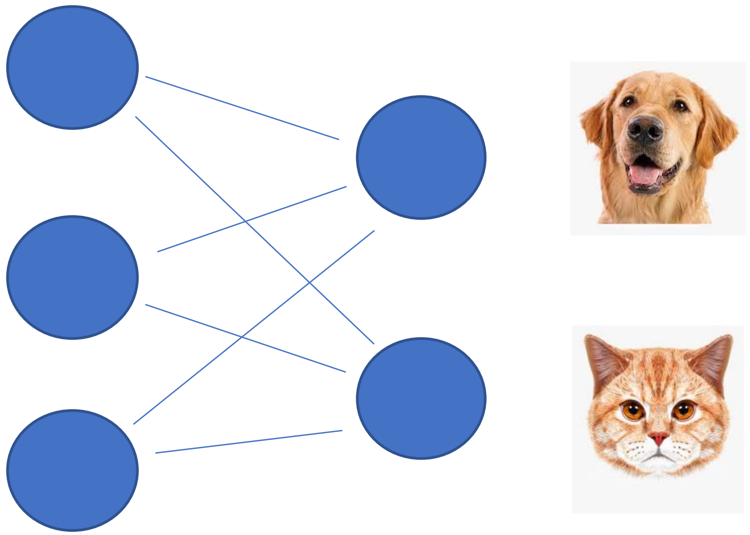
MAML – Task Overfitting and Memorization

Task overfitting: Model aligns too closely to a task and fails to generalize

Memorization problem: Meta-learner **memorizes the meta-training tasks** rather than **learning to adapt**

Example: Instead of ***learning to classify cats***, we want to ***learn to rapidly adapting to classify cats***

Solution: Per-task random assignment of image classes to N-way classification labels



Experiments – Main Questions

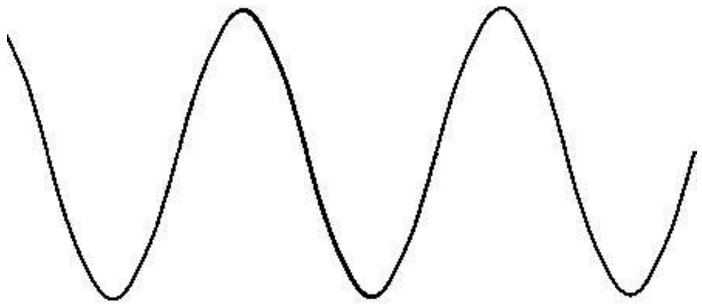
- Can MAML enable **fast learning** of new tasks?
- Can MAML be used for meta-learning in **multiple different domains**?
- Can MAML models **continue to improve** with additional gradient updates?

Experiments - Regression

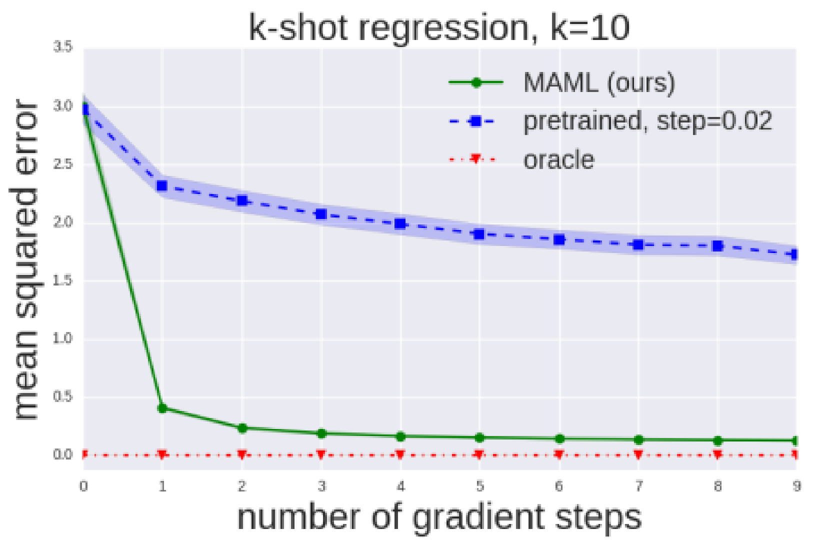
Task: Regressing to a sine wave (varying amplitude and phase) given K data points

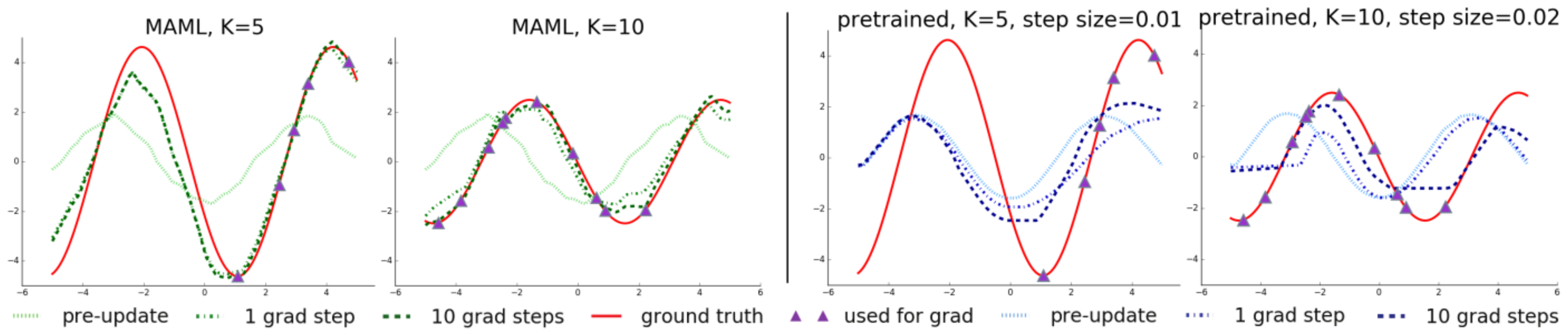
MAML: Meta-training on all tasks with MAML + fine-tuning on K data points

Baseline: Pretraining on all tasks with SGD + fine-tuning on K data points



Experiments - Regression





Experiments - Classification

Task: Few shot classification of N unseen classes with only K instances

- Handwritten character classification on **Omniglot**
  - 20 instances of 1623 chars from 50 alphabets

- Image classification on **MiniImagenet**
  - 64 train, 24 val, 12 test classes

Experiments - Classification

Omniglot (Lake et al., 2011)	5-way Accuracy 1-shot	5-way Accuracy 5-shot	20-way Accuracy 1-shot	20-way Accuracy 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%**

MiniImagenet (Ravi & Larochelle, 2017)	5-way Accuracy 1-shot	5-way Accuracy 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%**

Experiments - Classification

Omniglot (Lake et al., 2011)	5-way Accuracy 1-shot	5-way Accuracy 5-shot	20-way Accuracy 1-shot	20-way Accuracy 5-shot
MANN, no conv (Santoro et al., 2016)	82.8%	94.9%	–	–
MAML, no conv (ours)	**89.7** $\pm$ **1.1%**	**97.5** $\pm$ **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** $\pm$ **0.4%**	**99.9** $\pm$ **0.1%**	**95.8** $\pm$ **0.3%**	**98.9** $\pm$ **0.2%**

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

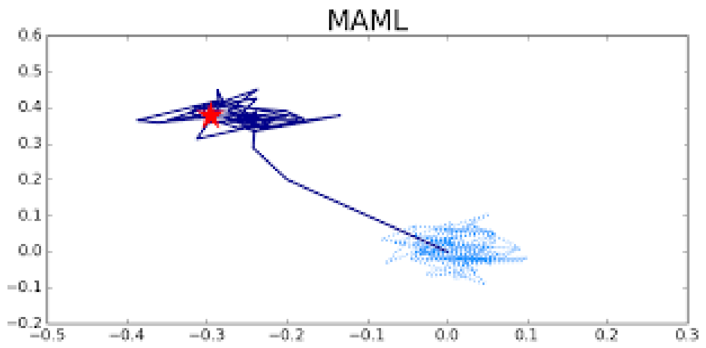
Experiments - Reinforcement Learning

Task: 2D Navigation - move the point agent to a goal

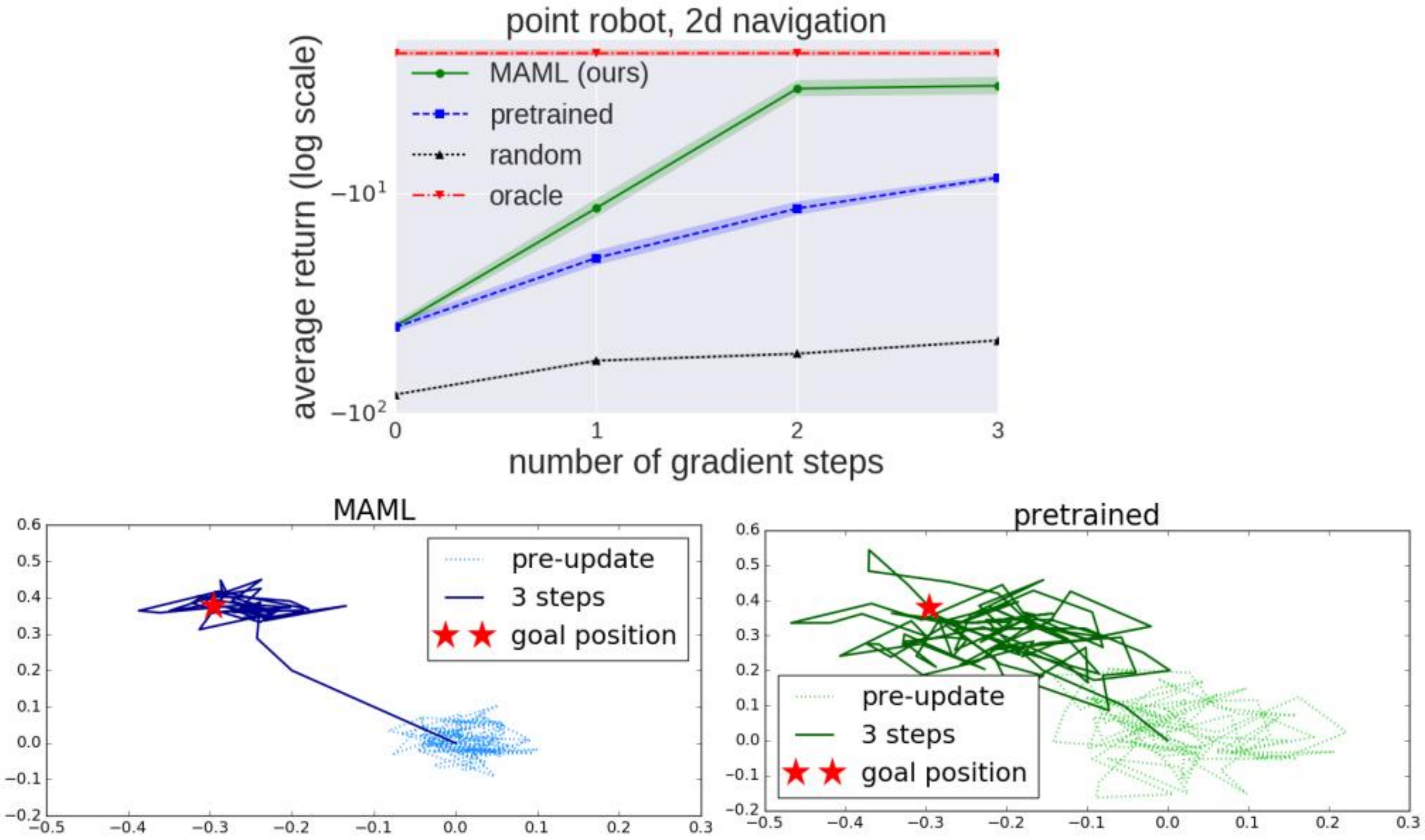
MAML: Meta-training a policy on all tasks with MAML + fine-tuning

Baseline 1 (pretrained): Pretraining a policy on all tasks + fine-tuning

Baseline 2 (random): Training a policy from scratch



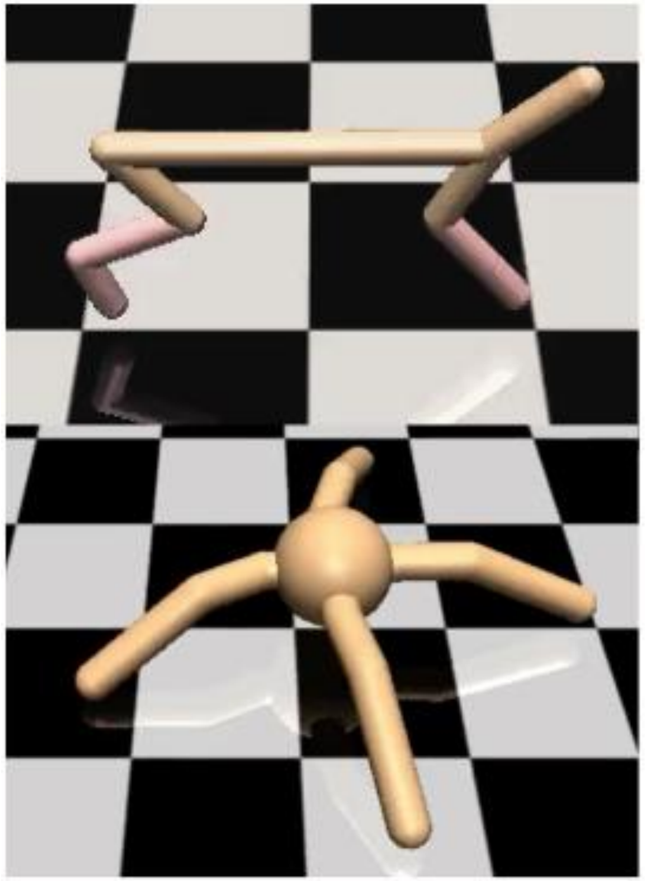
Experiments - Reinforcement Learning



Experiments - Reinforcement Learning

Tasks: Locomotion in MuJoCo [2] with two robots (cheetah and ant)

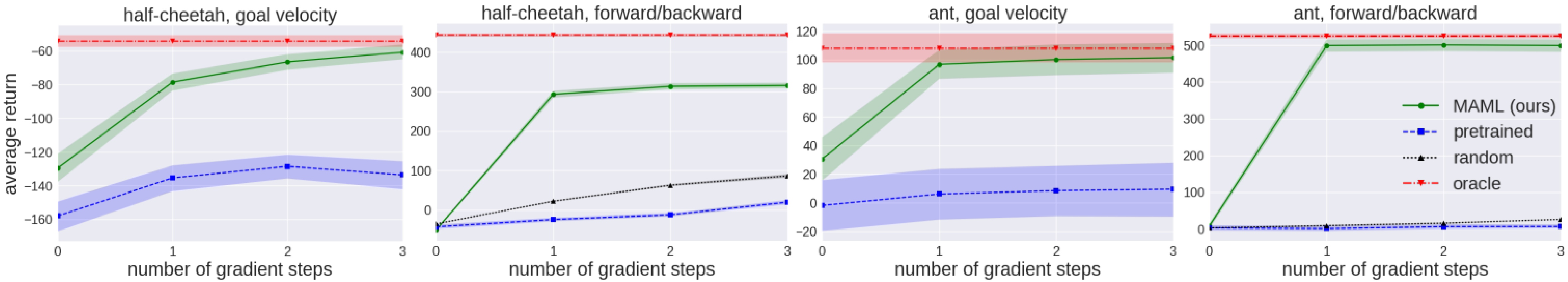
- Run in a particular direction
- Run at a particular velocity



Baseline 1 (pretrained): Pretraining a policy on all tasks + fine-tuning

Baseline 2 (random): Training a policy from scratch

Experiments - Reinforcement Learning



Experiments – Main Questions

- Can MAML enable **fast learning** of new tasks? **YES!**
- Can MAML be used for meta-learning in **multiple different domains**? **YES!**
- Can MAML models **continue to improve** with additional gradient updates? **Yes, but further exploration is required**
- **MAML beats the baselines and achieves the SotA**

Discussion

MAML: a model-agnostic meta-learning method based on gradient descent

Pros:

- Model agnostic
- Only requirement is a differentiable task
- No extra parameters
- Step towards general-purpose meta-learning

Cons:

- Learning rate's influence
- Computationally costly
- Hard to train

- **Reptile [3]:** Proposes a new algorithm with only first-order derivatives

- **MAML++ [4]:** Stabilizes MAML training and proposes improvements such as learning the learning rate

- **Meta-SGD [5]:** Learns all components of a meta-optimizer (initialization, update direction and learning rate)

References

[1] Caelles, Sergi, et al. "One-shot video object segmentation." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.

[2] Todorov, Emanuel, Tom Erez, and Yuval Tassa. "Mujoco: A physics engine for model-based control." *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012.

[3] Nichol, Alex, Joshua Achiam, and John Schulman. "On first-order meta-learning algorithms." *arXiv preprint arXiv:1803.02999* (2018).

[4] Antoniou, Antreas, Harrison Edwards, and Amos Storkey. "How to train your maml." *arXiv preprint arXiv:1810.09502* (2018).

[5] Li, Zhenguo, et al. "Meta-sgd: Learning to learn quickly for few-shot learning." *arXiv preprint arXiv:1707.09835* (2017).

Discussion

MAML: a model-agnostic meta-learning method based on gradient descent

Pros:

- Model agnostic
- Only requirement is a differentiable task
- No extra parameters
- Step towards general-purpose meta-learning

Cons:

- Learning rate's influence
- Computationally costly
- Hard to train