Bayesian Model Agnostic Meta Learning

Taesup Kim, Jaesik Yoon, Ousmane Dia, Sungwoon Kim,
Yoshua Bengio and Sungjin Ahn
NIPS 2018

19. 02. 20
Mingyu Kim

*** This material quoted Junyoung Yi's material**

Introduction to meta-learning

Thanks to

Chapter 1 and 2 is referred from the Junyoung Yi's material, which describe MAML and its background.
As below, you are able to be accessible to this material

https://www.slideshare.net/ssuser62b35f/introduction-to-maml-model-agnostic-meta-learning-with-discussions-124492943?fbclid=IwAR1DmX0PkyEb68nniKH49AWAXqEfr40YyLxlNDpqco1dUVHmvP_O_tHk5n0

Meta learning (a.k.a Few shots learning / Learning to learn)

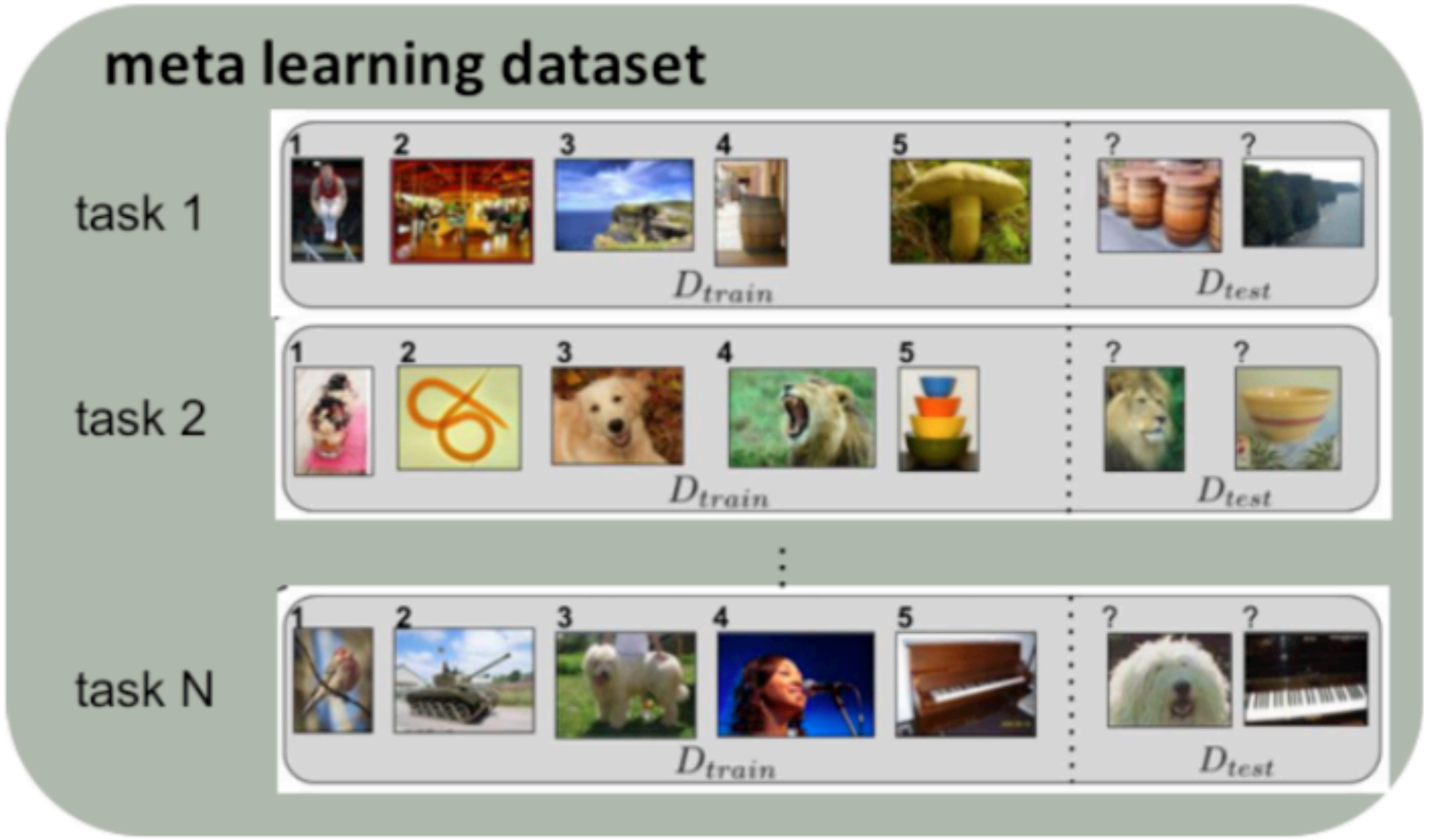
Definition

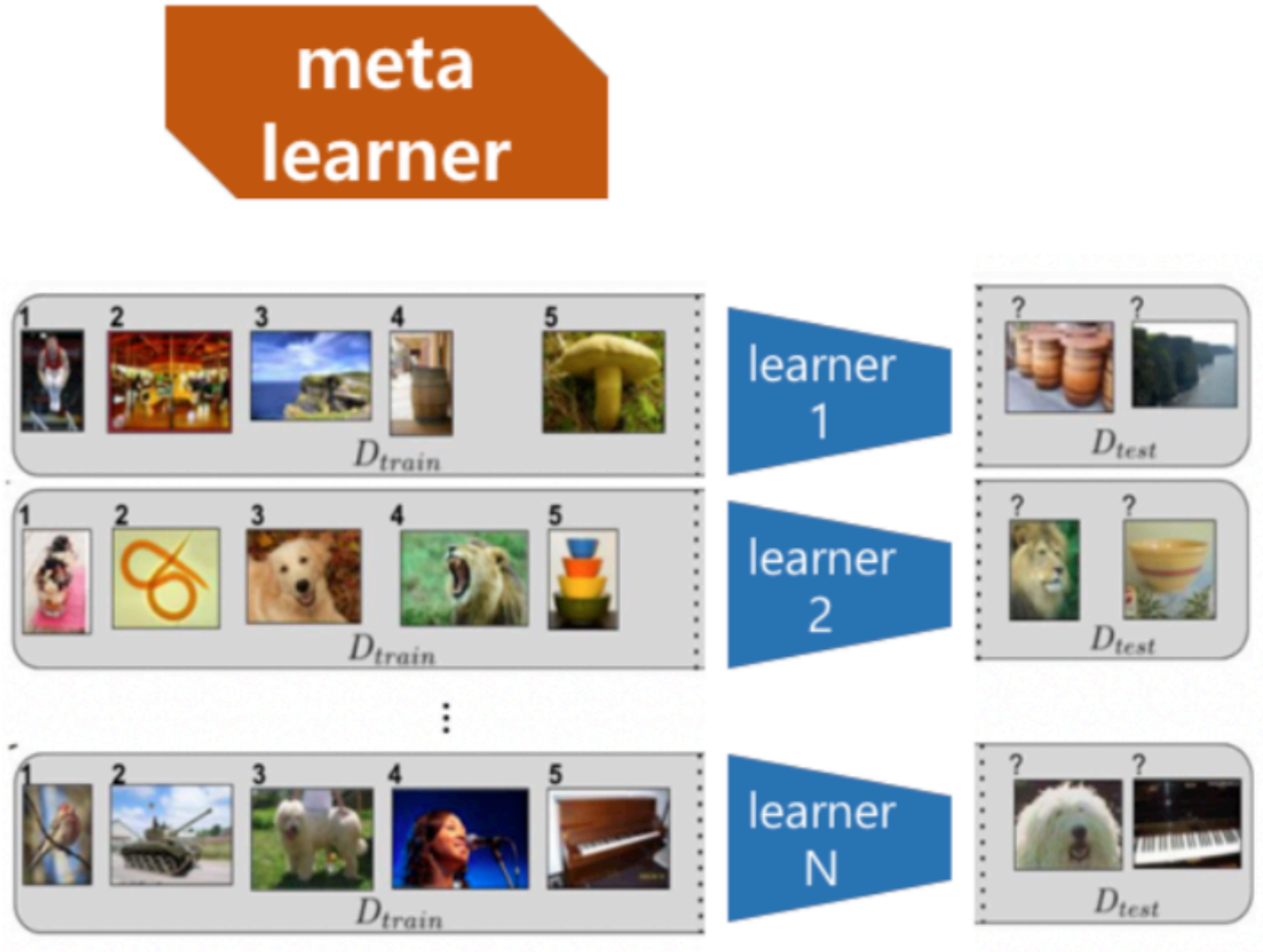
Learner is trained by meta-learner to be able to learn on many different tasks.

Goal

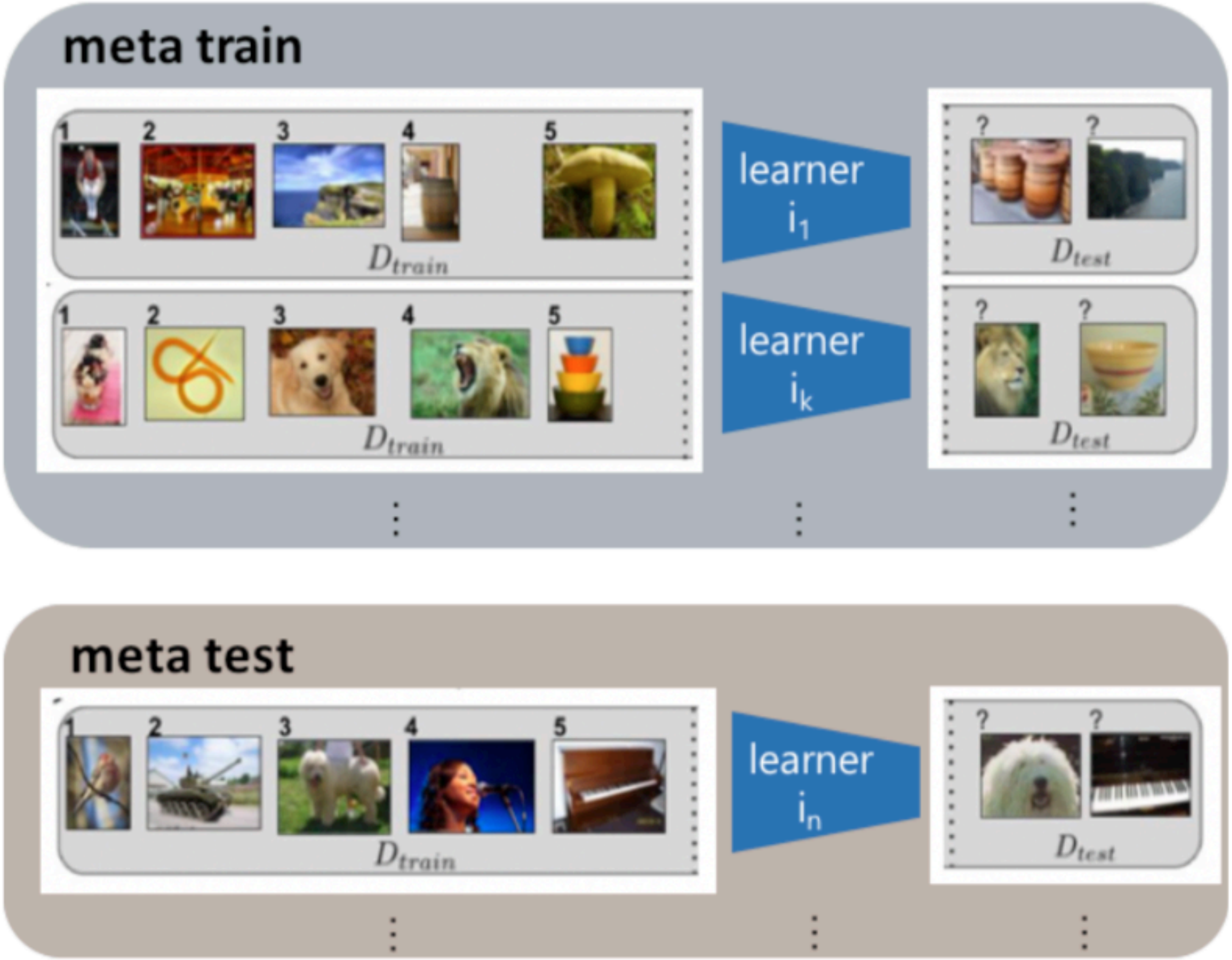
Learner quickly learn new tasks from a small amount of new data.

Meta Supervised Learning

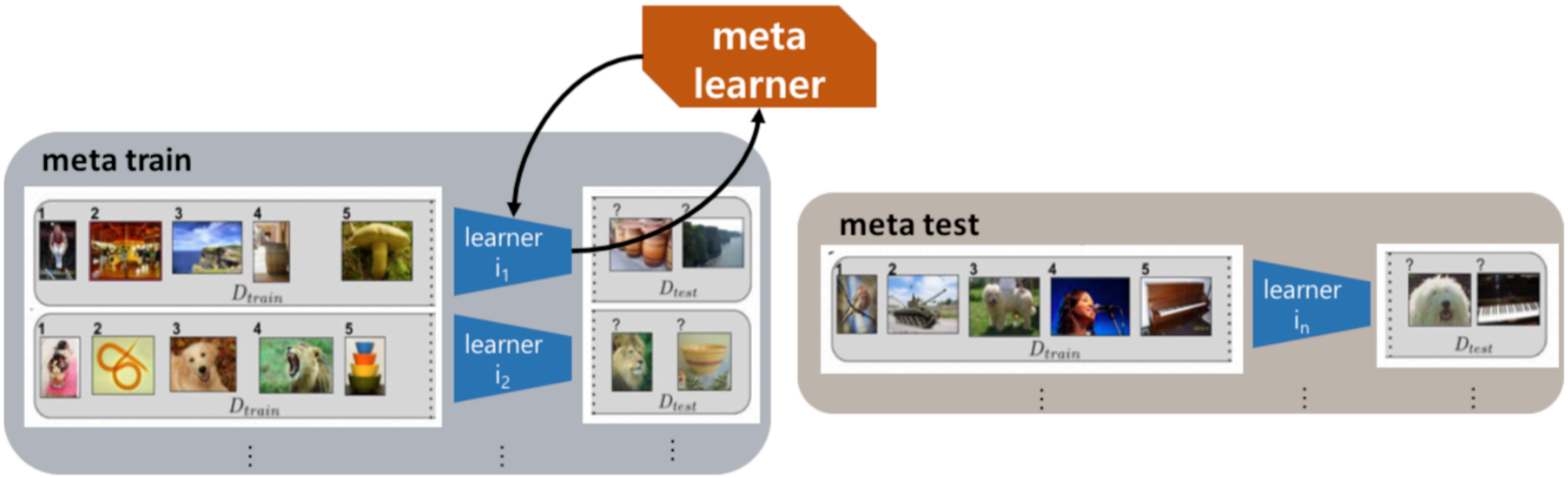


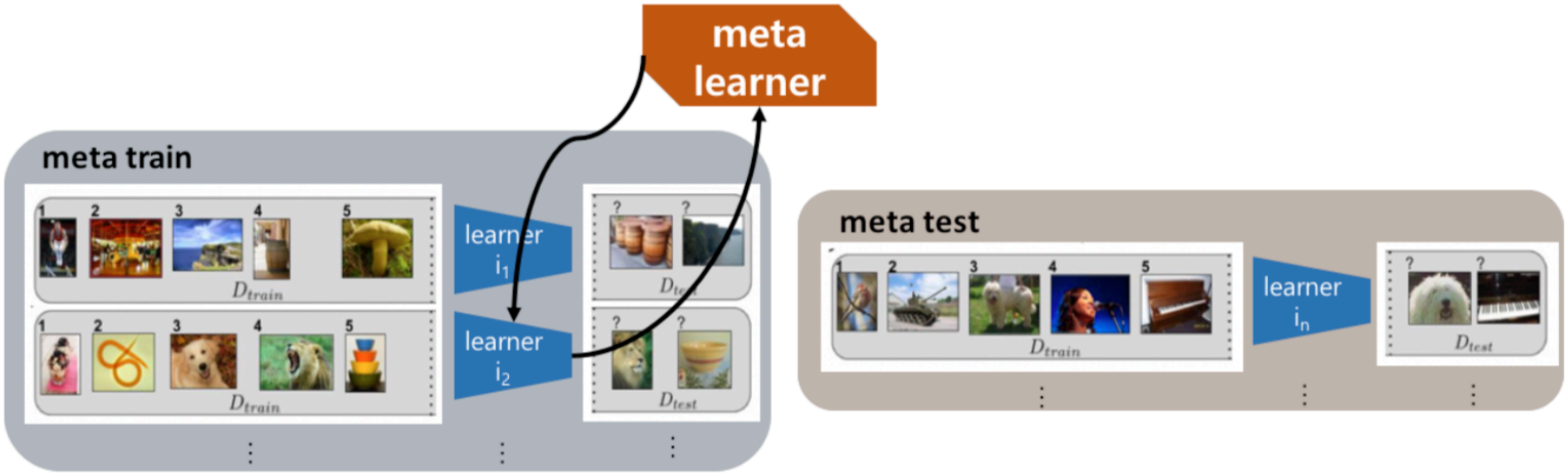


Meta Supervised Learning

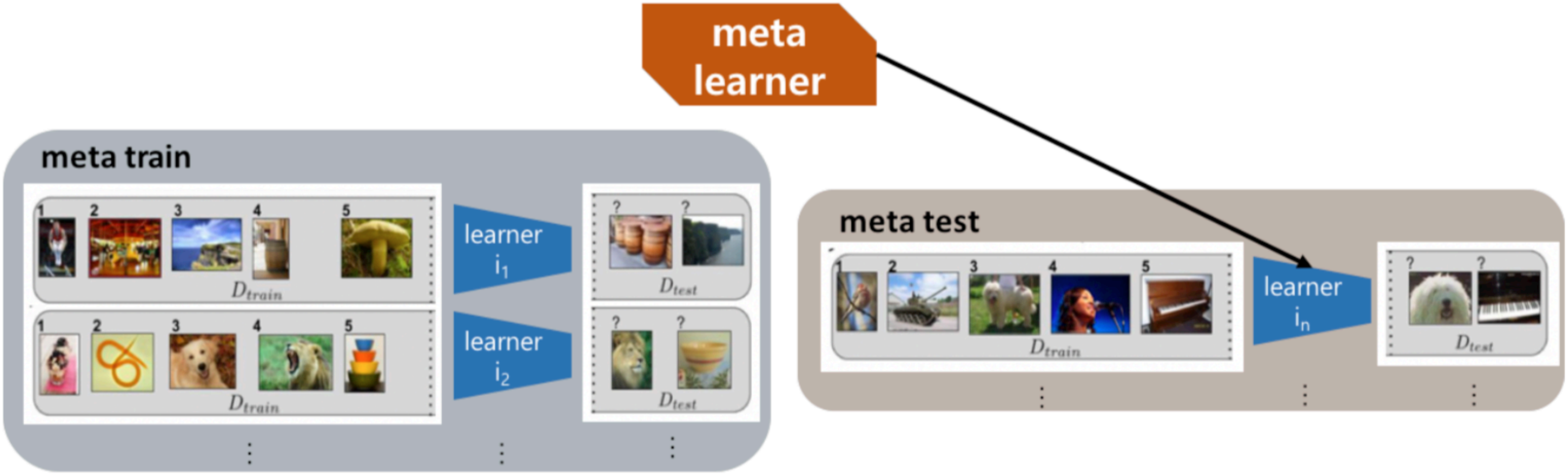


Meta Supervised Learning





Meta Supervised Learning



Meta learning

- Well Known Example

Fine Tuning

- Type of Meta-learning

	Model-based	Metric-based	Optimization-based
Key idea	RNN; memory	Metric learning	Gradient descent
How $P_\theta(y\|\mathbf{x})$ is modeled?	$f_\theta(\mathbf{x}, S)$	$\sum_{(\mathbf{x}_i, y_i) \in S} k_\theta(\mathbf{x}, \mathbf{x}_i) y_i$ (*)	$P_{g_\phi(\theta, S^L)}(y\|\mathbf{x})$

Introduction to Model Agnostic Meta Learning

Model Agnostic Meta Learning (called "MAML")

- Goal

Quick adapt to new tasks on distribution with only small amount of data and with only few gradient steps, even one gradient step.

- Learner

Learn a new tasks by using a single gradient step

- Meta - learner

Lean a generalized parameter initialization of model

Characteristic of MAML

The MAML learner's weight are updated using the gradient, rather than a learned update.

- No require additional parameters nor require a particular learner architecture

Fast adaptability through good parameter initialization

- Explicitly optimizes to learn internal representation (i.e suitable for many tasks)
- Maximize sensitivity of new tasks losses to the model parameters.

Characteristics of MAML

Model Agnostic (No matter what model is)

Classification / Regression with differentiable losses, Policy Gradient RL,

The model should be parameterized.

No other assumption on the form of the model.

Task Agnostic (No matter what task is)

Adopted all knowledge-transfer tasks.

No other assumption is required.

Model Agnostic Meta Learning

Some Internal representations are more transferable than others

Desired model parameter set is θ such that :

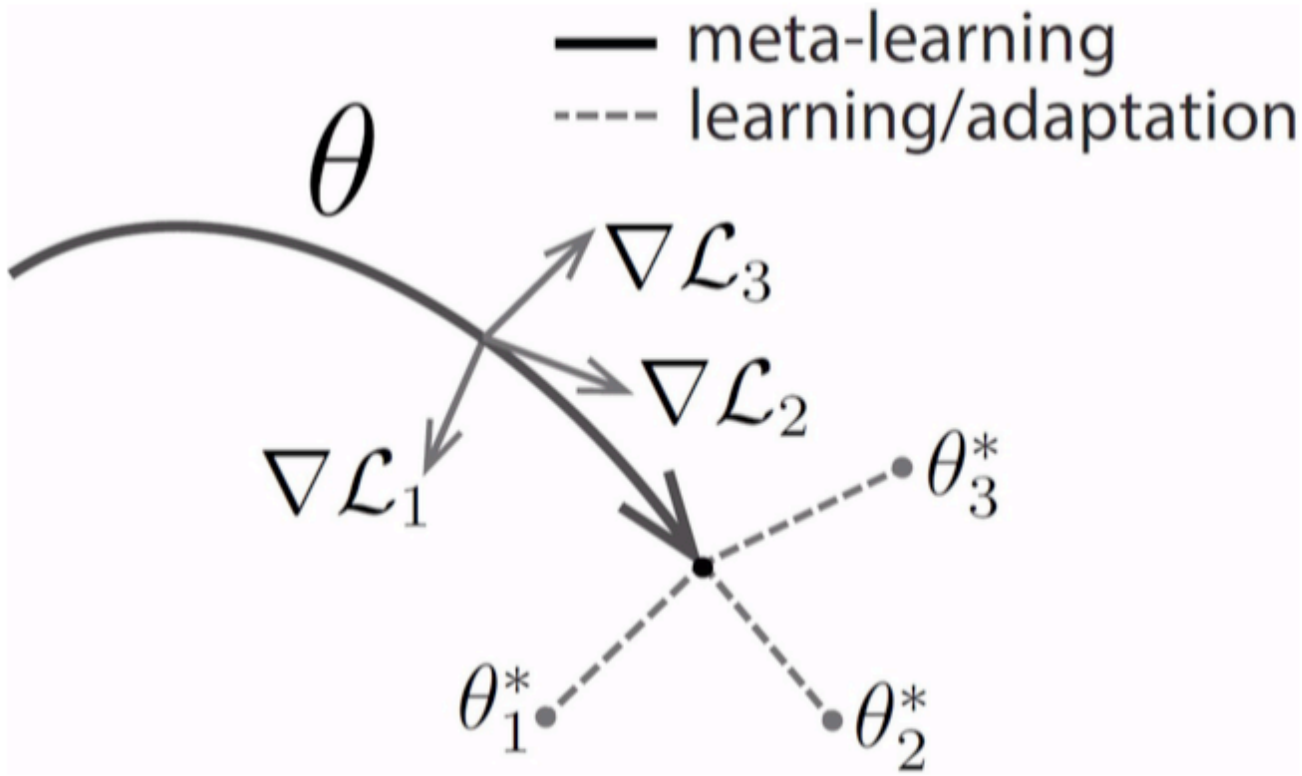
Applying one (or a small # of) gradient step to be θ on a new task will produce maximally effective behavior

Find θ that commonly decrease loss of each task after adaption

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 $\mathcal{L}_{\mathcal{T}_i}$
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 $\mathcal{L}_{\mathcal{T}_i}$
11: **end while**

Intuition of MAML



* **Inner Update** (Learner)

$$\phi_j = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(D^{tr}, \theta)) \quad \forall j$$

* **Outer Update** (Meta - Learner)

$$\theta = \theta - \beta \nabla_\theta \sum_{\mathcal{T}_j \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_j}(D^{te}, \phi_j)$$

Gradient of Gradient

- From line 10 in Algorithm 2,

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

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

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

($\mathcal{L}$ is differentiable)

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

$$= \theta - \beta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \boxed{(I - \alpha \nabla^2_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta))} \nabla_{\theta'_i} \mathcal{L}_{\mathcal{T}_i}(f_{\theta'_i})$$

Calculation of Hessian matrix is required.

Gradient of Gradient

- Update rule of MAML:

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T} \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}\left(f_{\theta - \alpha \nabla \mathcal{L}_{\mathcal{T}_i}(f_\theta)}\right)$$

- Update rule of MAML with 1st order approximation:

$$\delta \leftarrow \nabla \mathcal{L}_{\mathcal{T}_i}(f_\theta)$$

(Regard δ as constant)

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T} \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}\left(f_{\theta - \alpha \delta}\right)$$

Gradient of Gradient

- From line 10 in Algorithm 2,

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

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

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

($\mathcal{L}$ is differentiable)

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

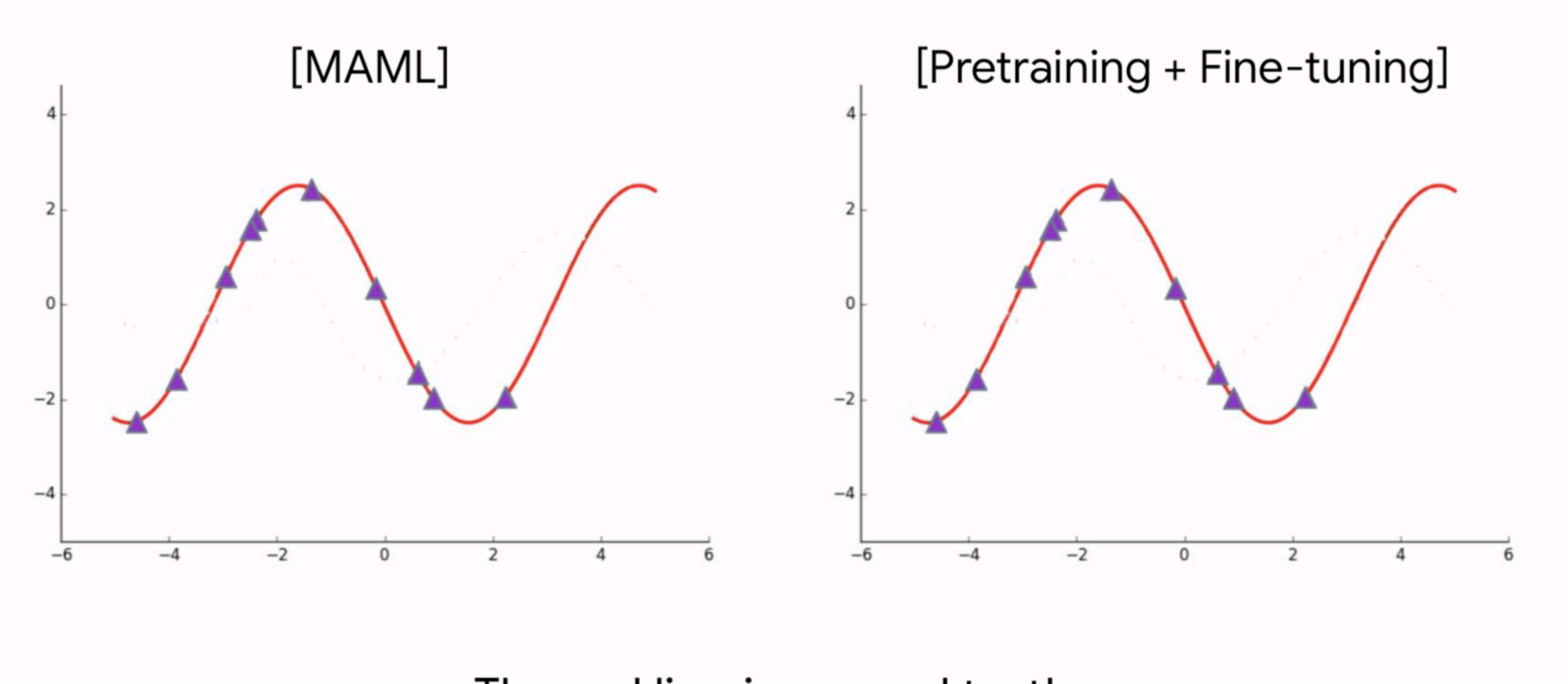
$$= \theta - \beta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \boxed{(I - \alpha \nabla^2_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta))} \nabla_{\theta'_i} \mathcal{L}_{\mathcal{T}_i}(f_{\theta'_i})$$

In 1st order approximation,
we regard this as identity matrix *I*.

Experimental result of MAML

Tasks :

$$y_i = a_i sin b_i x + c$$

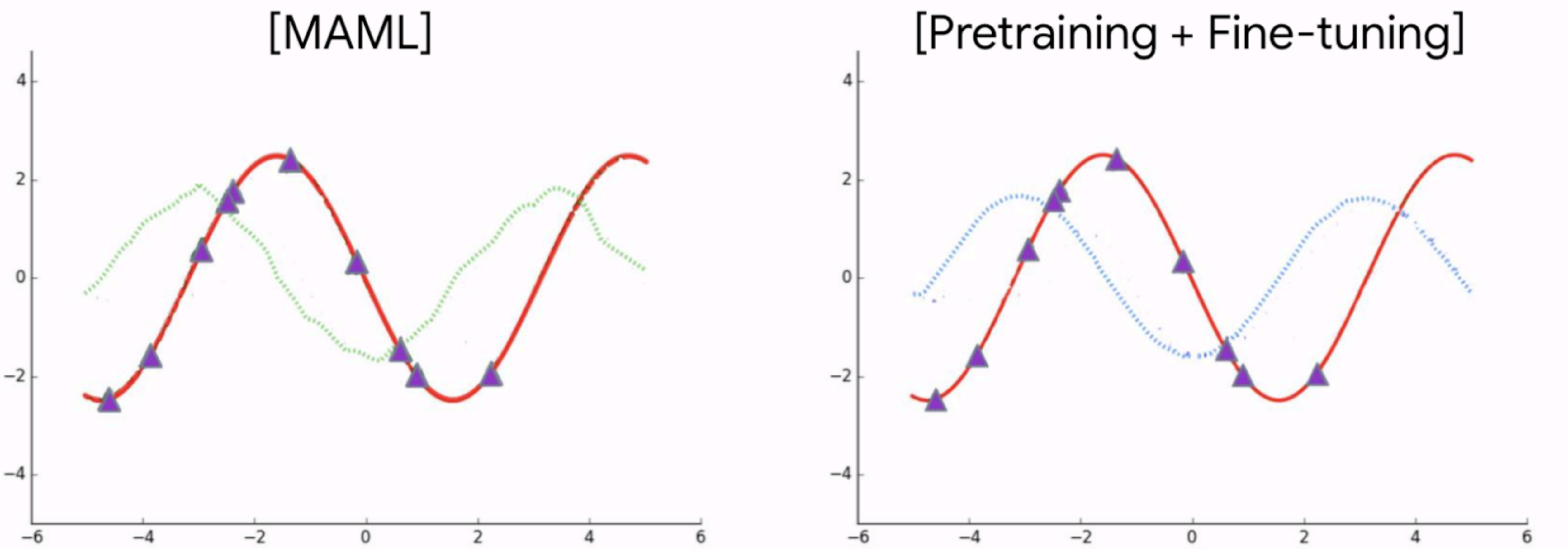


The red line is ground truth.
Fit this sine function with only few (10) samples.

Experimental result of MAML

10 shots Meta Learning :

Meta Learner model



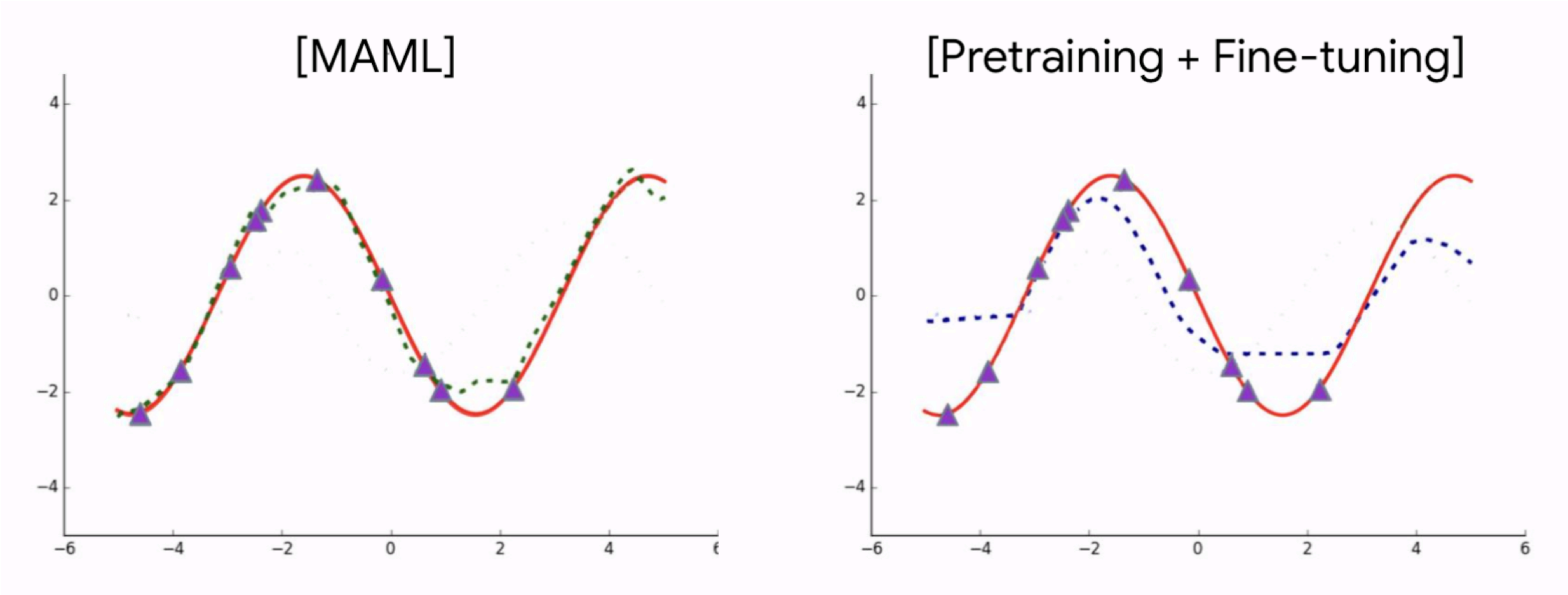
Above plots are the pre-trained function of two models.
(The prediction of meta-parameter of MAML,
The prediction of co-learned parameter of vanilla multi-task learning)

Experimental result of MAML

10 shots Meta Learning :

Learners model

10 gradient step updates



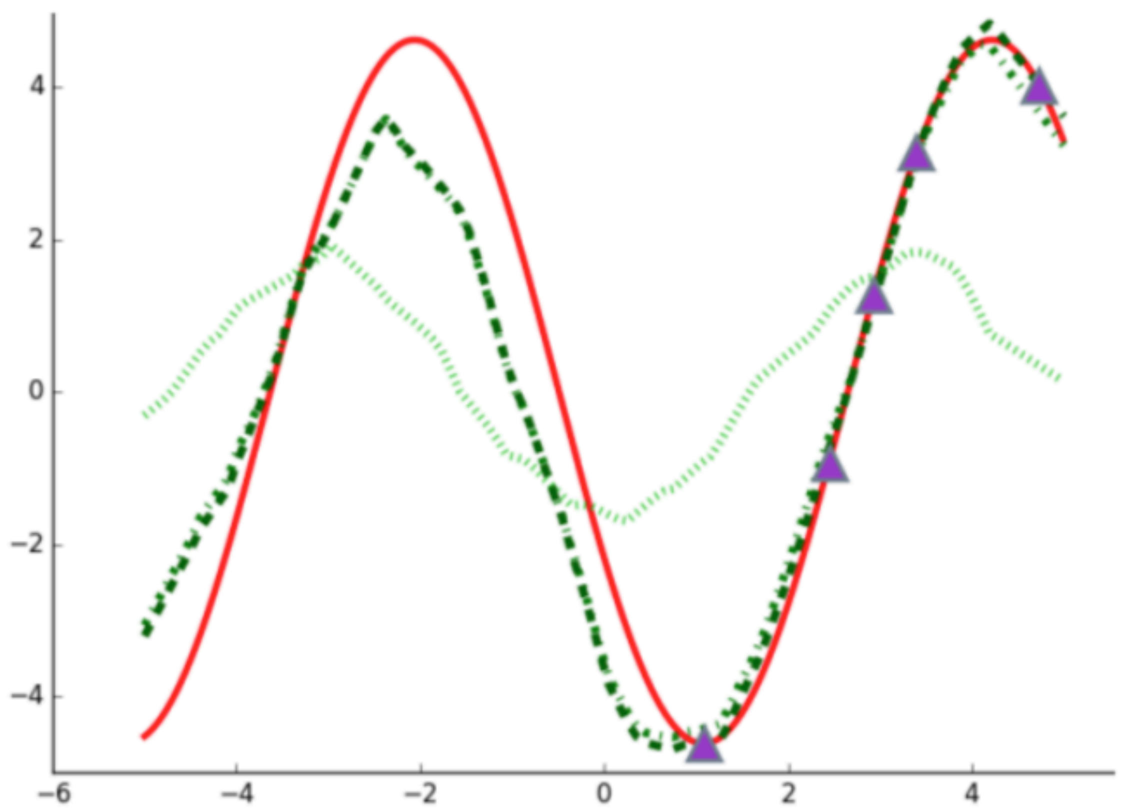
Experimental result of MAML

10 shots Meta Learning :

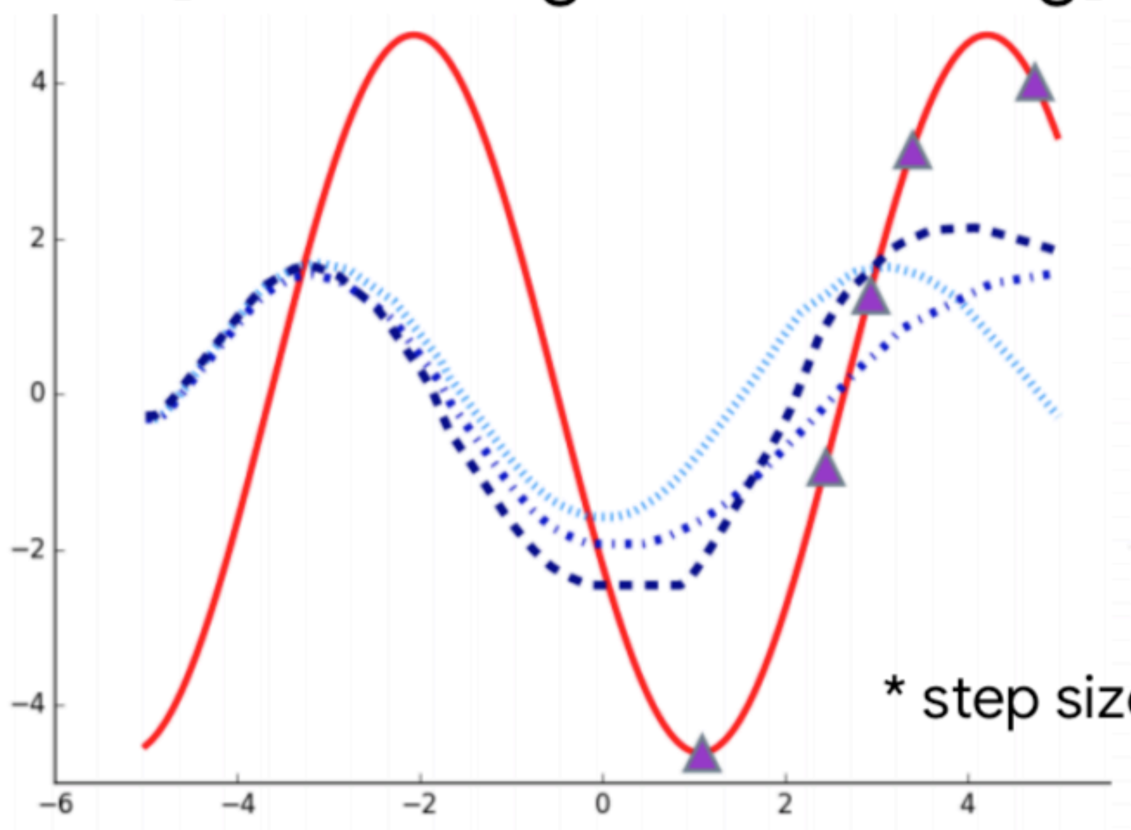
Learners model

10 gradient step updates

[MAML]



[Pretraining + Fine-tuning]



* step size 0.01

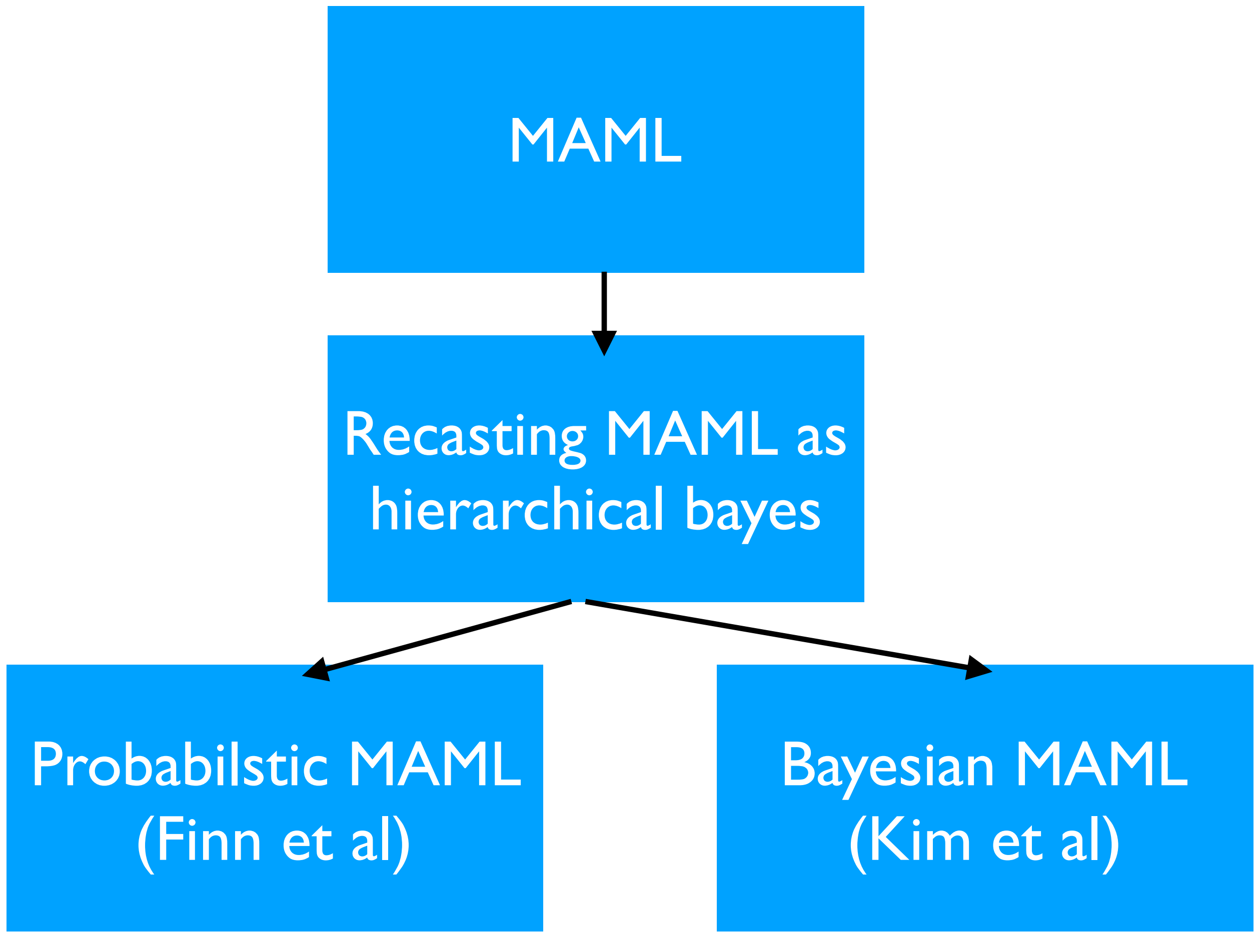
In the 5-shot learning, the difference is pervasive.

Graphical Representation Of MAML

Overview

1. Model Agnostic Meta Learning with Bayesian Approach
  - Recasting Gradient-based meta-learning as hierarchical bayes(2018), Grant et al. ICLR2018
  - Probabilistic Model Agnostic Meta learning(2018), Finn et al.Achieve
   + Bayesian MAML(2018),

2. Summarize bayesian approach in MAML



Gradient descents as finding MAP procedure

Santos(1993) represented that gradient descent means MAP for prior \theta

$$\phi_j = \theta - \gamma \nabla log p(X|\theta)$$

*Inner Update

$$\min_{\theta} \sum_{i} ||y_{ji} - \theta^T x_{ji}||_2^2 + ||\phi_j - \theta||_2^2 \qquad \min_{\phi_j} p(X|\phi)p(\phi|\theta)$$

In Grant(2018), MAML inner update means MAP using bayesian prior \theta

$$p(X|\theta) = \prod_{j} (\int p(x_{j1}, \cdots, x_{jn}|\phi_j)p(\phi_j|\theta)d\phi_j)$$

Outer update is same as finding out \theta^*

$$\theta^* = argmin_{\theta} p(X|\theta)$$

This procedure is called "A Probabilistic Interpretation of MAML"

MAML represented by probabilistic Interpretation

$$p(X|\theta) = \prod_j \left(\int p(x_{j1}, \cdots, x_{jn}|\phi_j) p(\phi_j|\theta) d\phi_j \right)$$

*** Inner Update**

$$\phi_j^* = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(D^{tr}, \theta))$$ solved,

*** MAML procedure is regarded as empirical Bayes**

$$-logP(X|\theta) \approx \sum_j [-logp(x_{j_{N+1}}, \ldots, x_{j_{N+M}}|\hat{\phi}_j)]$$

MAML represented by probabilistic Interpretation

MAML algorithm can be written as hierarchy Bayes

Algorithm `MAML-HB` ($\mathscr{D}$)

Initialize $\boldsymbol{\theta}$ randomly
while *not converged* **do**
  Draw J samples $\mathcal{T}_1, \ldots, \mathcal{T}_J \sim p_{\mathscr{D}}(\mathcal{T})$
  Estimate $\mathbb{E}_{\mathbf{x} \sim p_{\mathcal{T}_1}(\mathbf{x})}[-\log p(\mathbf{x} \mid \boldsymbol{\theta})], \ldots, \mathbb{E}_{\mathbf{x} \sim p_{\mathcal{T}_J}(\mathbf{x})}[-\log p(\mathbf{x} \mid \boldsymbol{\theta})]$ using `ML-···`
  Update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \beta \nabla_{\boldsymbol{\theta}} \sum_j \mathbb{E}_{\mathbf{x} \sim p_{\mathcal{T}_j}(\mathbf{x})}[-\log p(\mathbf{x} \mid \boldsymbol{\theta})]$
end

Algorithm 2: Model-agnostic meta-learning as hierarchical Bayesian inference. The choices of the subroutine `ML-···` that we consider are defined in Subroutine 3 and Subroutine 4.

Subroutine `ML-POINT` ($\boldsymbol{\theta}, \mathcal{T}$)

Draw N samples $\mathbf{x}_1, \ldots, \mathbf{x}_N \sim p_{\mathcal{T}}(\mathbf{x})$
Initialize $\boldsymbol{\phi} \leftarrow \boldsymbol{\theta}$
for k *in* $1, \ldots, K$ **do**
  Update $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} + \alpha \nabla_{\boldsymbol{\phi}} \log p(\mathbf{x}_1, \ldots, \mathbf{x}_N \mid \boldsymbol{\phi})$
end
Draw M samples $\mathbf{x}_{N+1}, \ldots, \mathbf{x}_{N+M} \sim p_{\mathcal{T}}(\mathbf{x})$
return $-\log p(\mathbf{x}_{N+1}, \ldots, \mathbf{x}_{N+M} \mid \boldsymbol{\phi})$

Laplace Approximation for Meta Adaption

Laplace Approximation of hierarchical Bayes

$$\int p\left(\mathbf{X}_j \mid \boldsymbol{\phi}_j\right) p\left(\boldsymbol{\phi}_j \mid \boldsymbol{\theta}\right) \mathrm{d}\boldsymbol{\phi}_j \approx p\left(\mathbf{X}_j \mid \boldsymbol{\phi}_j^*\right) p\left(\boldsymbol{\phi}_j^* \mid \boldsymbol{\theta}\right) \det(\mathbf{H}_j/2\pi)^{-\frac{1}{2}}$$

Laplace Approximation
- **mean : \phi^***
- **covariance : H_j**

There is no closed form of above equation.

So that, the authors supposed this covariance can be found by neural network.

$$\mathbf{H}_j = \nabla^2_{\boldsymbol{\phi}_j}\left[-\log p\left(\mathbf{X}_j \mid \boldsymbol{\phi}_j\right)\right] + \nabla^2_{\boldsymbol{\phi}_j}\left[-\log p\left(\boldsymbol{\phi}_j \mid \boldsymbol{\theta}\right)\right] \ .$$

Solve easily, assumed that fixed constant diagonal covariance : \tau

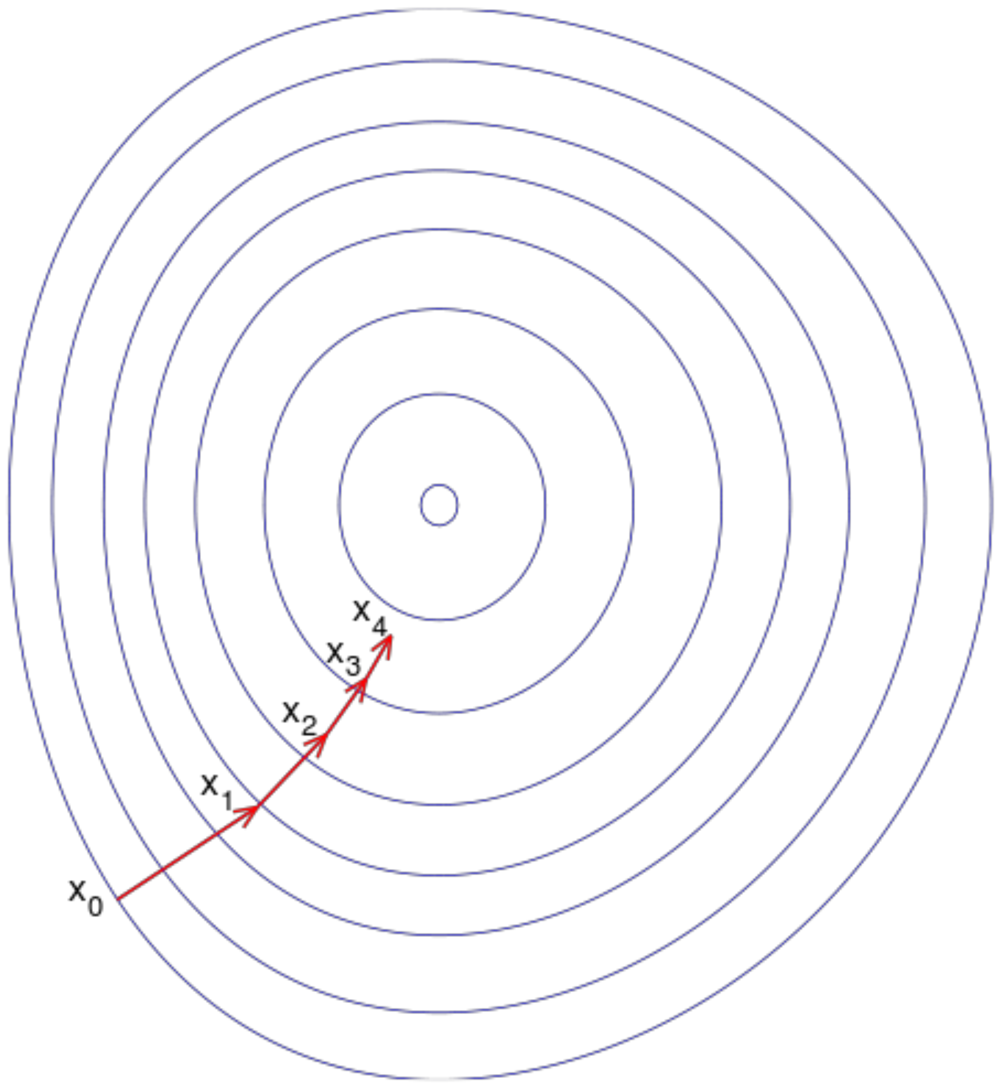
Subroutine ML-LAPLACE $(\boldsymbol{\theta}, \mathcal{T})$
- Draw N samples $\mathbf{x}_1, \ldots, \mathbf{x}_N \sim p_{\mathcal{T}}(\mathbf{x})$
- Initialize $\boldsymbol{\phi} \leftarrow \boldsymbol{\theta}$
- **for** k *in* $1, \ldots, K$ **do**
  - Update $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} + \alpha \nabla_{\boldsymbol{\phi}} \log p(\mathbf{x}_1, \ldots, \mathbf{x}_N \mid \boldsymbol{\phi})$
- **end**
- Draw M samples $\mathbf{x}_{N+1}, \ldots, \mathbf{x}_{N+M} \sim p_{\mathcal{T}}(\mathbf{x})$
- Estimate quadratic curvature $\hat{\mathbf{H}}$
- **return** $-\log p(\mathbf{x}_{N+1}, \ldots, \mathbf{x}_{N+M} \mid \boldsymbol{\phi}) + \eta \log \det(\hat{\mathbf{H}})$

Stein Variational Gradient Desecnt

Bayesian Inference : MCMC / SVGD

https://chi-feng.github.io/mcmc-demo/app.html#HamiltonianMC,banana

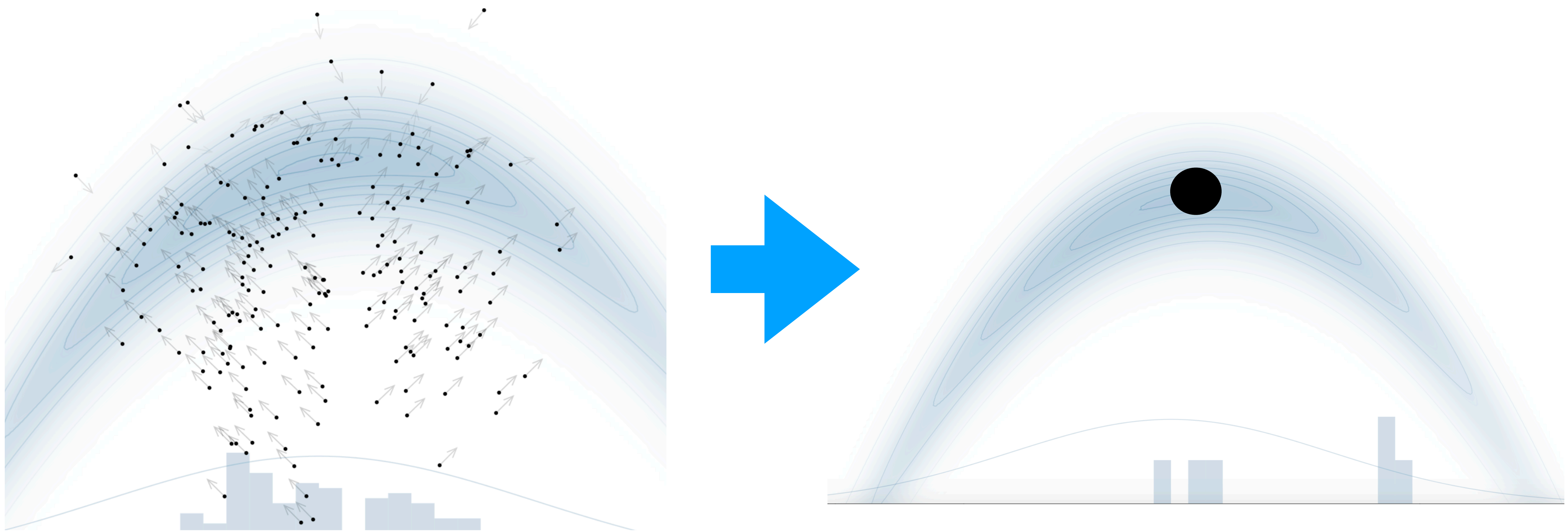
Gradient descent



$$x^{t+1} = x_t + \epsilon \cdot \phi(x)$$

$$\phi(x) = -\nabla_x f(x)$$

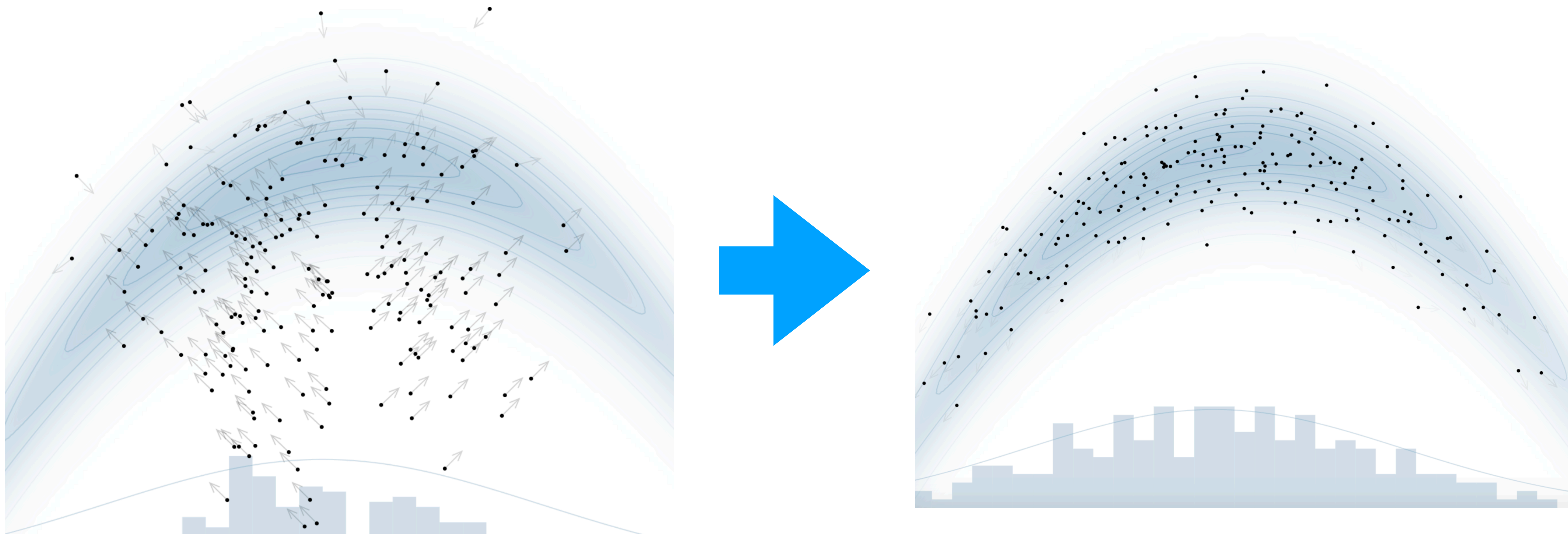
MLE / MAP with Gradient descent



$$x^{t+1} = x_t + \epsilon \cdot \frac{\mathrm{d}\, logp(x)}{dx}$$

If several modes, this method captures only one mode.

Stein variational gradient descent



$$x^{t+1} = x_t + \epsilon \cdot \phi(x)$$

$$\phi^*(x) = -\mathbb{E}_x[\nabla_x logp(x) \cdot k(\cdot, x) + \nabla_x k(\cdot, x)]$$

It is required to have only M particles to obtain M samples from specific PDF

No rejected sample !

Algorithm

- Iterative Update

$$\boldsymbol{T}_l^*(x) = x + \epsilon_l \boldsymbol{\phi}^*_{q_l,p}(x).$$

$$q_0 \xrightarrow{\boldsymbol{T}_0^*} q_1 \xrightarrow{\boldsymbol{T}_1^*} q_2 \xrightarrow{\boldsymbol{T}_2^*} \cdots \to q_\infty = p(\boldsymbol{\phi}^*_{q_l,p} = 0)$$

At each step, KLD decreases by an amount of $\epsilon_l \mathbb{S}(q_l, p)$

Algorithm 1 Bayesian Inference via Variational Gradient Descent

Input: A target distribution with density function $p(x)$ and a set of initial particles $\{x_i^0\}_{i=1}^n$.
Output: A set of particles $\{x_i\}_{i=1}^n$ that approximates the target distribution.
for iteration ℓ **do**

$$x_i^{\ell+1} \leftarrow x_i^\ell + \epsilon_\ell \hat{\phi}^*(x_i^\ell) \quad \text{where} \quad \hat{\phi}^*(x) = \frac{1}{n}\sum_{j=1}^n \left[k(x_j^\ell, x)\nabla_{x_j^\ell} \log p(x_j^\ell) + \nabla_{x_j^\ell} k(x_j^\ell, x)\right], \quad (8)$$

where ϵ_ℓ is the step size at the ℓ-th iteration.
end for

Experiment Result of SVGD

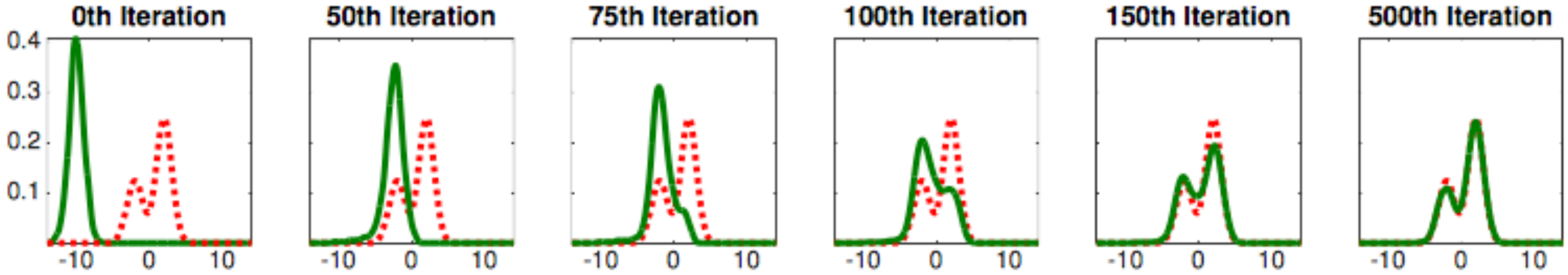


Figure 1: Toy example with 1D Gaussian mixture. The red dashed lines are the target density function and the solid green lines are the densities of the particles at different iterations of our algorithm (estimated using kernel density estimator) . Note that the initial distribution is set to have almost zero overlap with the target distribution, and our method demonstrates the ability of escaping the local mode on the left to recover the mode on the left that is further away. We use $n = 100$ particles.

Bayesian MAML

Bayesian Model Agnostic Meta Learning

MAML objective

$$p(\mathcal{D}_{\mathcal{T}}^{\text{val}} \mid \theta_0, \mathcal{D}_{\mathcal{T}}^{\text{trn}}) = \prod_{\tau \in \mathcal{T}} p(\mathcal{D}_{\tau}^{\text{val}} \mid \theta'_{\tau} = \theta_0 + \alpha \nabla_{\theta_0} \log p(\mathcal{D}_{\tau}^{\text{trn}} \mid \theta_0)),$$

Grant 2018 and This paper

$$p(\mathcal{D}_{\mathcal{T}}^{\text{val}} \mid \theta_0, \mathcal{D}_{\mathcal{T}}^{\text{trn}}) = \prod_{\tau \in \mathcal{T}} \left(\int p(\mathcal{D}_{\tau}^{\text{val}} \mid \theta_{\tau}) p(\theta_{\tau} \mid \mathcal{D}_{\tau}^{\text{trn}}, \theta_0) \mathrm{d}\theta_{\tau} \right).$$

Grant 2018 : Assume task specific posterior as isotropic gaussian fixed variance

This paper : model this property with SGVD and

$$p(\theta_{\tau} | \mathcal{D}_{\tau}^{\text{trn}}) \propto \prod_{(x,y) \in \mathcal{D}_{\tau}^{\text{trn}}} \mathcal{N}(y | f_W(x), \gamma^{-1}) \prod_{w \in W} \mathcal{N}(w | 0, \lambda^{-1}) \text{Gamma}(\gamma | a, b) \text{Gamma}(\lambda | a', b')$$

Chapter 5

Bayesian Agnostic Meta Learning

Algorithm 3 Bayesian Meta-Learning with Chaser Loss (BMAML)

Initialize Θ_0
for $t = 0, \dots$ until converge **do**
  Sample a mini-batch of tasks $\mathcal{T}_t$ from $p(\mathcal{T})$
  for each task $\tau \in \mathcal{T}_t$ **do**
    Compute chaser $\Theta_\tau^n(\Theta_0) = \text{SVGD}_n(\Theta_0; \mathcal{D}_\tau^{\text{trn}}, \alpha)$
    Compute leader $\Theta_\tau^{n+s}(\Theta_0) = \text{SVGD}_s(\Theta_\tau^n(\Theta_0); \mathcal{D}_\tau^{\text{trn}} \cup \mathcal{D}_\tau^{\text{val}}, \alpha)$
  end for
  $\Theta_0 \leftarrow \Theta_0 - \beta \nabla_{\Theta_0} \sum_{\tau \in \mathcal{T}_t} d_s(\Theta_\tau^n(\Theta_0) \parallel \text{stopgrad}(\Theta_\tau^{n+s}(\Theta_0)))$
end for

Introduce and follows

$$p(\theta_\tau | D_\tau^{trn}) \propto p(D_\tau^{trn} | \theta_\tau) p(\theta_\tau)$$

Experimental Results

Experimental result of MAML

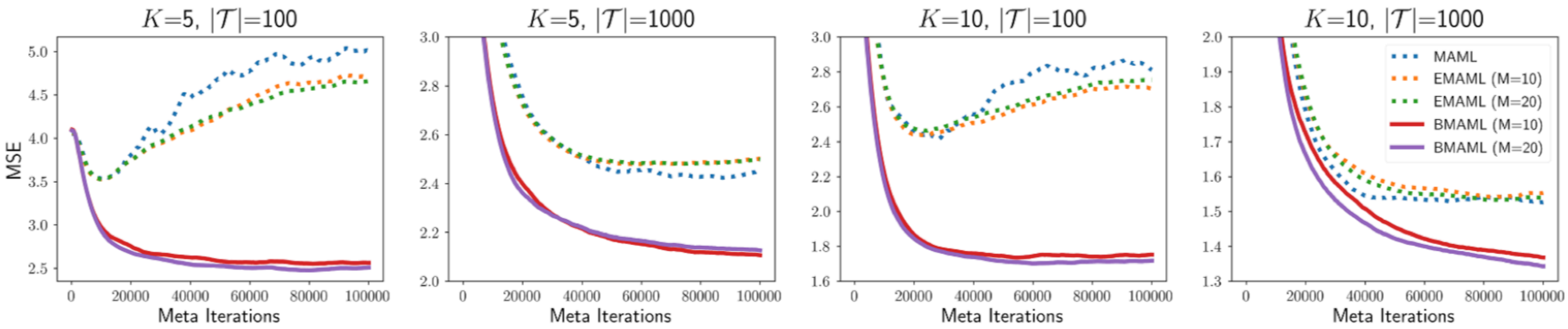


Figure 1: Sinusoidal regression experimental results (meta-testing performance) by varying the number of examples (K-shot) given for each task and the number of tasks $|\mathcal{T}|$ used for meta-training.

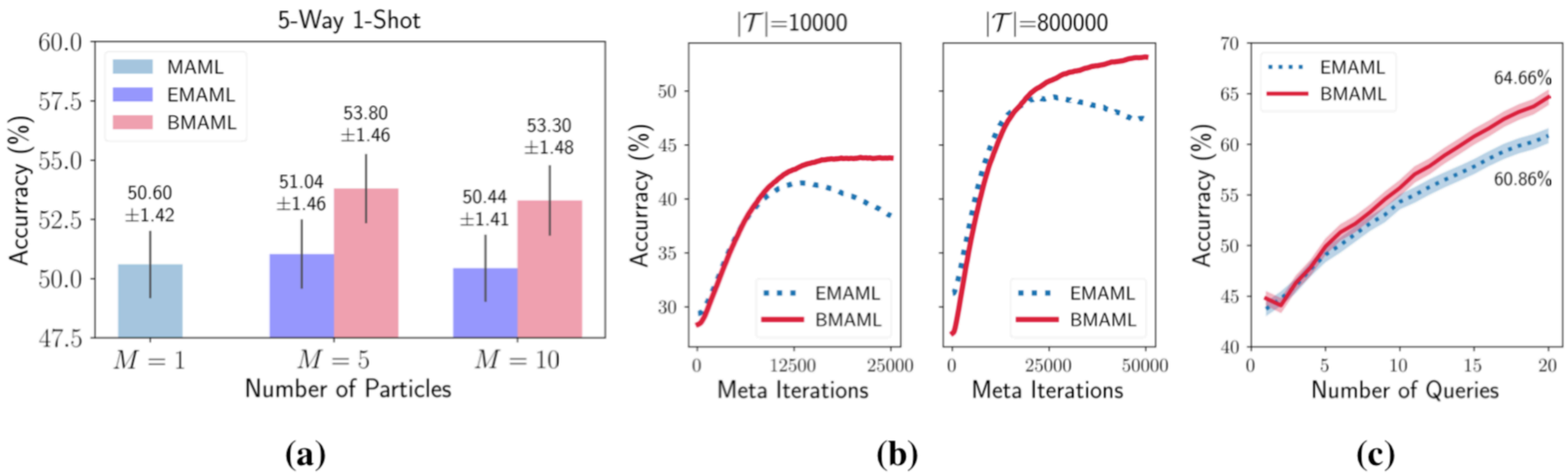


Figure 2: Experimental results in *mini*Imagenet dataset: (a) few-shot image classification using different number of particles, (b) using different number of tasks for meta-training, and (c) active learning setting.

Reference

Reference

Qiang Liu and Dilin Wang(2016), *"Stein Variational Gradient Descent :A general Purpose Bayesian Inference"*, NIPS2016

Chelsea Finn, Pieter Abbeel and Sergey Levine(2017), *"Model Agnostic Meta Learning for Fast Adaption of Deep network"*, ICML2017

Erin Grant, Chelsea Finn, Sergey Levine, Trevor Darrell and Thomas Griffiths(2018), *"Recasting Gradient Based Meta Learning as Hierarchical Bayes"*, ICLR2018

Taesup Kim, Jaesik Yoon, Ousmane Dia, Sungwoon Kim, Yoshua Bengio and Sungjin Ahn(2018), *"Bayesian Model Agnostic Meta Learning"*, NeurIPS2018

Thanks