Meta-Learning:
Learning-to-Learn in Neural Networks

Changhoon, Kevin Jeong
Seoul National University
chjeong@bi.snu.ac.kr

Agenda

- Meta-Learning: Overview

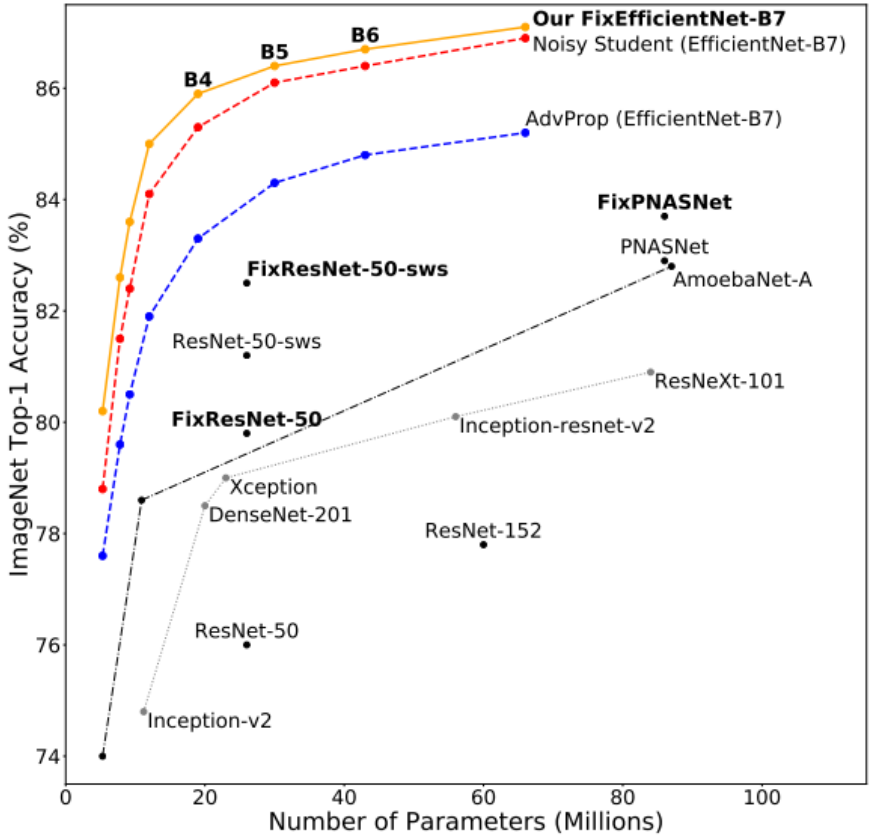
- Meta Supervised Learning
  - Black-Box / Model-based approach
  - Optimization-based approach
  - Non-parametric approach

- Meta Reinforcement Learning
  - Recurrent policies approach
  - Optimization-based approach
  - POMDPs-based approach

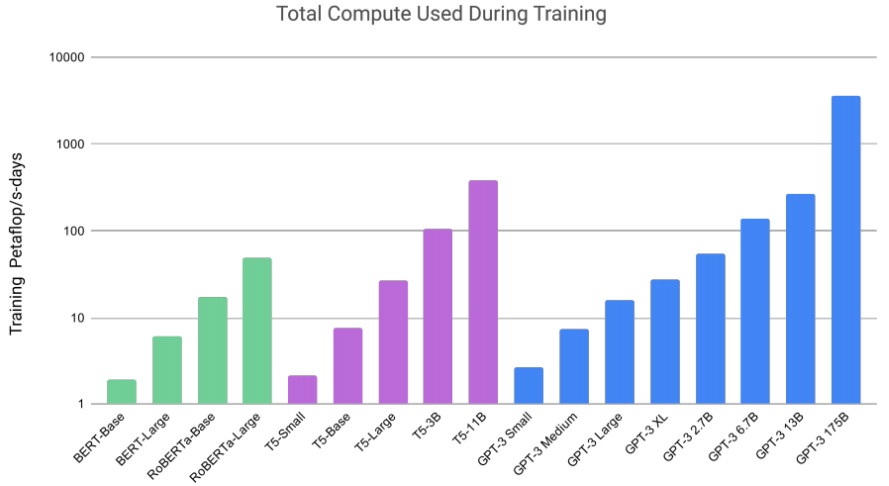
Meta-Learning: Overview

Deep Learning: State Of The ART

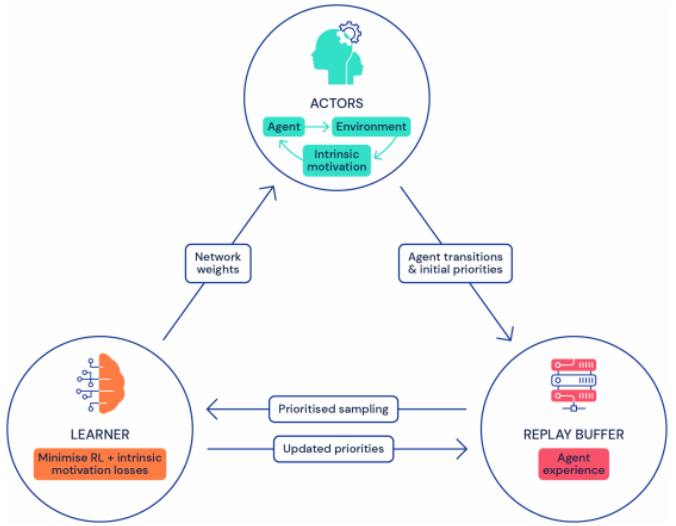
- Large, diverse dataset, large computation → Broad Generalization



FixEfficientNet



GPT-3



Agent57

Touvron, Hugo, et al. "Fixing the train-test resolution discrepancy: FixEfficientNet." 2020.

Brown, Tom B., et al. "Language models are few-shot learners." 2020.

Badia, Adrià Puigdomènech, et al. "Agent57: Outperforming the atari human benchmark." 2020.

Difference between Machine and Human

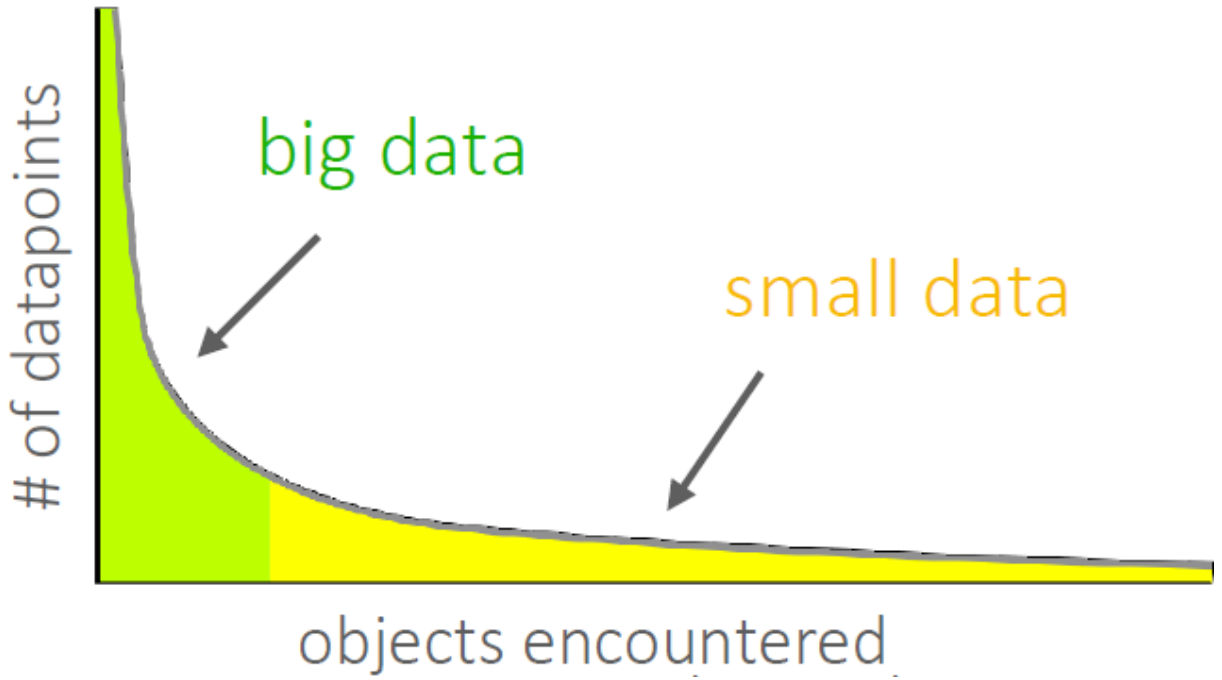
Machine is a specialist

Human is a generalist

Problem Statement

What if your data has a long tail?

What if you want a general-purpose AI system in the real world?

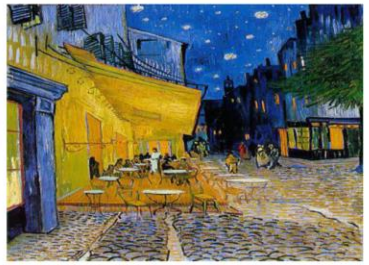


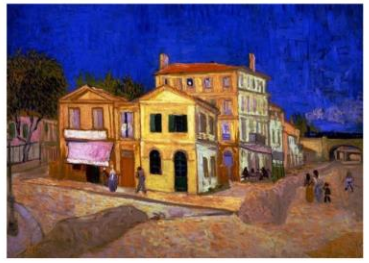
Few-Shot Learning

Can you classify the image using just 3 datapoints?

Training data

Van Gogh





Paul Cezanne

Test datapoint

By Gogh or Cezanne?

→ Few-Shot Learning
(2 way 3 shot)

Few-Shot Learning

How did you accomplish this?

By leveraging prior experience!

What is the Meta-Learning?

- Meta Learning in Computer Science(Wikipedia)
  - Meta-learning is a subfield of machine learning where automatic learning algorithms are applied on metadata about machine learning experiments: learning-to-learn

- How can we understand the meta-learning?
  - Mechanistic view
    - For implementation the algorithm
    - Training the neural network uses a meta-dataset, which itself consists of many datasets(each for a task)

  - Probabilistic view
    - For making it easier to understand the algorithm
    - Extract prior information, and learning a new task uses this prior and small training set to infer most likely posterior parameters

Formalizing the Meta Learning

- Conventional Machine Learning
  - (Supervised Learning) Given a training dataset $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$, we can train a model $\hat{y} = f_\theta(x)$ parameterized by θ, by solving,

$$\theta^* = \arg\min_\theta \mathcal{L}(\mathcal{D}; \theta, \omega)$$

- $\mathcal{L}$ is a loss function that measures the match between true labels and those predicted by $f_\theta(\cdot)$
- ω is the condition to make explicit the dependence of this solution on factors such as choice of optimizer(e.g. Adam, SGD, etc.) for or function class for f (e.g. CNN, RNN, etc.)

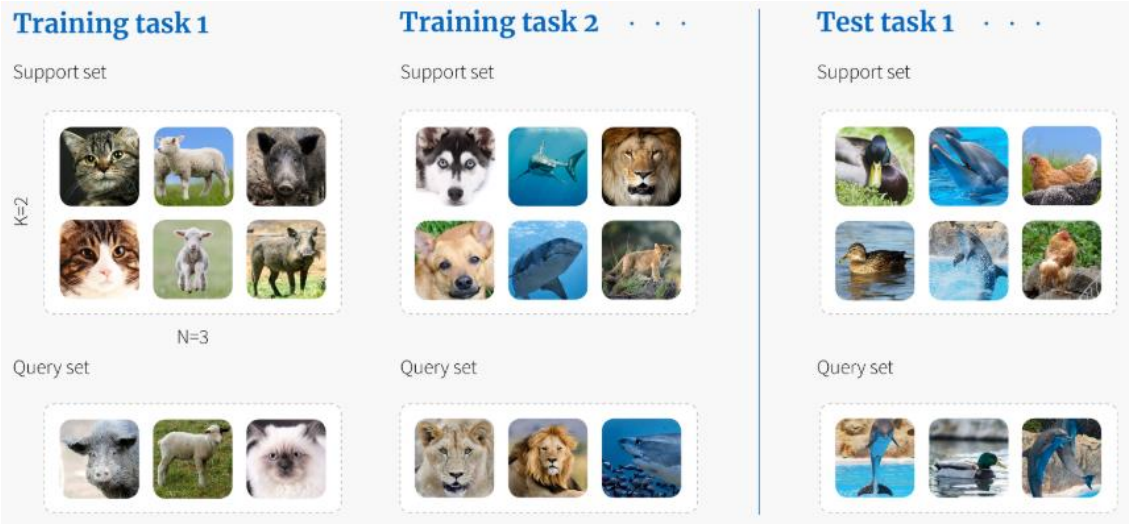
* ω is pre-specified, can we learn the learning algorithm itself, rather than assuming it is pre-specified and fixed?

Formalizing the Meta Learning

- What is a task?
  - A task: $\mathcal{T} = \{\mathcal{D}, \mathcal{L}\}$

For now: dataset $\mathcal{D}$, loss function $\mathcal{L}$ → model f_θ

- Different tasks can vary based on:
  - different objects
  - different people
  - different objectives
  - different lighting conditions
  - different languages
  - …



CS330: Deep Multi-Task and Meta-Learning, Tutorial #2: few-shot learning and meta-learning(by W. Zi)

Formalizing the Meta Learning

- The bad news
  - Different tasks need to share some structure
  - If this doesn't hold, you are better off using sing-task learning

- The good news
  - There are many tasks with shared structure!



  - the laws of physics, languages all develop for similar pupose, rules of English, etc.

Formalizing the Meta Learning

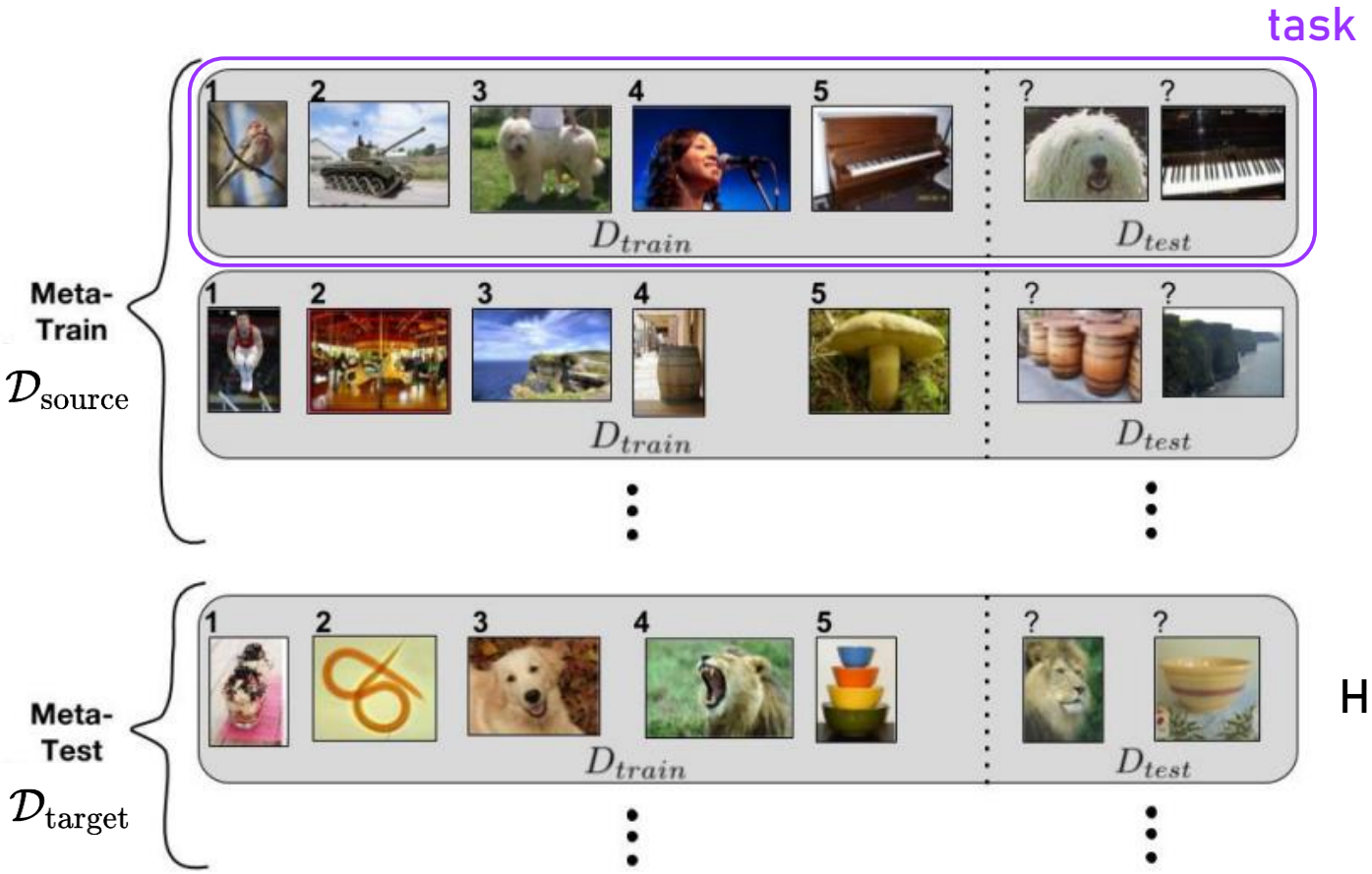
- Meta-Learning: Task-Distribution View
  - ω specifies 'how-to-learn' and is often evaluated in terms of performance over a distribution of tasks(meta-knowledge, inductive bias)
  - Learning how to learn solving by,

$$\min_{\omega} \mathop{\mathbb{E}}_{\mathcal{T} \sim p(\mathcal{T})} \mathcal{L}(\mathcal{D}; \omega)$$

How can we solve this problem in practice?

Meta-datasets

- The datasets of datasets(Supervised learning)



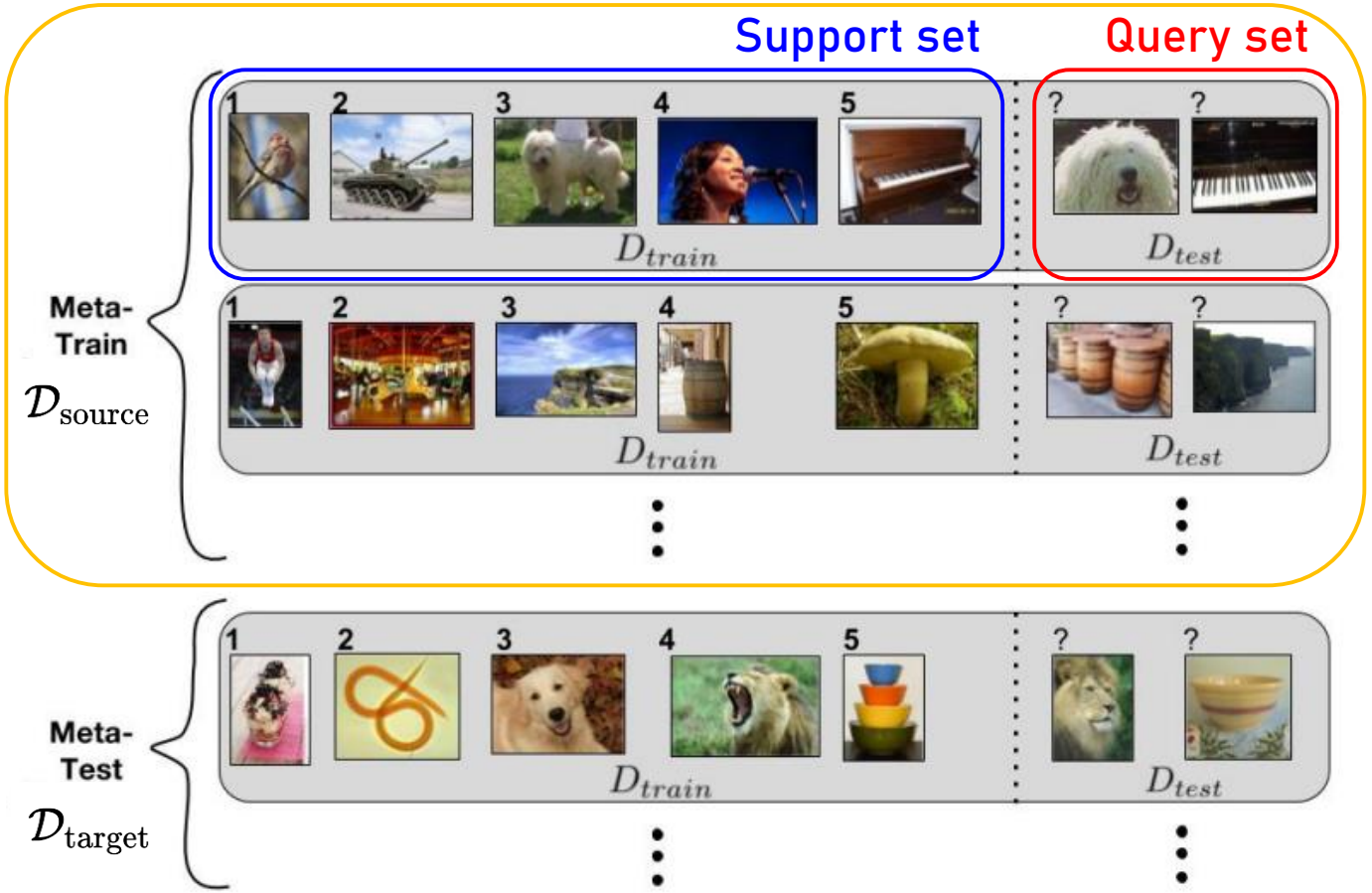
$$\mathcal{T} \sim p(\mathcal{T})$$

Held-out classes

Ravi, Sachin, and Hugo Larochelle. "Optimization as a model for few-shot learning." 2016.

Meta-Learning: Overview

- Meta-training(Supervised learning)



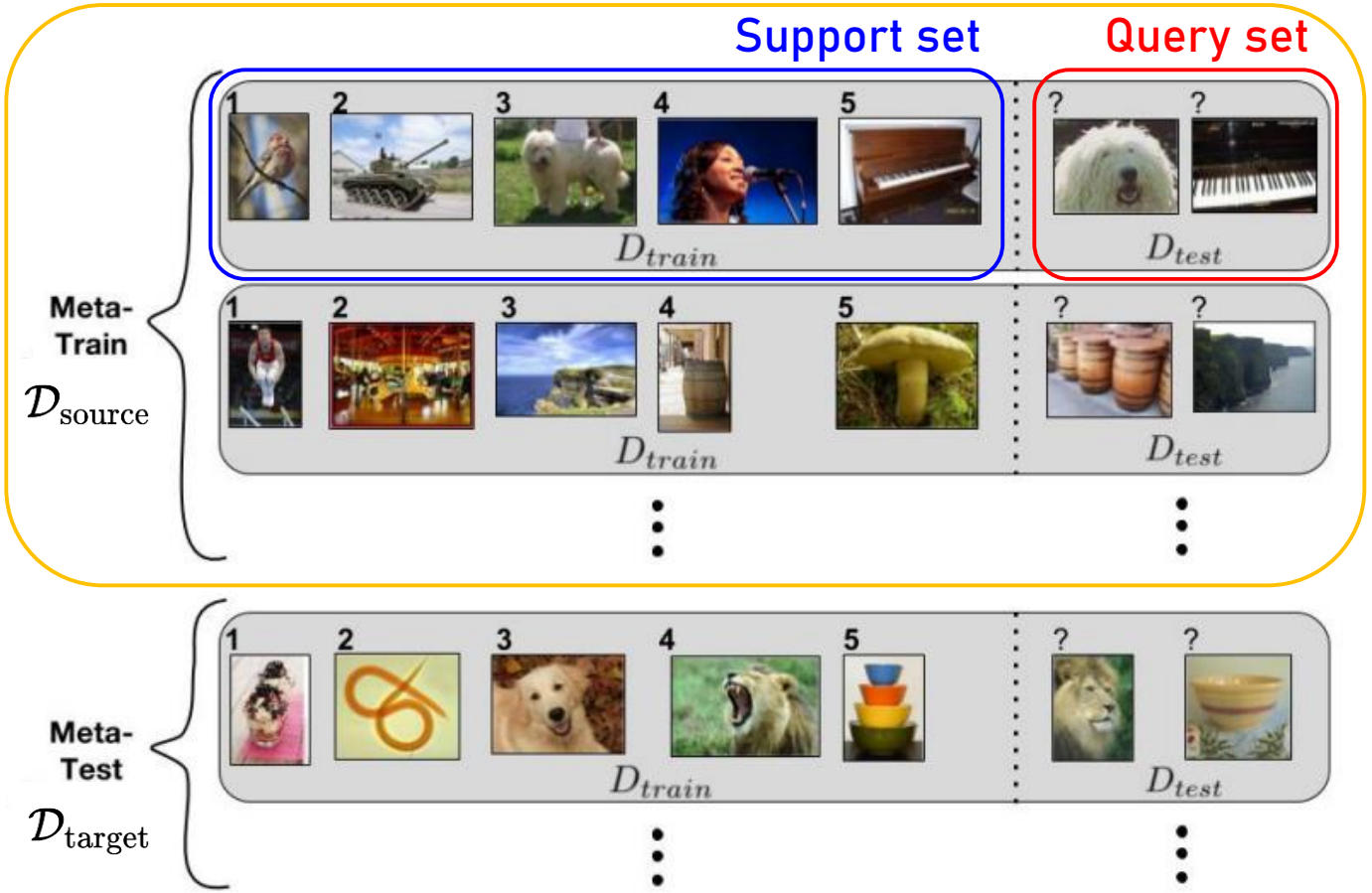
$$\mathcal{T} \sim p(\mathcal{T})$$

$$\mathcal{D}_{\text{source}} = \left\{ \left(\mathcal{D}_{\text{source}}^{\text{support}}, \mathcal{D}_{\text{source}}^{\text{query}} \right)^{(i)} \right\}$$

Ravi, Sachin, and Hugo Larochelle. "Optimization as a model for few-shot learning." 2016.

Meta-Learning: Overview

- Meta-training(Supervised learning)



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

$$\mathcal{D}_{\text{source}} = \left\{ \left(\mathcal{D}_{\text{source}}^{\text{support}}, \mathcal{D}_{\text{source}}^{\text{query}} \right)^{(i)} \right\}$$

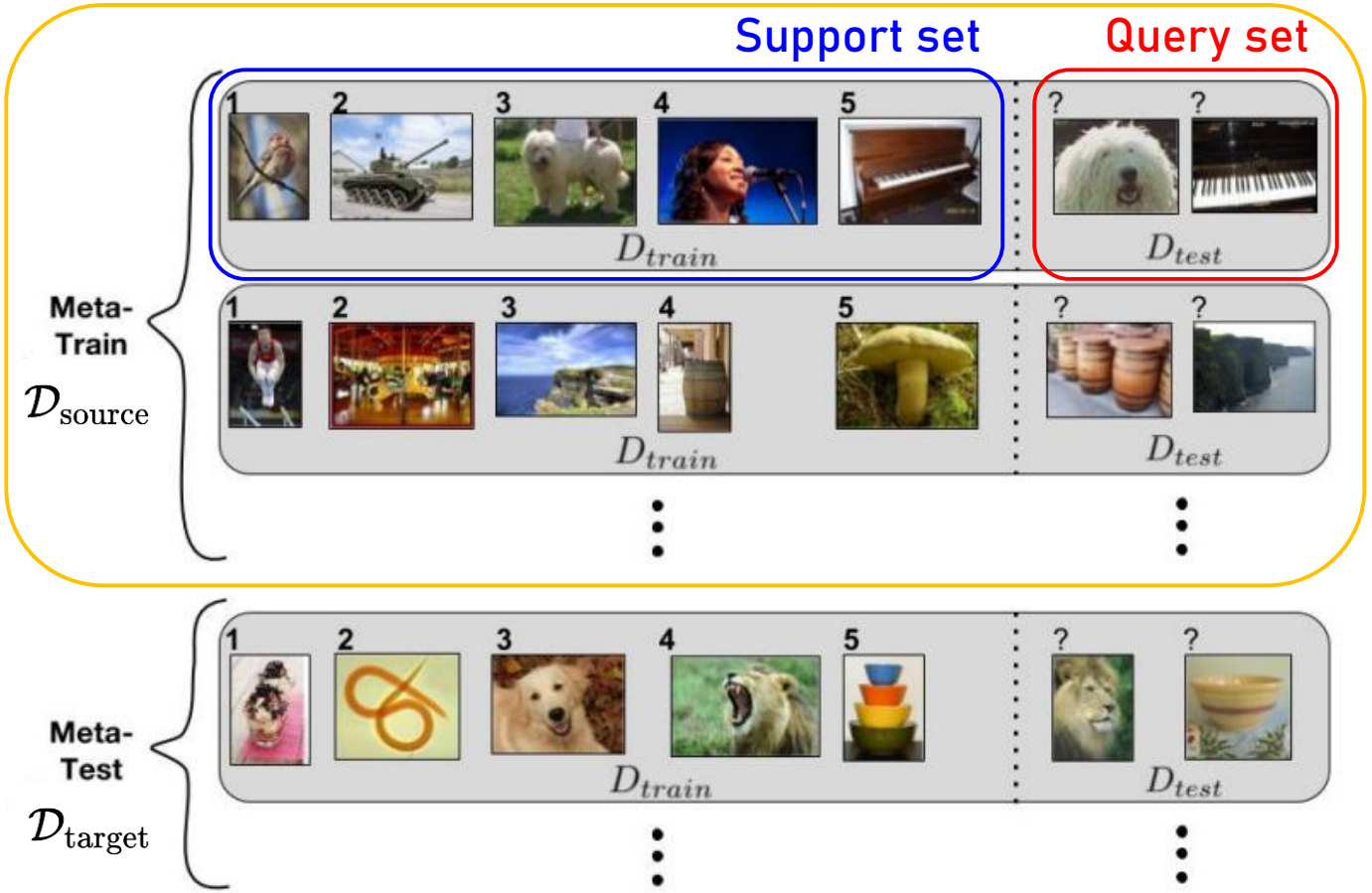
Learning how to learn
(meta-learning)
But how?

$$\omega^* = \arg\max_{\omega} \log p(\omega \mid \mathcal{D}_{\text{source}})$$

Ravi, Sachin, and Hugo Larochelle. "Optimization as a model for few-shot learning." 2016.

Meta-Learning: Overview

- **Meta-training**(Supervised learning)



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

$$\mathcal{D}_{\text{source}} = \left\{ \left(\mathcal{D}_{\text{source}}^{\text{support}}, \mathcal{D}_{\text{source}}^{\text{query}} \right)^{(i)} \right\}$$

Learning how to learn
(meta-learning)
But how?

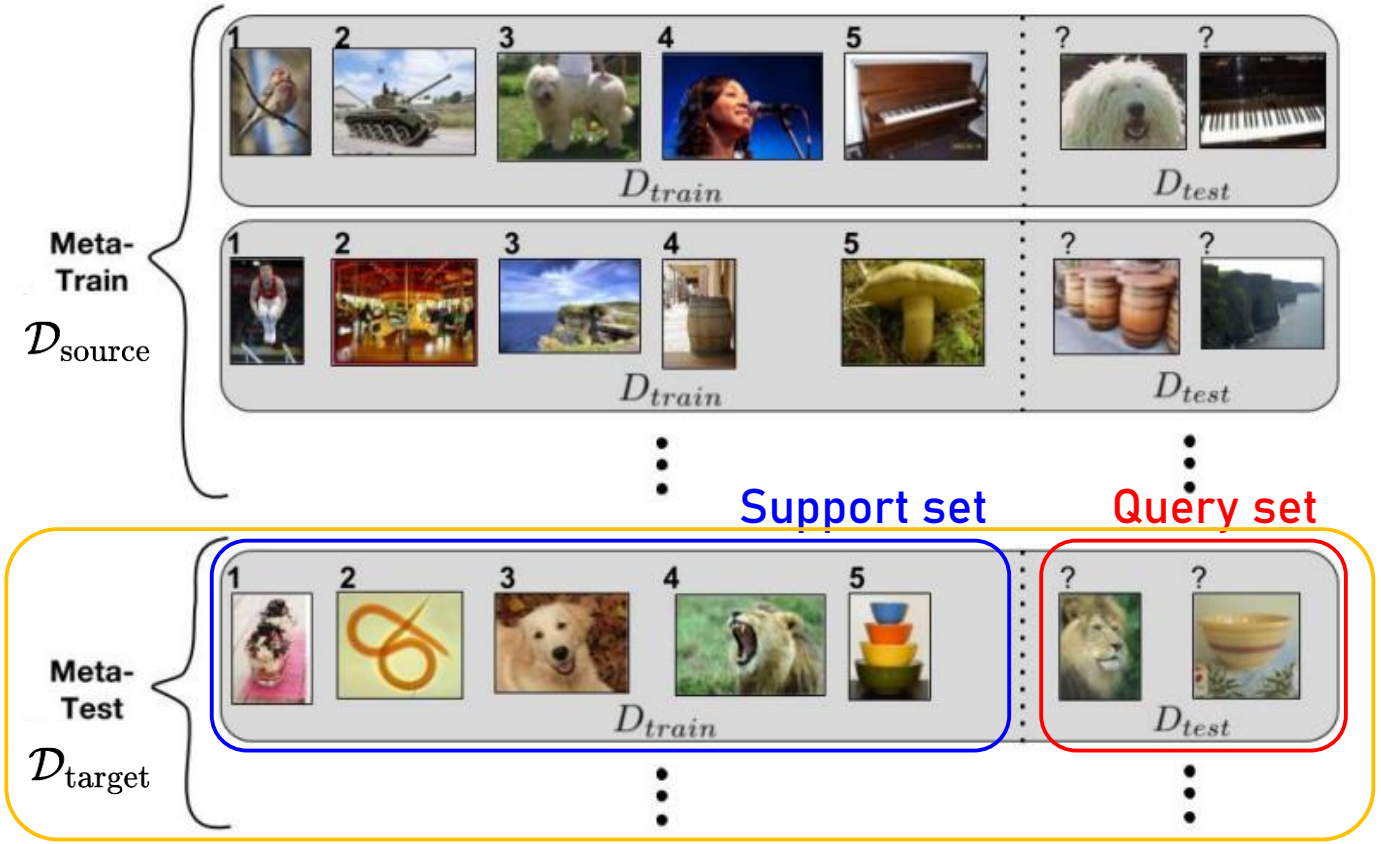
$$\omega^* = \arg\max_{\omega} \log p(\omega \mid \mathcal{D}_{\text{source}})$$

Black-Box / Model-based approach
Optimization-based approach
Non-parametric approach

Ravi, Sachin, and Hugo Larochelle. "Optimization as a model for few-shot learning." 2016.

Meta-Learning: Overview

- Meta-testing(Supervised learning)



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

$$\mathcal{D}_{\text{target}} = \left\{ \left(\mathcal{D}_{\text{target}}^{\text{support}}, \mathcal{D}_{\text{target}}^{\text{query}} \right)^{(i)} \right\}$$

Use meta-knowledge to train the base model on each previously unseen target task

$$\theta^{*(i)} = \arg\max_{\theta} \log p\left(\theta \mid \omega^*, \mathcal{D}_{\text{target}}^{\text{support}(i)} \right)$$

Finally, we can evaluate the accuracy of $\theta^{*(i)}$ our meta-learner by the performance of on the test split of each target task $\mathcal{D}_{\text{target}}^{\text{test(i)}}$

Ravi, Sachin, and Hugo Larochelle. "Optimization as a model for few-shot learning." 2016.

Meta Supervised Learning

Benchmark Datasets in Meta Supervised Learning

- Omniglot Dataset
  - Consists of 20 instances of 1,623 characters from 50 different alphabets
  - Each instance was drawn by a different person

- Mini-ImageNet
  - Proposed by Ravi & Larochelle(2017)
  - Involves 64 training classes, 12 validation classes, and 24 test classes

Meta-learning Algorithms

- Black-Box / Model-based approach (SL 0, RL 0)
  - Santoro, Adam, et al. "Meta-learning with memory-augmented neural networks." *International conference on machine learning*. 2016.
  - Mishra, Nikhil, et al. "A simple neural attentive meta-learner." *arXiv preprint arXiv:1707.03141* (2017).
  - Munkhdalai, Tsendsuren, and Hong Yu. "Meta networks." *Proceedings of machine learning research* 70 (2017): 2554.
- Optimization-based approach (SL 0, RL 0)
  - Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." *arXiv preprint arXiv:1703.03400* (2017).
  - Nichol, Alex, Joshua Achiam, and John Schulman. "On first-order meta-learning algorithms." *arXiv preprint arXiv:1803.02999* (2018).
  - Rusu, Andrei A., et al. "Meta-learning with latent embedding optimization." *arXiv preprint arXiv:1807.05960* (2018).
- Non-parametric approach (SL 0, RL X)
  - Koch, Gregory, Richard Zemel, and Ruslan Salakhutdinov. "Siamese neural networks for one-shot image recognition." *ICML deep learning workshop*. Vol. 2. 2015.
  - Vinyals, Oriol, et al. "Matching networks for one shot learning." *Advances in neural information processing systems*. 2016.
  - Snell, Jake, Kevin Swersky, and Richard Zemel. "Prototypical networks for few-shot learning." *Advances in neural information processing systems*. 2017.

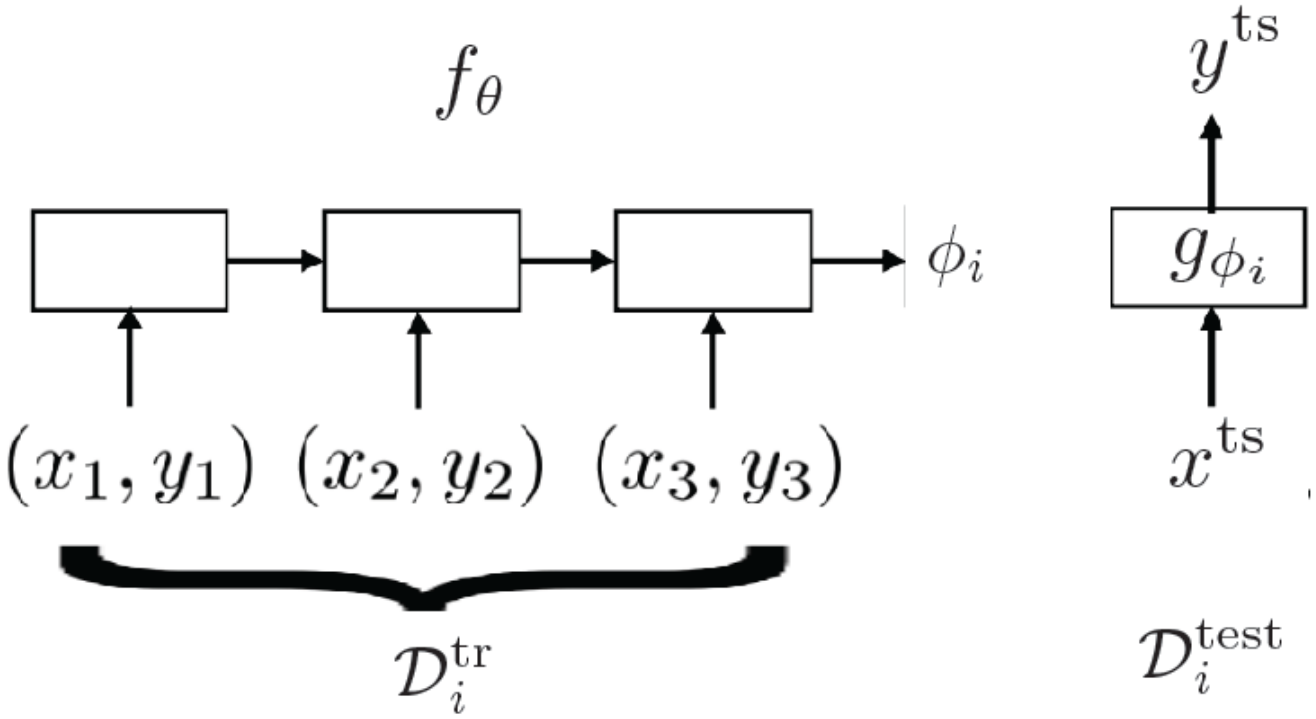
Meta-learning Algorithms

- Black-Box / Model-based approach (SL 0, RL 0)
  - Santoro, Adam, et al. "Meta-learning with memory-augmented neural networks." *International conference on machine learning*. 2016.
  - Mishra, Nikhil, et al. "A simple neural attentive meta-learner." *arXiv preprint arXiv:1707.03141* (2017).
  - Munkhdalai, Tsendsuren, and Hong Yu. "Meta networks." *Proceedings of machine learning research* 70 (2017): 2554.
- Optimization-based approach (SL 0, RL 0)
  - Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." *arXiv preprint arXiv:1703.03400* (2017).
  - Nichol, Alex, Joshua Achiam, and John Schulman. "On first-order meta-learning algorithms." *arXiv preprint arXiv:1803.02999* (2018).
  - Rusu, Andrei A., et al. "Meta-learning with latent embedding optimization." *arXiv preprint arXiv:1807.05960* (2018).
- Non-parametric approach (SL 0, RL X)
  - Koch, Gregory, Richard Zemel, and Ruslan Salakhutdinov. "Siamese neural networks for one-shot image recognition." *ICML deep learning workshop*. Vol. 2. 2015.
  - Vinyals, Oriol, et al. "Matching networks for one shot learning." *Advances in neural information processing systems*. 2016.
  - Snell, Jake, Kevin Swersky, and Richard Zemel. "Prototypical networks for few-shot learning." *Advances in neural information processing systems*. 2017.

Black-Box / Model-based approach

Black-Box / Model-based approach

- General approach: Train a recurrent network to represent $p(\phi_i \mid \mathcal{D}_i^{\text{tr}}, \theta)$



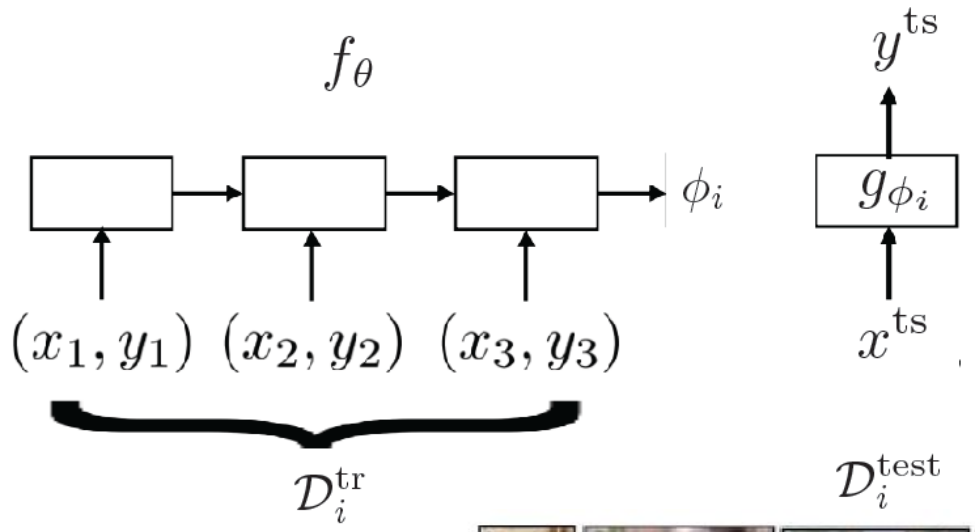
Train with standard supervised learning

$$\max_\theta \sum_{\mathcal{T}_i} \underbrace{\sum_{(x,y)\sim\mathcal{D}_i^{\text{test}}} \log g_{\phi_i}(y|x)}_{\mathcal{L}(\phi_i, \mathcal{D}_i^{\text{test}})}$$

$$\max_\theta \sum_{\mathcal{T}_i} \mathcal{L}(f_\theta(\mathcal{D}_i^{\text{tr}}), \mathcal{D}_i^{\text{test}})$$

Black-Box / Model-based approach

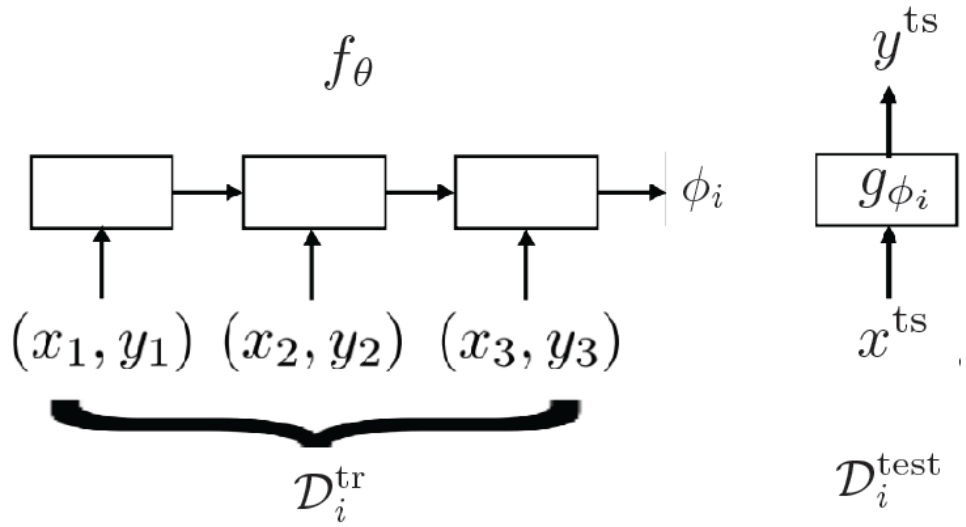
- General approach: Train a recurrent network to represent $p(\phi_i \mid \mathcal{D}_i^{\text{tr}}, \theta)$



1. Sample task $\mathcal{T}_i$ (or mini-batch of tasks)
2. Sample disjoint datasets $\mathcal{D}_i^{\text{tr}}, \mathcal{D}_i^{\text{test}}$ from $\mathcal{D}_i$

Black-Box / Model-based approach

- General approach: Train a recurrent network to represent $p(\phi_i \mid \mathcal{D}_i^{\text{tr}}, \theta)$



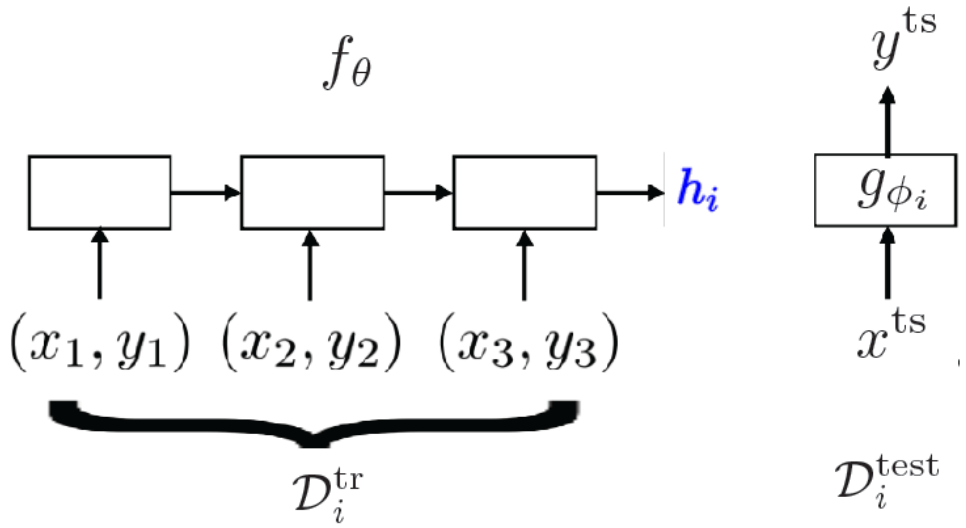
1. Sample task $\mathcal{T}_i$ (or mini-batch of tasks)
2. Sample disjoint datasets $\mathcal{D}_i^{\text{tr}}, \mathcal{D}_i^{\text{test}}$ from $\mathcal{D}_i$
3. Compute $\phi_i \leftarrow f_\theta(\mathcal{D}_i^{\text{tr}})$
4. Update θ using $\nabla_\theta \mathcal{L}(\phi_i, \mathcal{D}_i^{\text{test}})$

$\mathcal{D}_i^{\text{tr}}$ 13 $\mathcal{D}_i^{\text{test}}$

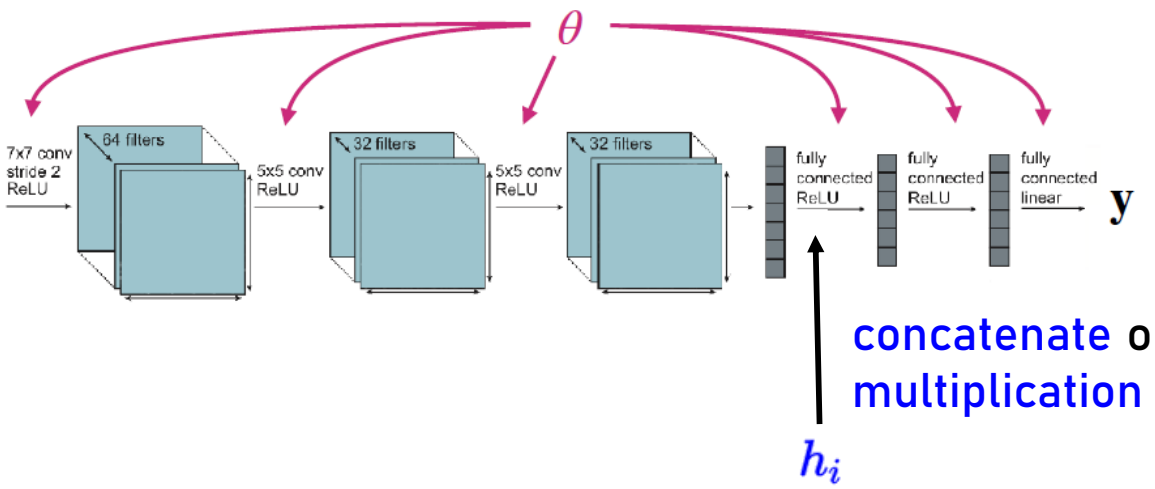
support set query set

Black-Box / Model-based approach

- Challenge: Output parameters does not scalable
  - No need to output all parameters of neural net, only sufficient statistics (Santoro et al., MANN, Mishra et al., SNAIL)



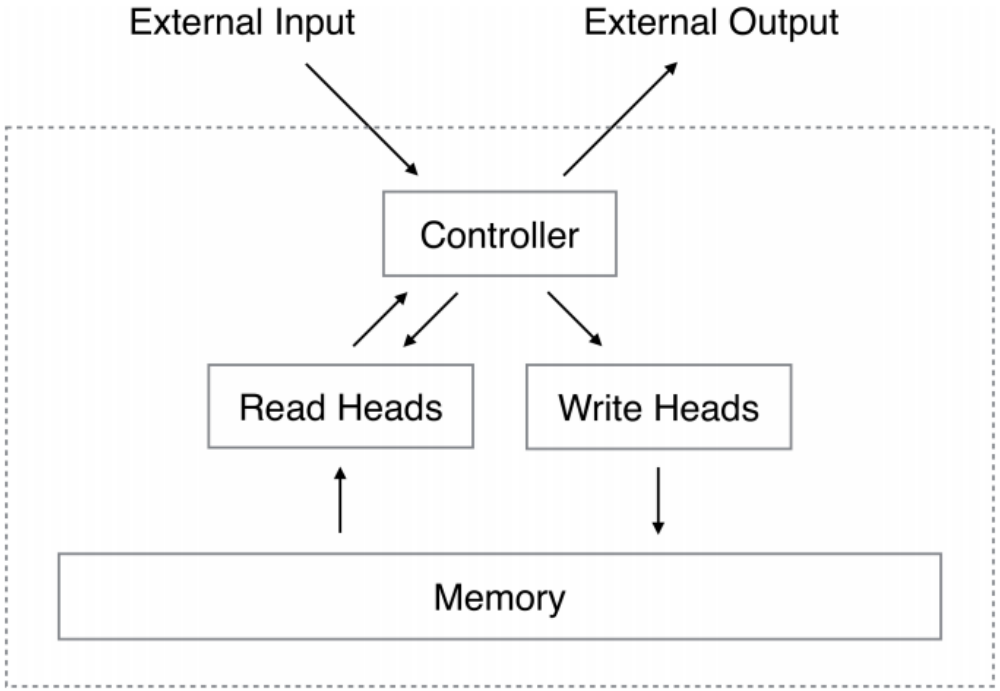
For example, in multi-task learning,

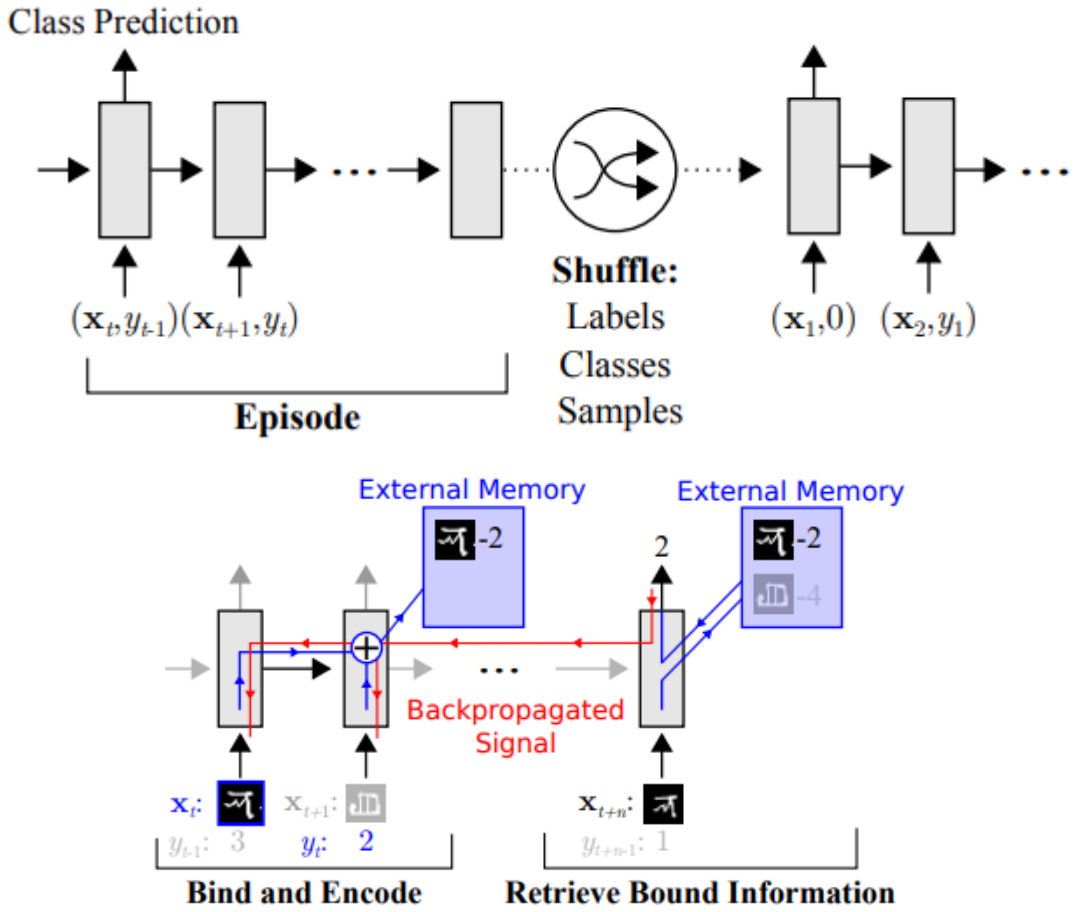


Low-dimensional vector h_i
Represents contextual task information

Black-Box / Model-based approach

- Memory-Augmented Neural Networks

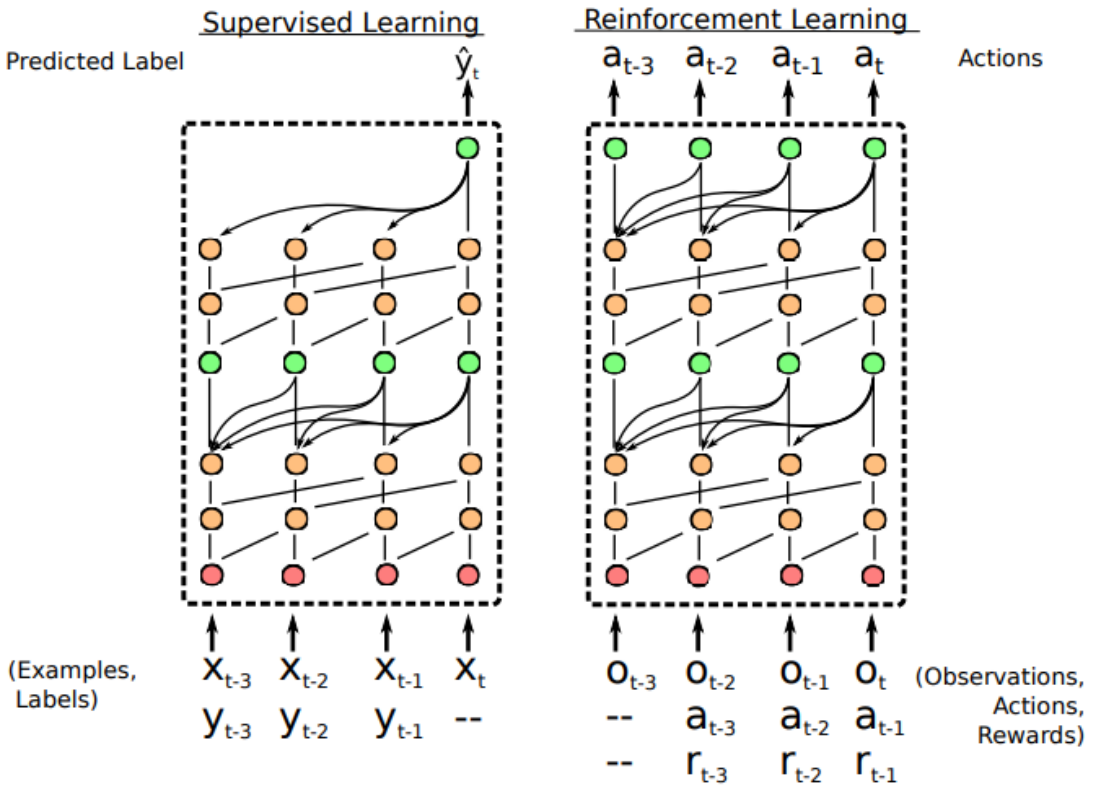




Graves, Alex, et al. "Neural turing machines." *arXiv*, 2014, Santoro, Adam, et al. "Meta-learning with memory-augmented neural networks." ICML 2016.

Black-Box / Model-based approach

- A Simple Neural AttentIve Meta-Learner(SNAIL)



function DENSEBLOCK(inputs, dilation rate R, number of filters D):
    xf, xg = CausalConv(inputs, R, D), CausalConv(inputs, R, D)
    activations = tanh(xf) * sigmoid(xg)
    return concat(inputs, activations)

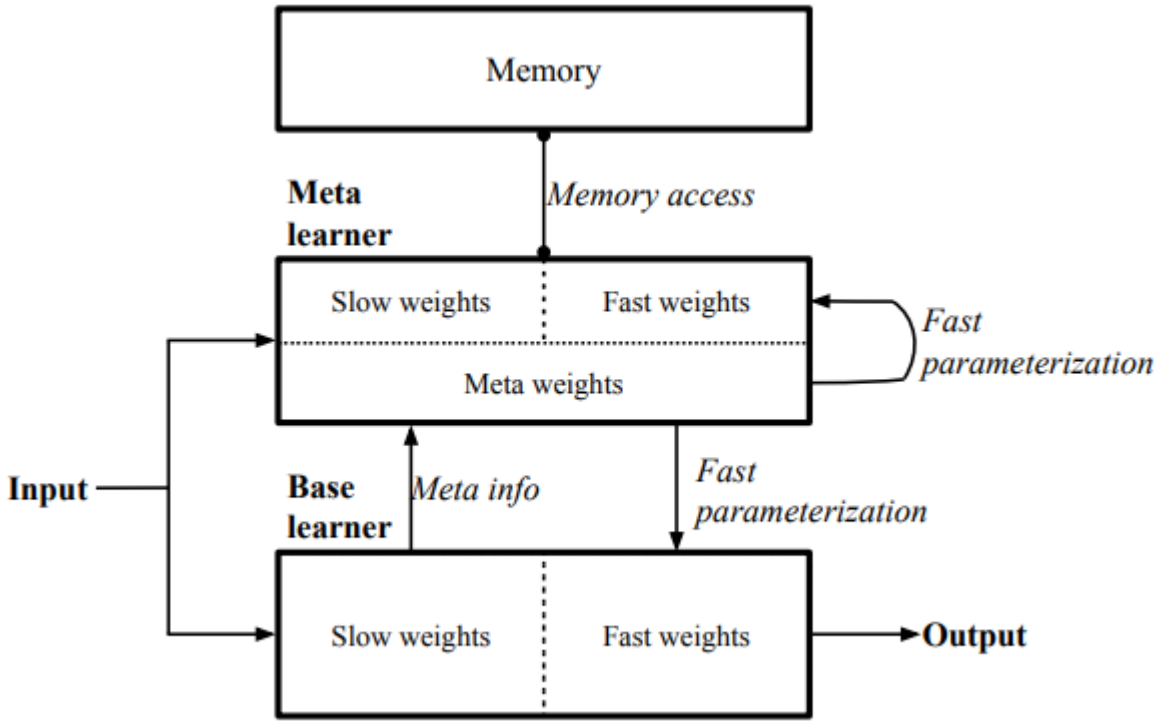
function TCBLOCK(inputs, sequence length T, number of filters D):
    for i in $1, \ldots, \lceil \log_2 T \rceil$ **do**
        inputs = DenseBlock(inputs, 2^i, D)
    return inputs

function ATTENTIONBLOCK(inputs, key size K, value size V):
    keys, query = affine(inputs, K), affine(inputs, K)
    logits = matmul(query, transpose(keys))
    probs = CausallyMaskedSoftmax(logits / $\sqrt{K}$)
    values = affine(inputs, V)
    read = matmul(probs, values)
    return concat(inputs, read)

Mishra, Nikhil, et al. "A simple neural attentive meta-learner." ICLR 2018

Black-Box / Model-based approach

- Meta Networks



Algorithm 1 MetaNet for one-shot supervised learning

Require: Support set $\{x'_i, y'_i\}_{i=1}^{N}$ and Training set $\{x_i, y_i\}_{i=1}^{L}$
Require: Base learner b, Dynamic representation learning function u, Fast weight generation functions m and d, and Slow weights $\theta = \{W, Q, Z, G\}$
Require: Layer augmentation scheme

Sample T examples from support set
for $i = 1, T$ **do**
  $\mathcal{L}_i \leftarrow loss_{emb}(u(Q, x'_i), y'_i)$
  $\nabla_i \leftarrow \nabla_Q \mathcal{L}_i$
end for
$Q^* = d(G, \{\nabla\}_{i=1}^{T})$
for $i = 1, N$ **do**
  $\mathcal{L}_i \leftarrow loss_{task}(b(W, x'_i), y'_i)$
  $\nabla_i \leftarrow \nabla_W \mathcal{L}_i$
  $W_i^* \leftarrow m(Z, \nabla_i)$
  Store W_i^* in i^{th} position of memory M
  $r'_i = u(Q, Q^*, x'_i)$
  Store r'_i in i^{th} position of index memory R
end for
$\mathcal{L}_{train} = 0$
for $i = 1, L$ **do**
  $r_i = u(Q, Q^*, x_i)$
  $a_i = attention(R, r_i)$
  $W_i^* = softmax(a_i)^\top M$
  $\mathcal{L}_{train} \leftarrow \mathcal{L}_{train} + loss_{task}(b(W, W_i^*, x_i), y_i)$
  {Alternatively the base learner can take as input r_i instead of x_i}
end for
Update θ using $\nabla_\theta \mathcal{L}_{train}$

Munkhdalai, Tsendsuren, and Hong Yu. "Meta networks." *Proceedings of machine learning research* 2017

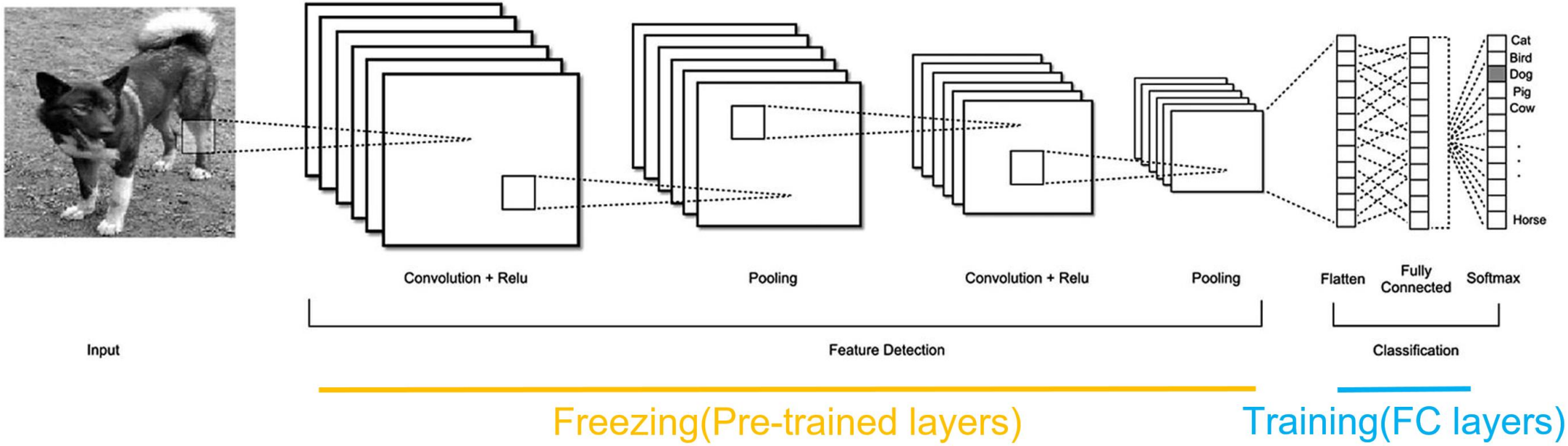
Black-Box / Model-based approach: Further Readings

- Graves, Alex, Greg Wayne, and Ivo Danihelka. "Neural turing machines." *arXiv preprint a rXiv:1410.5401* (2014).
- Weston, Jason, Sumit Chopra, and Antoine Bordes. "Memory networks." *arXiv preprint arXiv:1410.3916* (2014).
- Santoro, Adam, et al. "Meta-learning with memory-augmented neural networks." *International conference on machine learning*. 2016.
- Mishra, Nikhil, et al. "A simple neural attentive meta-learner." *arXiv preprint arXiv:1707.03141* (2017).
- Munkhdalai, Tsendsuren, and Hong Yu. "Meta networks." *Proceedings of machine learning research* 70 (2017): 2554.
- Garnelo, Marta, et al. "Conditional neural processes." *arXiv preprint arXiv:1807.01613* (2018).

Optimization-based approach

Optimization-based approach

- Can we utilize the pre-trained prior knowledge?
- One successful form of prior knowledge: initialization for fine-tuning
- What about Transfer Learning?



Can we do better?

Optimization-based approach

- Model-Agnostic Meta Learning

pre-trained parameters

Fine-tuning [test-time]

$$\phi \leftarrow \boldsymbol{\theta} - \alpha \nabla_\theta \mathcal{L}(\theta, \mathcal{D}^{\mathrm{tr}})$$

training data for new task

Meta-learning

$$\min_\theta \sum_{\text{task } i} \mathcal{L}(\theta - \alpha \nabla_\theta \mathcal{L}(\theta, \mathcal{D}_i^{\mathrm{tr}}), \mathcal{D}_i^{\mathrm{ts}})$$

Key idea: Over many tasks, learn parameter vector θ that transfers via fine-tuning

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

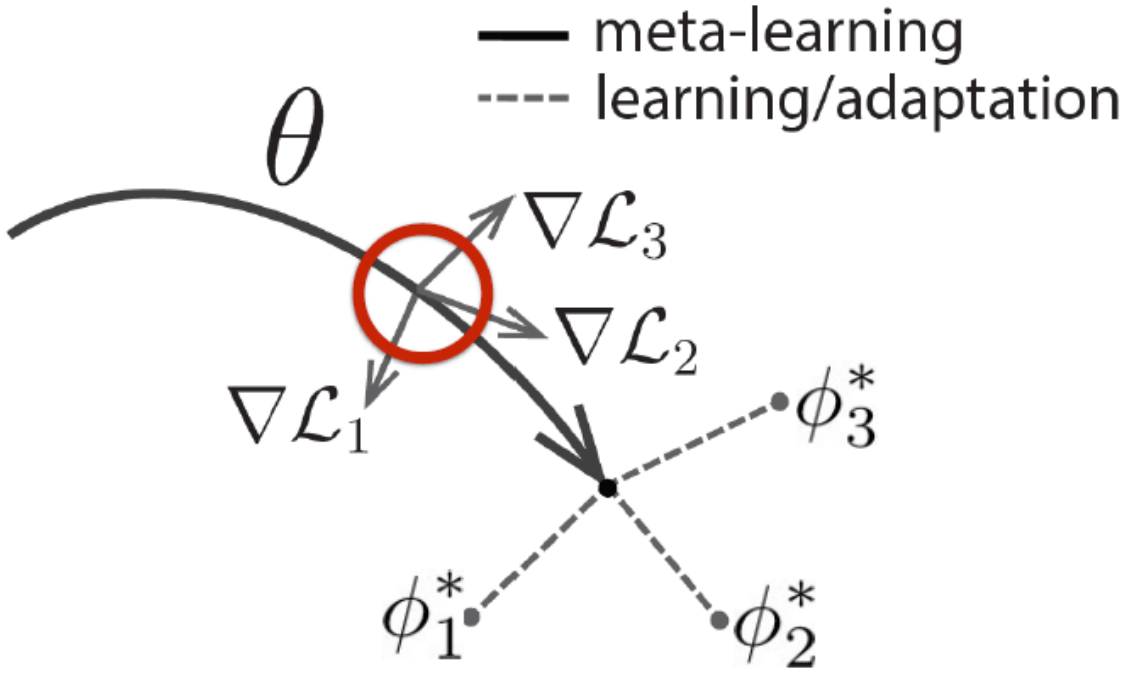
Optimization-based approach

- Model-Agnostic Meta Learning

$$\min_{\theta} \sum_{\text{task } i} \mathcal{L}(\theta - \alpha \nabla_{\theta} \mathcal{L}(\theta, \mathcal{D}_i^{\text{tr}}), \mathcal{D}_i^{\text{ts}})$$

θ parameter vector being meta-learned

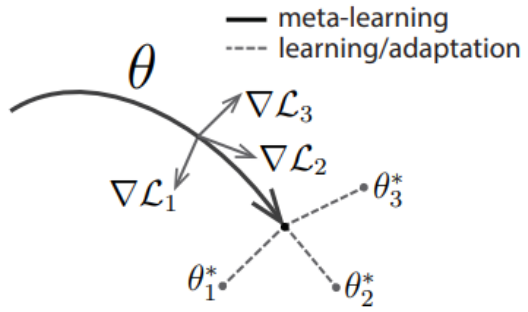
ϕ_i^* optimal parameter vector for task i

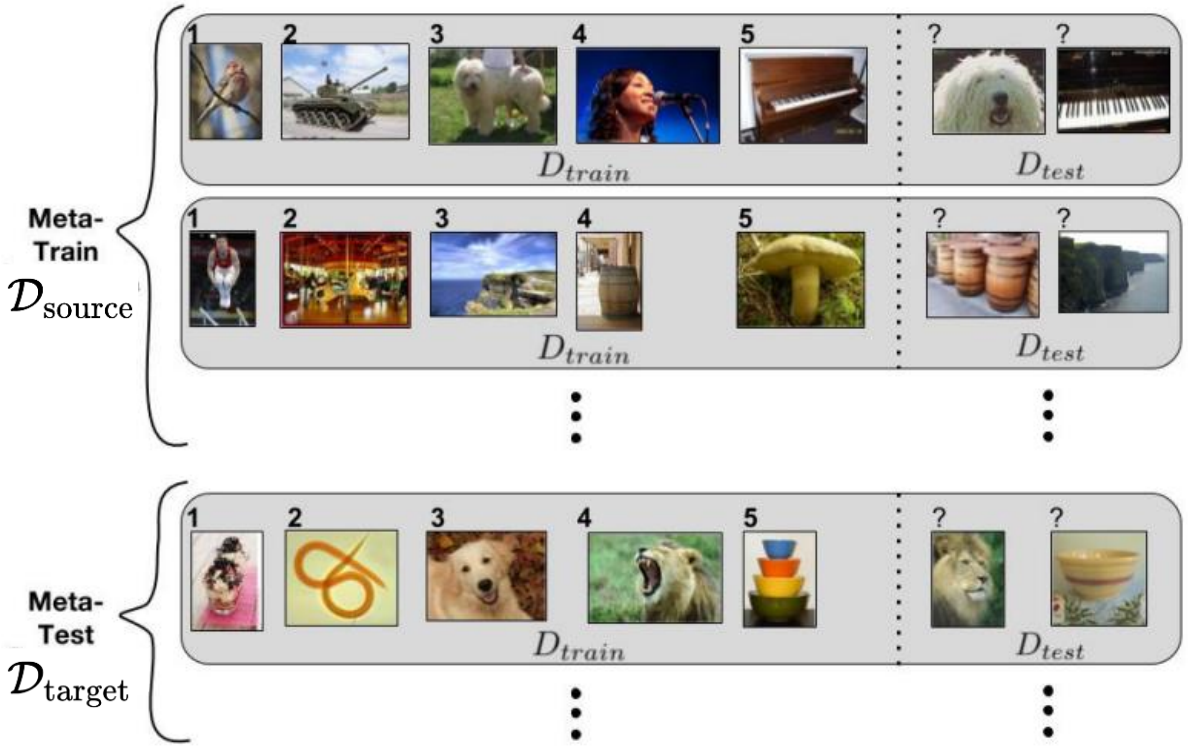


Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta Learning(Supervised Learning)





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

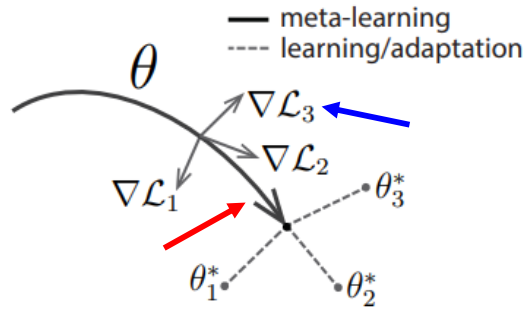
Algorithm 2 MAML for Few-Shot Supervised Learning

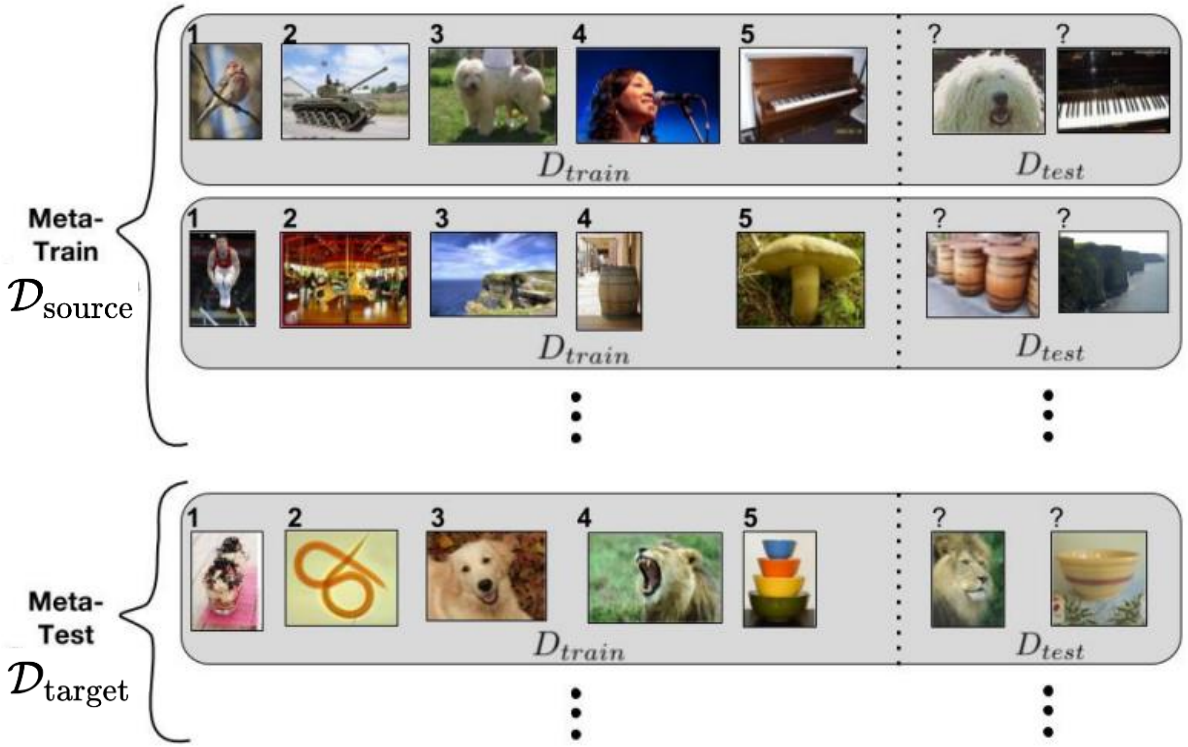
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K datapoints $\mathcal{D} = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ 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**

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta Learning(Supervised Learning)





Algorithm 2 MAML for Few-Shot Supervised Learning

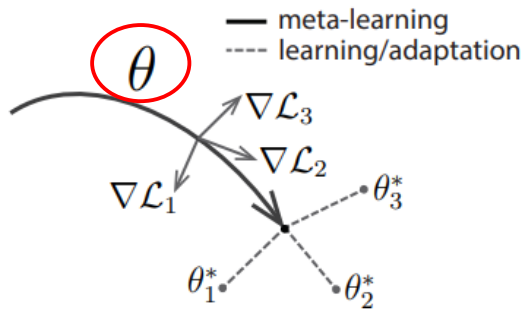
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ — meta-initialization(outer-loop)
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$ — adaptation(inner-loop)
4: **for all** $\mathcal{T}_i$ **do**
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**

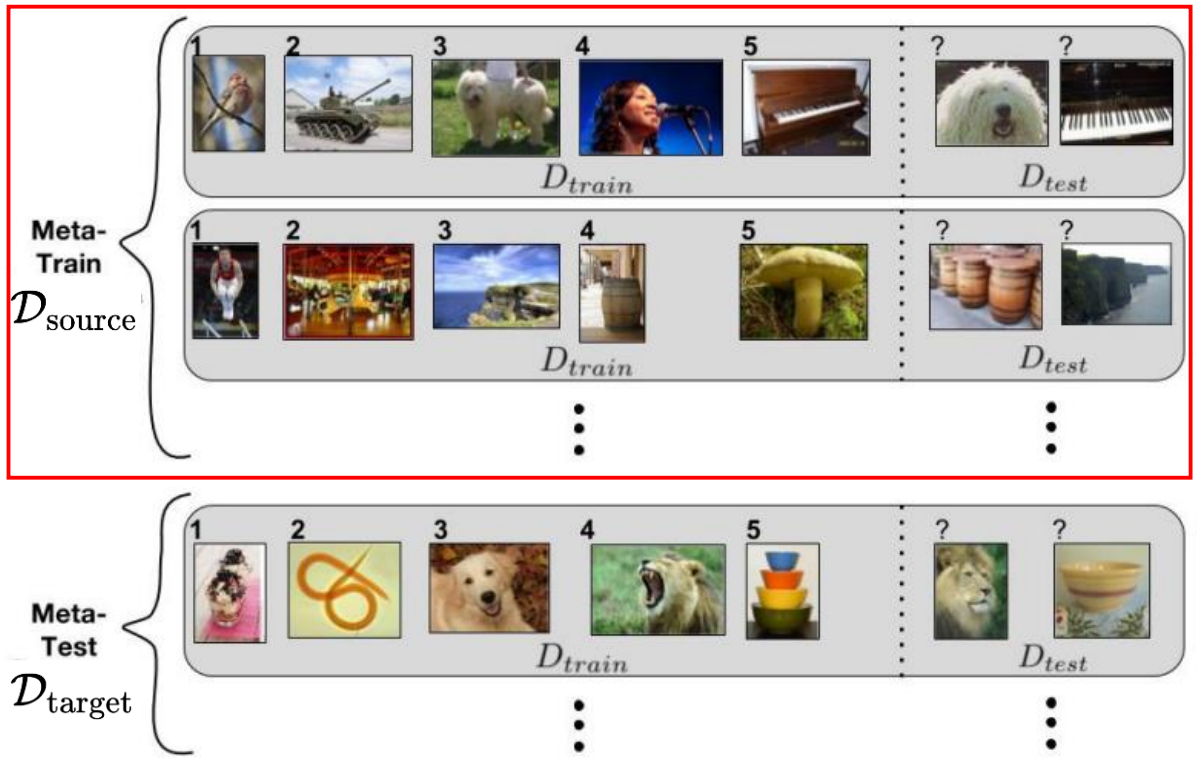
$$\mathcal{L}_{\mathcal{T}_i}(f_\phi) = \sum_{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}} \mathbf{y}^{(j)} \log f_\phi\left(\mathbf{x}^{(j)}\right) + \left(1 - \mathbf{y}^{(j)}\right) \log\left(1 - f_\phi\left(\mathbf{x}^{(j)}\right)\right)$$

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta Learning(Supervised Learning)





Algorithm 2 MAML for Few-Shot Supervised Learning

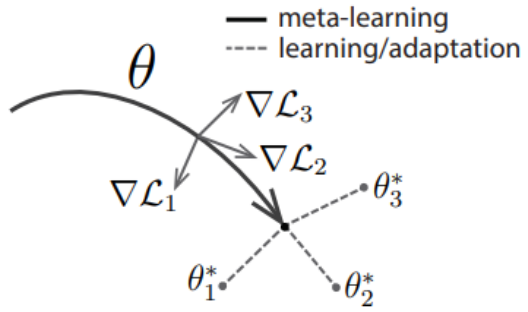
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K datapoints $\mathcal{D} = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ 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**

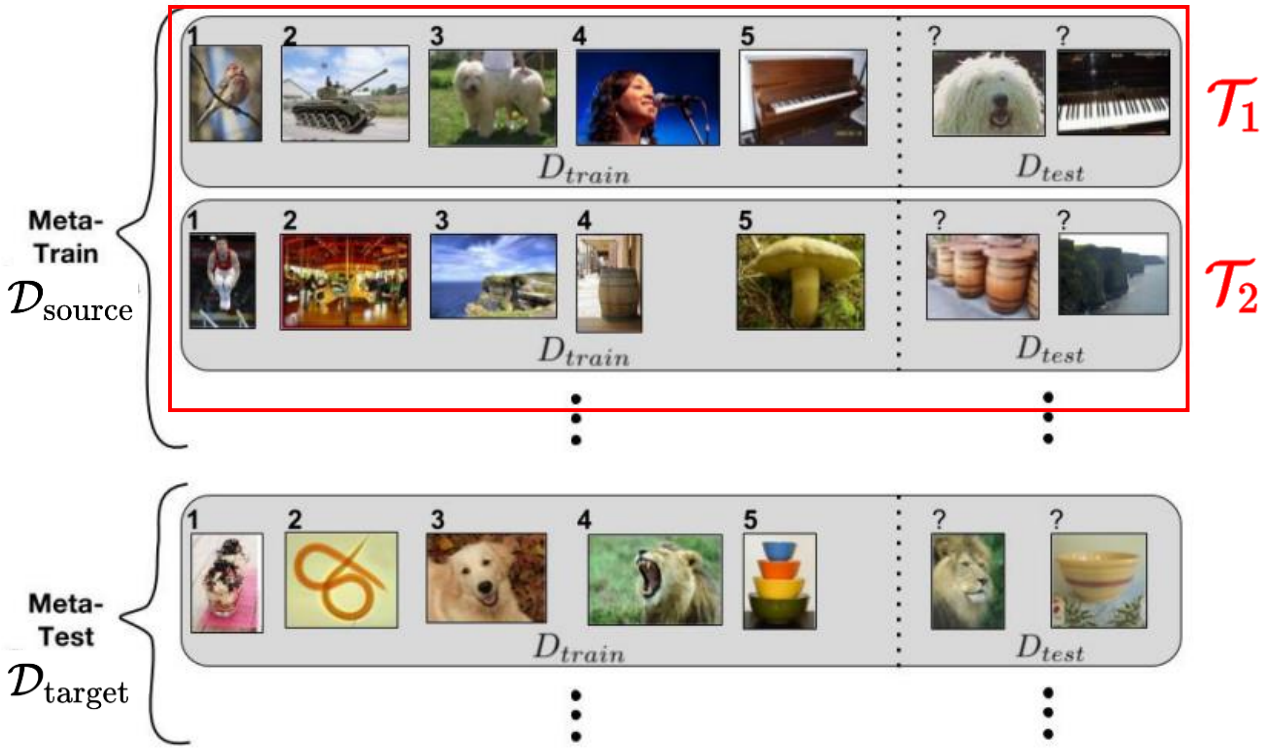
$$\mathcal{L}_{\mathcal{T}_i}(f_\phi) = \sum_{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}} \mathbf{y}^{(j)} \log f_\phi\left(\mathbf{x}^{(j)}\right) + \left(1 - \mathbf{y}^{(j)}\right) \log\left(1 - f_\phi\left(\mathbf{x}^{(j)}\right)\right)$$

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta Learning(Supervised Learning)





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

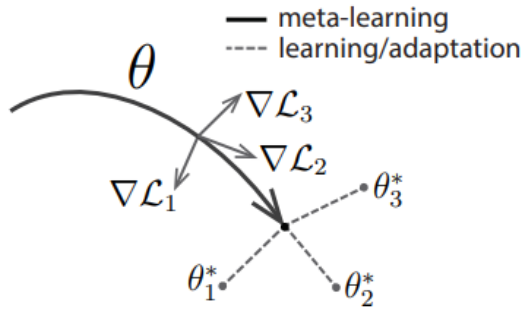
Algorithm 2 MAML for Few-Shot Supervised Learning

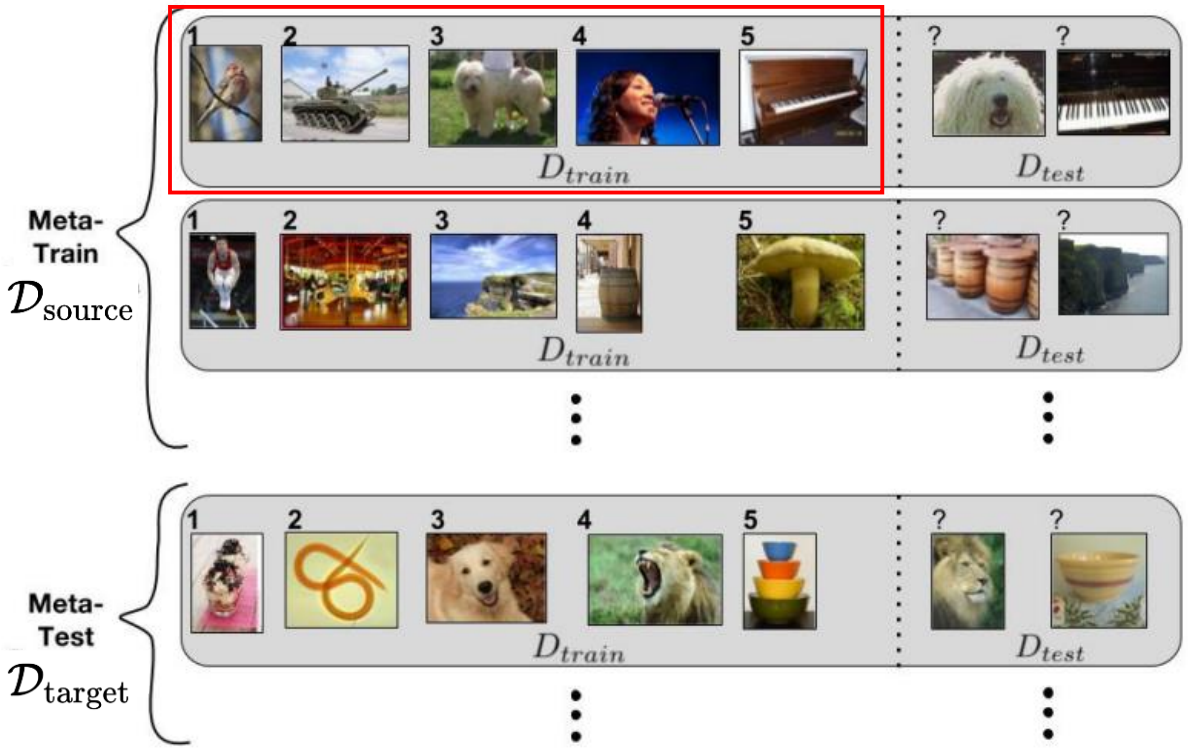
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K datapoints $\mathcal{D} = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ 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**

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta Learning(Supervised Learning)





Algorithm 2 MAML for Few-Shot Supervised Learning

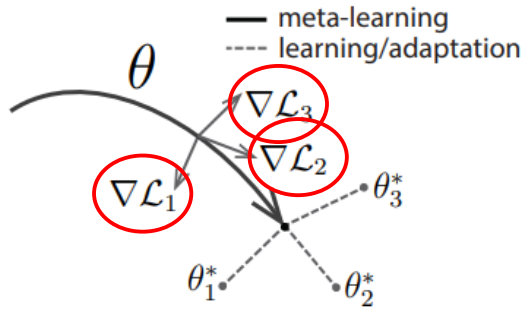
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K datapoints $\mathcal{D} = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ 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**

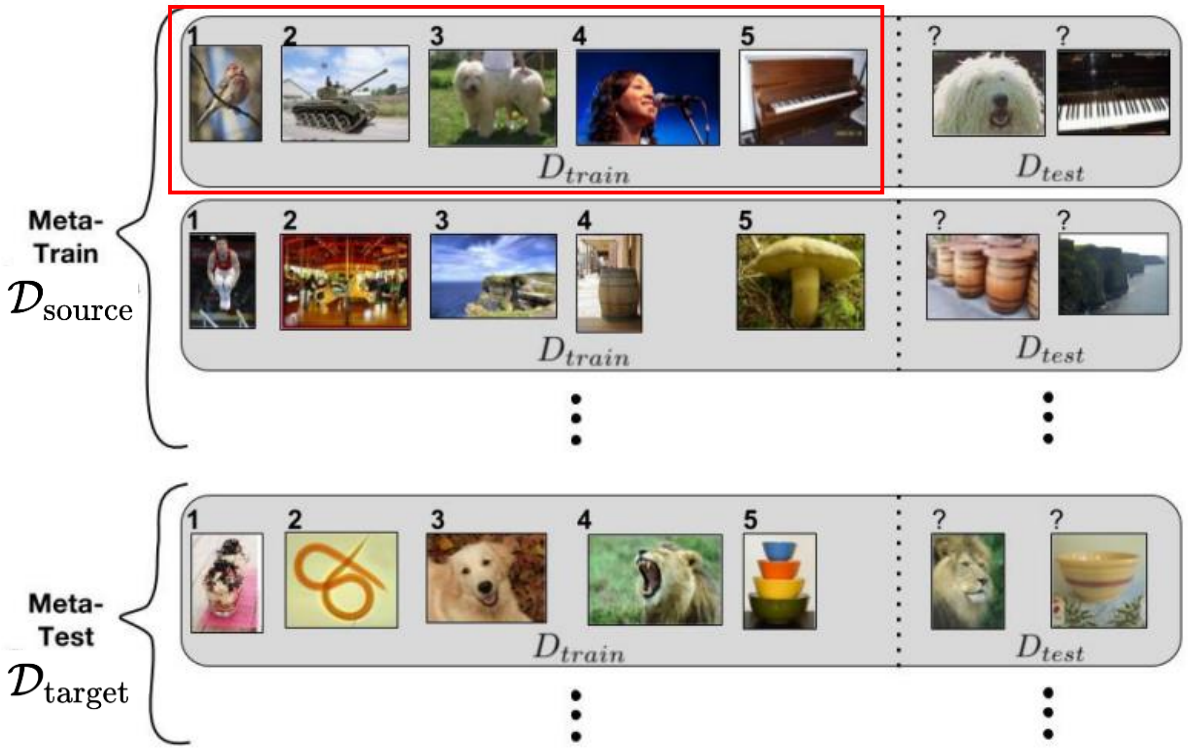
$$\mathcal{L}_{\mathcal{T}_i}(f_\phi) = \sum_{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}} \mathbf{y}^{(j)} \log f_\phi\left(\mathbf{x}^{(j)}\right) + \left(1 - \mathbf{y}^{(j)}\right) \log\left(1 - f_\phi\left(\mathbf{x}^{(j)}\right)\right)$$

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta Learning(Supervised Learning)





Algorithm 2 MAML for Few-Shot Supervised Learning

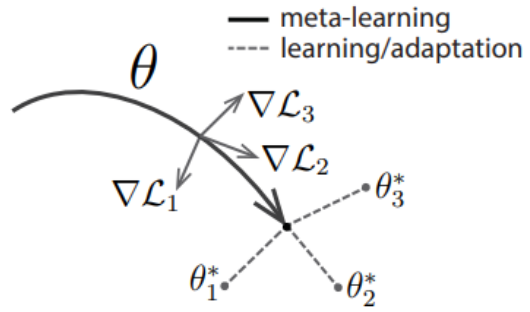
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K datapoints $\mathcal{D} = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ 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**

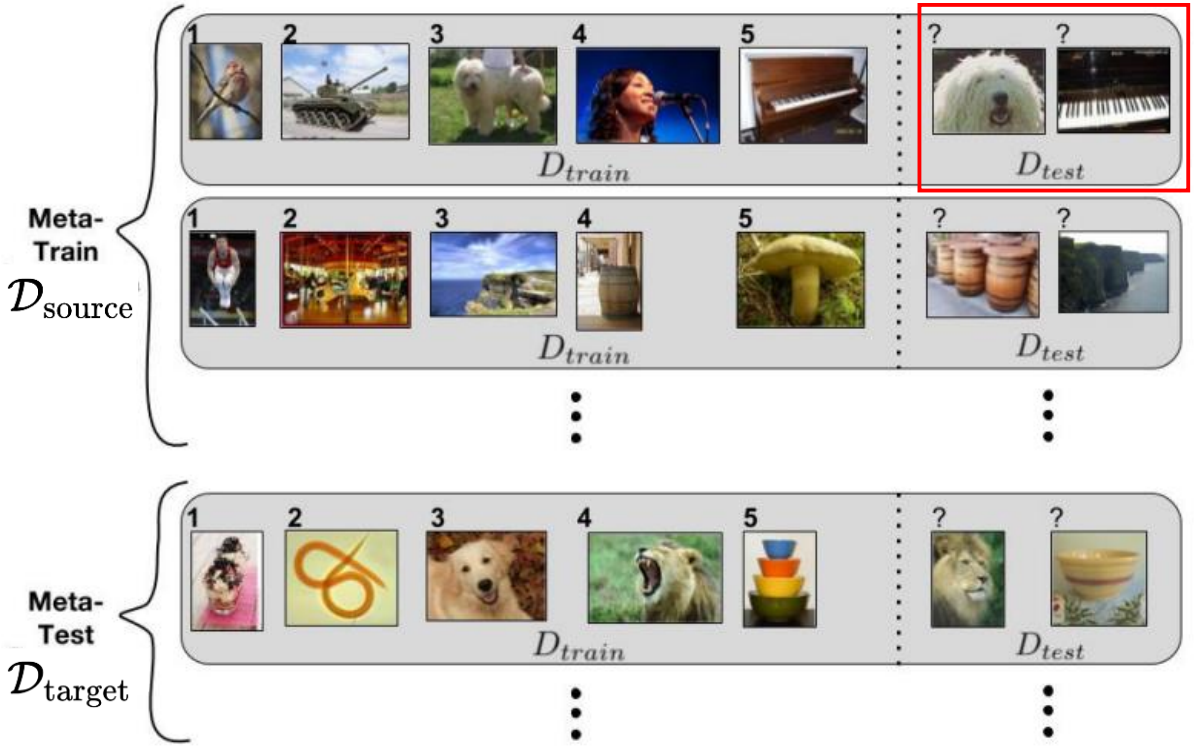
$$\mathcal{L}_{\mathcal{T}_i}(f_\phi) = \sum_{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}} \mathbf{y}^{(j)} \log f_\phi\left(\mathbf{x}^{(j)}\right) + \left(1 - \mathbf{y}^{(j)}\right) \log\left(1 - f_\phi\left(\mathbf{x}^{(j)}\right)\right)$$

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta Learning(Supervised Learning)





Algorithm 2 MAML for Few-Shot Supervised Learning

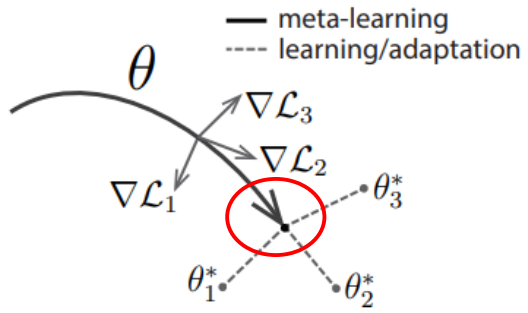
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K datapoints $\mathcal{D} = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ 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**

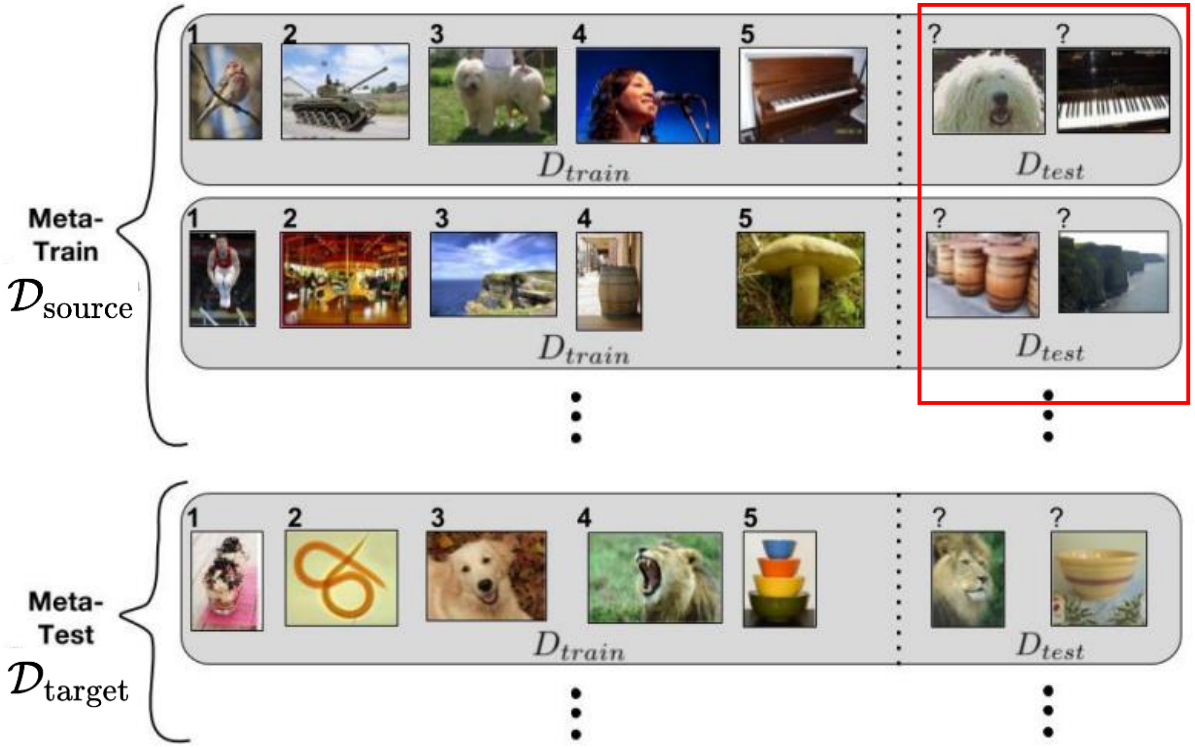
$$\mathcal{L}_{\mathcal{T}_i}(f_\phi) = \sum_{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}} \mathbf{y}^{(j)} \log f_\phi\left(\mathbf{x}^{(j)}\right) + \left(1 - \mathbf{y}^{(j)}\right) \log\left(1 - f_\phi\left(\mathbf{x}^{(j)}\right)\right)$$

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta Learning(Supervised Learning)





Algorithm 2 MAML for Few-Shot Supervised Learning

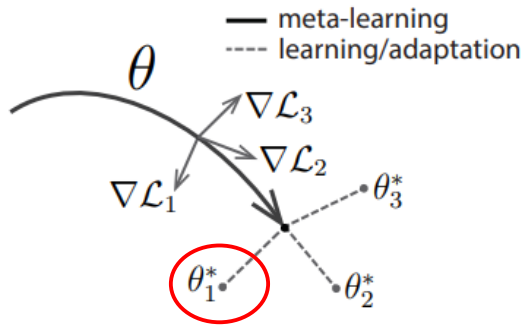
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K datapoints $\mathcal{D} = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ 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**

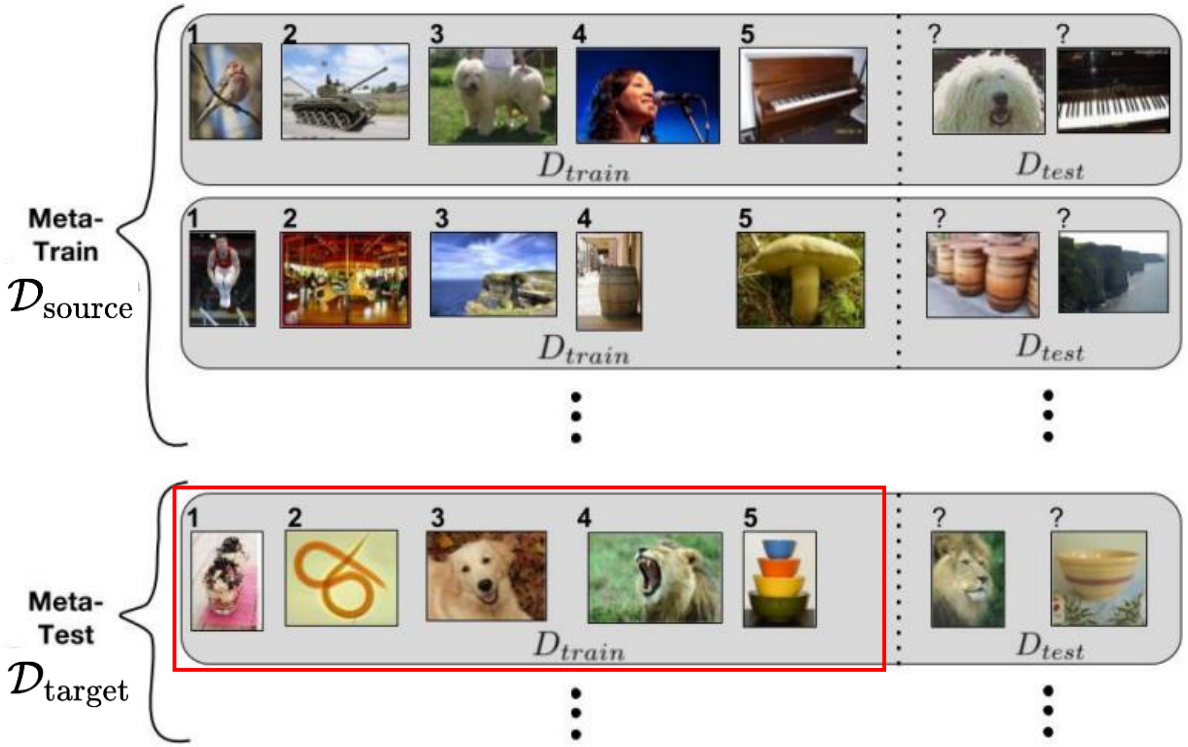
$$\mathcal{L}_{\mathcal{T}_i}(f_\phi) = \sum_{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}} \mathbf{y}^{(j)} \log f_\phi\left(\mathbf{x}^{(j)}\right) + \left(1 - \mathbf{y}^{(j)}\right) \log\left(1 - f_\phi\left(\mathbf{x}^{(j)}\right)\right)$$

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta Learning(Supervised Learning)





Fine Tuning(Adaptation)

$$\theta_1^* \leftarrow \boldsymbol{\theta} - \alpha \nabla_\theta \mathcal{L}\left(\theta, \mathcal{D}_{target}^{\text{support}}\right)$$

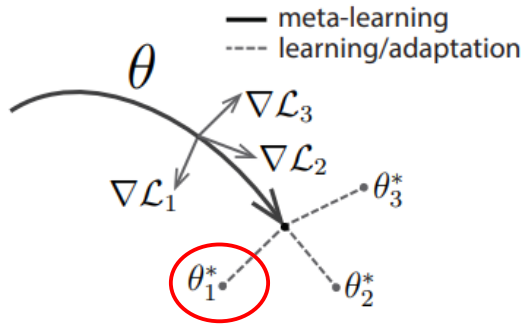
Just 1 or few gradient steps

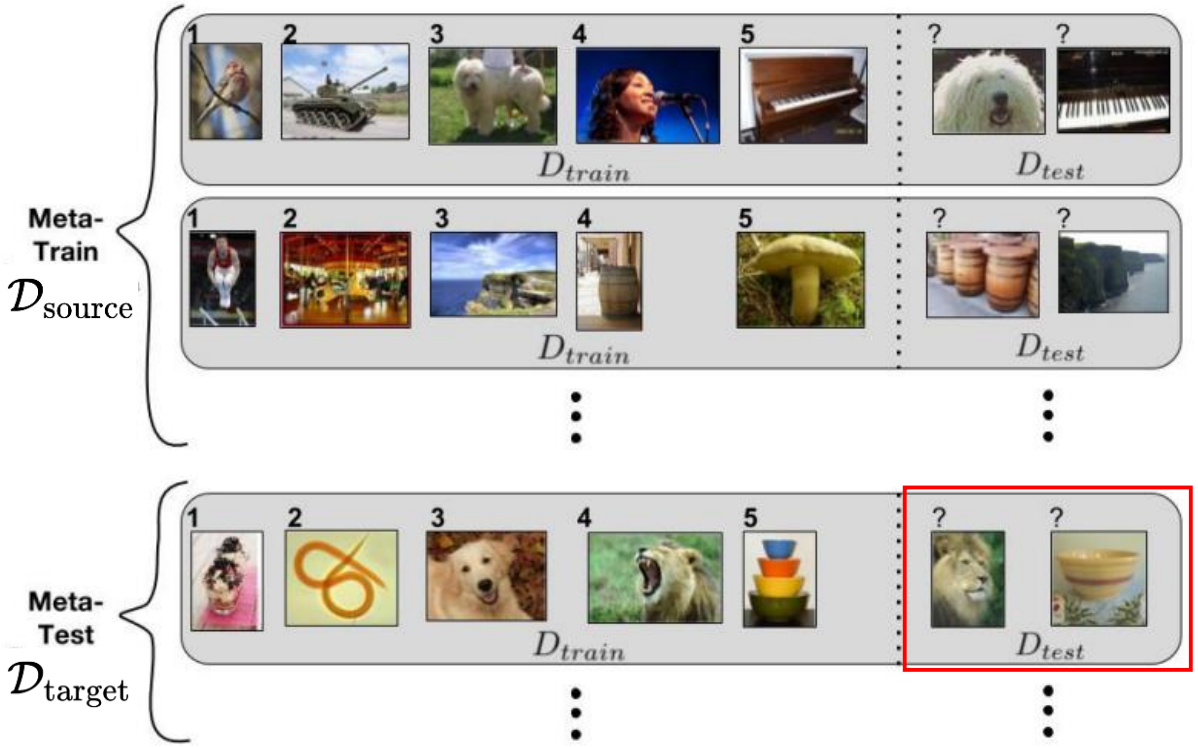
$$\mathcal{L}_{\mathcal{T}_i}(f_\phi) = \sum_{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}} \mathbf{y}^{(j)} \log f_\phi\left(\mathbf{x}^{(j)}\right) + \left(1 - \mathbf{y}^{(j)}\right) \log\left(1 - f_\phi\left(\mathbf{x}^{(j)}\right)\right)$$

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta Learning(Supervised Learning)





Fine Tuning(Adaptation)

$$\theta_1^* \leftarrow \boldsymbol{\theta} - \alpha \nabla_\theta \mathcal{L}\left(\theta, \mathcal{D}_{target}^{\text{support}}\right)$$

Just 1 or few gradient steps

Evaluation

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

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

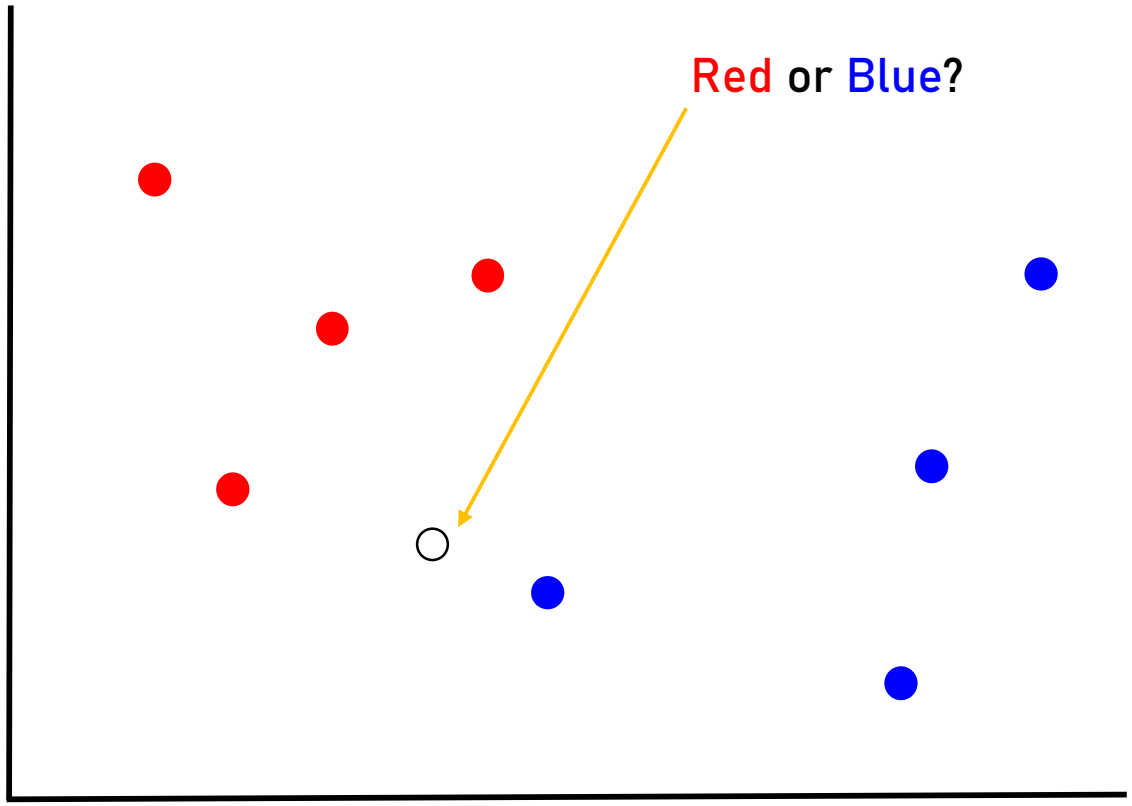
Optimization-based approach: Further Readings

- Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." *arXiv preprint arXiv:1703.03400* (2017).
- Ravi, Sachin, and Hugo Larochelle. "Optimization as a model for few-shot learning." (2016).
- Nichol, Alex, Joshua Achiam, and John Schulman. "On first-order meta-learning algorithms." *arXiv preprint arXiv:1803.02999* (2018).
- Rusu, Andrei A., et al. "Meta-learning with latent embedding optimization." *arXiv preprint arXiv:1807.05960* (2018).
- Antoniou, Antreas, Harrison Edwards, and Amos Storkey. "How to train your MAML." *arXiv preprint arXiv:1810.09502* (2018).
- Lee, Yoonho, and Seungjin Choi. "Gradient-based meta-learning with learned layerwise metric and subspace." *arXiv preprint arXiv:1801.05558* (2018).
- Fallah, Alireza, Aryan Mokhtari, and Asuman Ozdaglar. "On the convergence theory of gradient-based model-agnostic meta-learning algorithms." *International Conference on Artificial Intelligence and Statistics*. 2020.
- Song, Xingyou, et al. "Es-maml: Simple hessian-free meta learning." *arXiv preprint arXiv:1910.01215* (2019).
- Triantafillou, Eleni, et al. "Meta-dataset: A dataset of datasets for learning to learn from few examples." *arXiv preprint arXiv:1903.03096* (2019).

Non-parametric approach

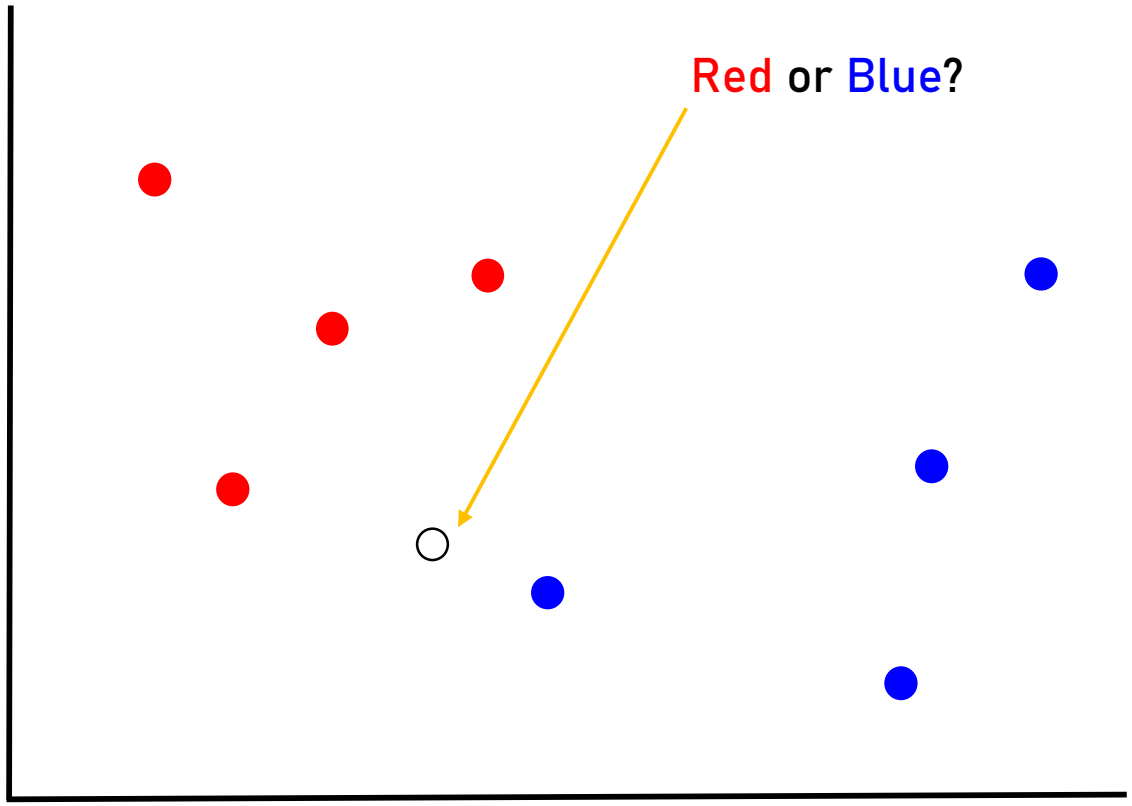
Non-parametric approach

- K-NN(K-Nearest Neighbor) Algorithm
  → Lazy & Non-parametric Learner



Non-parametric approach

- K-NN(K-Nearest Neighbor) Algorithm
  → Lazy & Non-parametric Learner



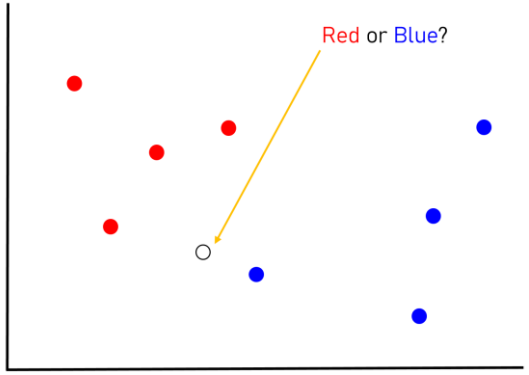
1-Nearest Neighbor
→ Blue

3-Nearset Neighbor
→ Red

5-Nearset Neighbor
→ Red

Non-parametric approach

- In low data regimes(e.g. 2D data), non-parametric methods are simple, works well



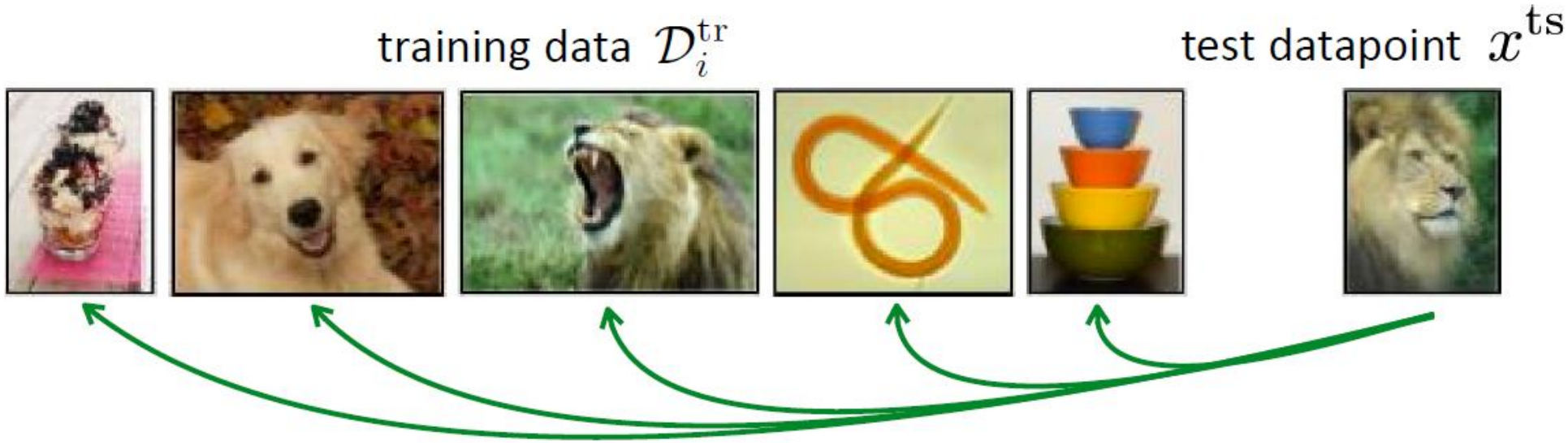
- What about high dimensional data?
  - During meta-training
    - Still want to be parametric
  - During meta-testing
    - Few-shot learning ↔ low data regime

Can we train parametric meta-learners that produce effective non-parametric learners?

Finn, Chelsea, Sergey Levine. “Meta-Learning: from Few-Shot Learning to Rapid Reinforcement Learning." ICML Tutorial 2019

Non-parametric approach

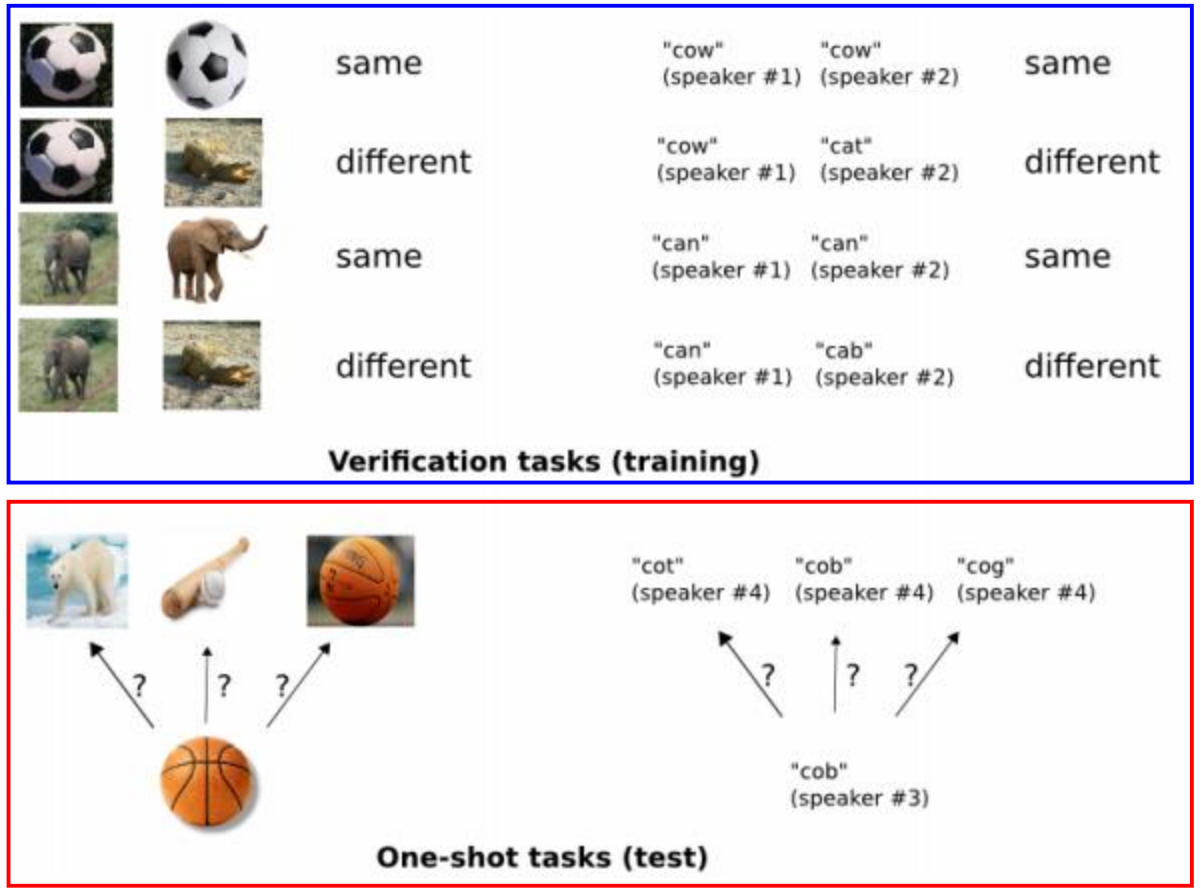
- At meta-test time, Compare test image(query) with training images(support set)
  - Which space do we use?
  - Which distance metric do we use?



Finn, Chelsea, Sergey Levine. “Meta-Learning: from Few-Shot Learning to Rapid Reinforcement Learning." ICML Tutorial 2019

Non-parametric approach

- Siamese Neural Networks



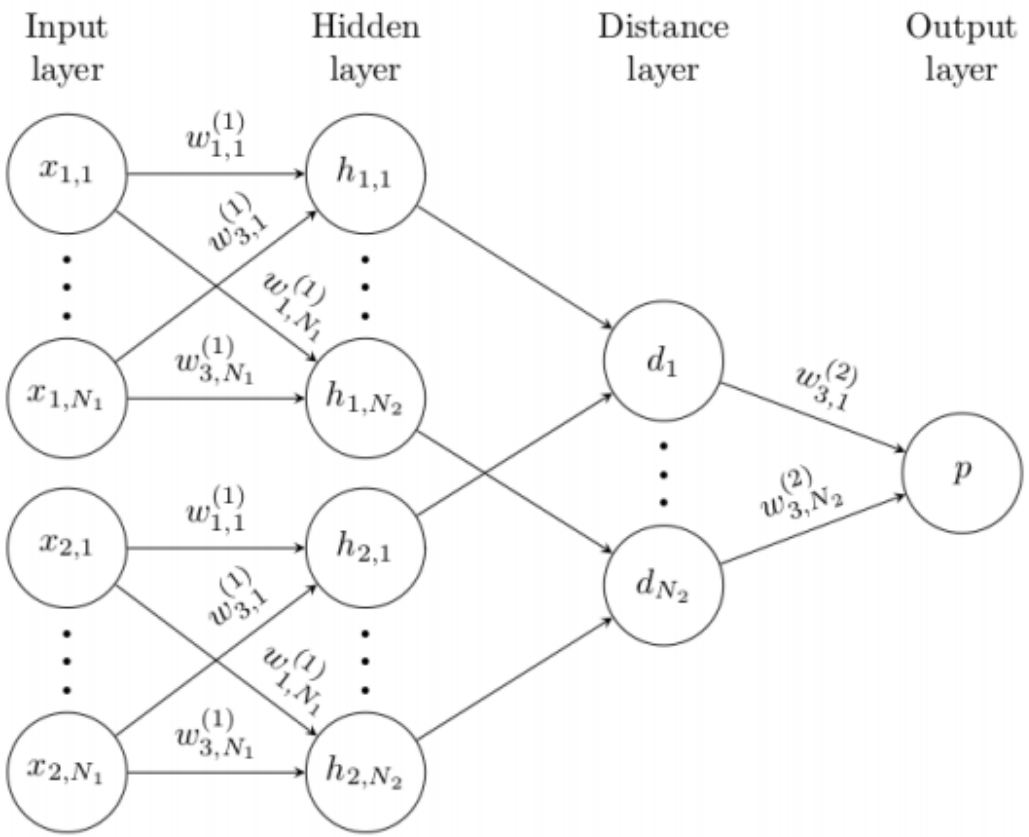
meta-training

meta-testing

Koch, Gregory, et al. "Siamese neural networks for one-shot image recognition." *ICML deep learning workshop*, 2015.

Non-parametric approach

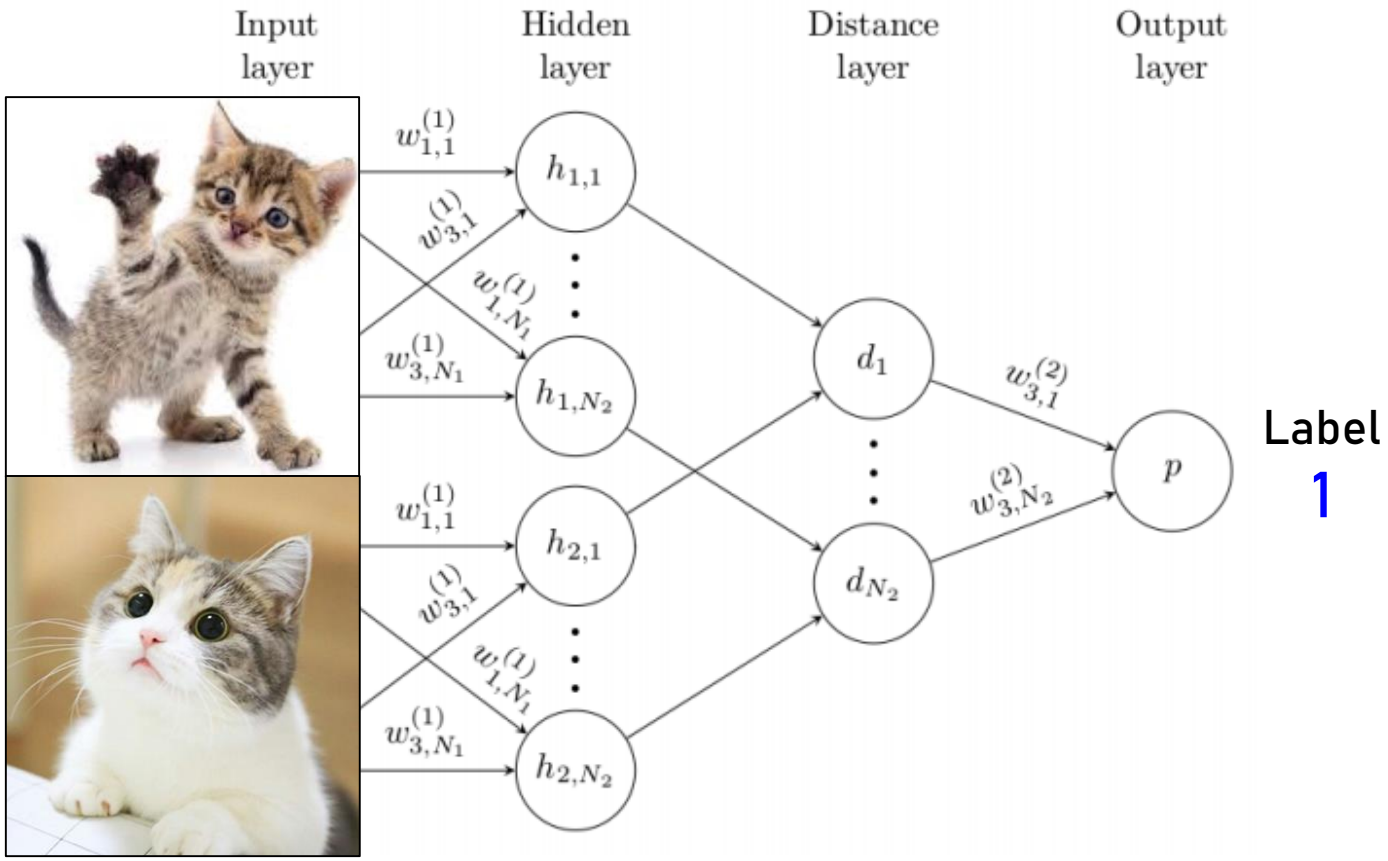
- Siamese Neural Networks
  - A simple example



Koch, Gregory, et al. "Siamese neural networks for one-shot image recognition." *ICML deep learning workshop*, 2015.

Non-parametric approach

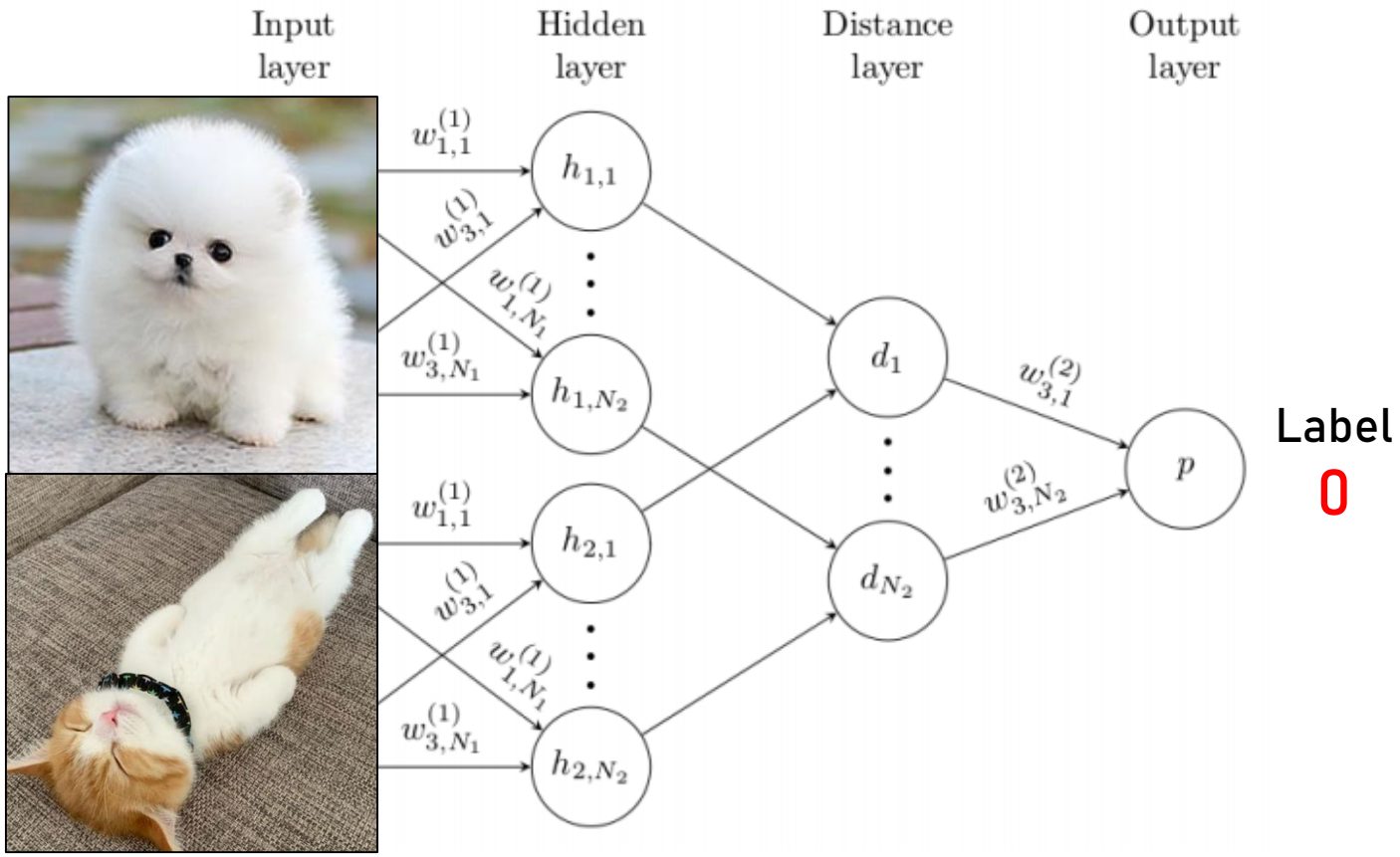
- Siamese Neural Networks
  - A simple example



Koch, Gregory, et al. "Siamese neural networks for one-shot image recognition." *ICML deep learning workshop*, 2015.

Non-parametric approach

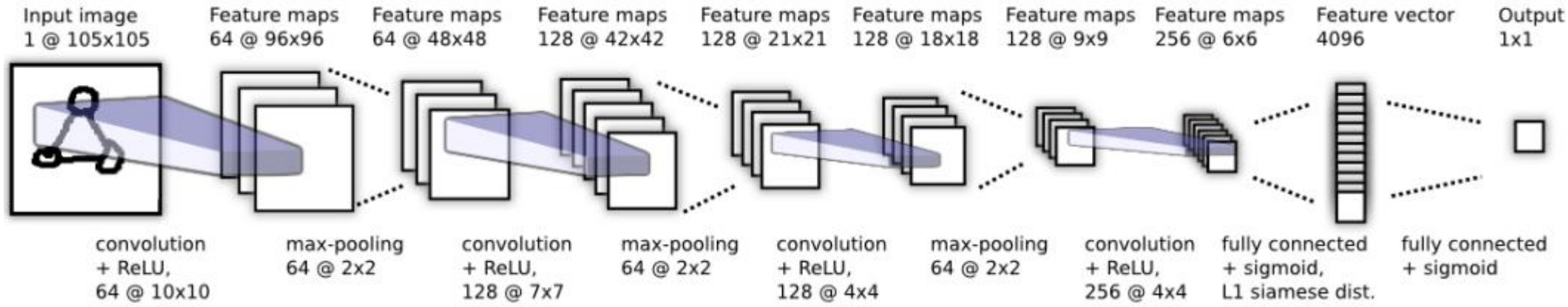
- Siamese Neural Networks
  - A simple example



Koch, Gregory, et al. "Siamese neural networks for one-shot image recognition." *ICML deep learning workshop*, 2015.

Non-parametric approach

- Siamese Neural Networks



Siamese twin is not depicted, but joins immediately after the 4096 unit fully-connected layer where the L1 component-wise distance between vectors is computed

Same class: $y\left(x_1^{(i)}, x_2^{(i)}\right) = 1$

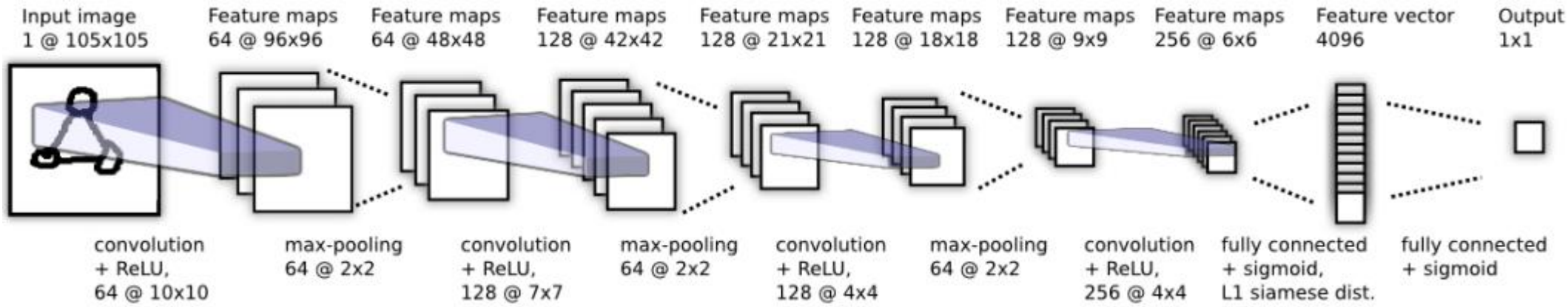
Otherwise: $y\left(x_1^{(i)}, x_2^{(i)}\right) = 0$

$$\mathcal{L}\left(x_1^{(i)}, x_2^{(i)}\right) = \mathbf{y}\left(x_1^{(i)}, x_2^{(i)}\right) \log \mathbf{p}\left(x_1^{(i)}, x_2^{(i)}\right) + \left(1 - \mathbf{y}\left(x_1^{(i)}, x_2^{(i)}\right)\right) \log\left(1 - \mathbf{p}\left(x_1^{(i)}, x_2^{(i)}\right)\right) + \boldsymbol{\lambda}^T |\mathbf{w}|^2$$

Koch, Gregory, et al. "Siamese neural networks for one-shot image recognition." *ICML deep learning workshop*, 2015.

Non-parametric approach

- Siamese Neural Networks



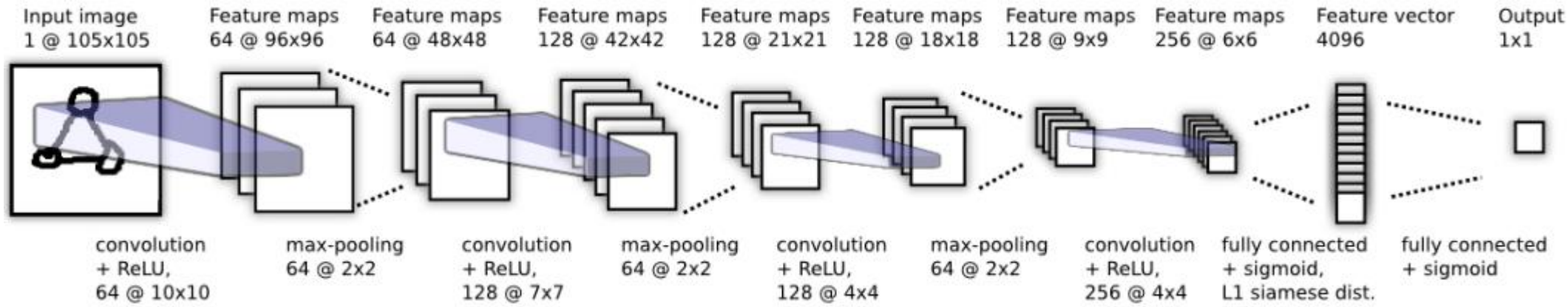
Method	Test
Humans	95.5
Hierarchical Bayesian Program Learning	95.2
Affine model	81.8
Hierarchical Deep	65.2
Deep Boltzmann Machine	62.0
Simple Stroke	35.2
1-Nearest Neighbor	21.7
Siamese Neural Net	58.3
Convolutional Siamese Net	92.0

Test Accuracy on Omniglot Dataset

Koch, Gregory, et al. "Siamese neural networks for one-shot image recognition." *ICML deep learning workshop*, 2015.

Non-parametric approach

- Siamese Neural Networks



Method	Test
Humans	95.5
Hierarchical Bayesian Program Learning	95.2
Affine model	81.8
Hierarchical Deep	65.2
Deep Boltzmann Machine	62.0
Simple Stroke	35.2
1-Nearest Neighbor	21.7
Siamese Neural Net	58.3
Convolutional Siamese Net	92.0

Test Accuracy on Omniglot Dataset

meta-training: 2-way classification

meta-testing: N-way classification

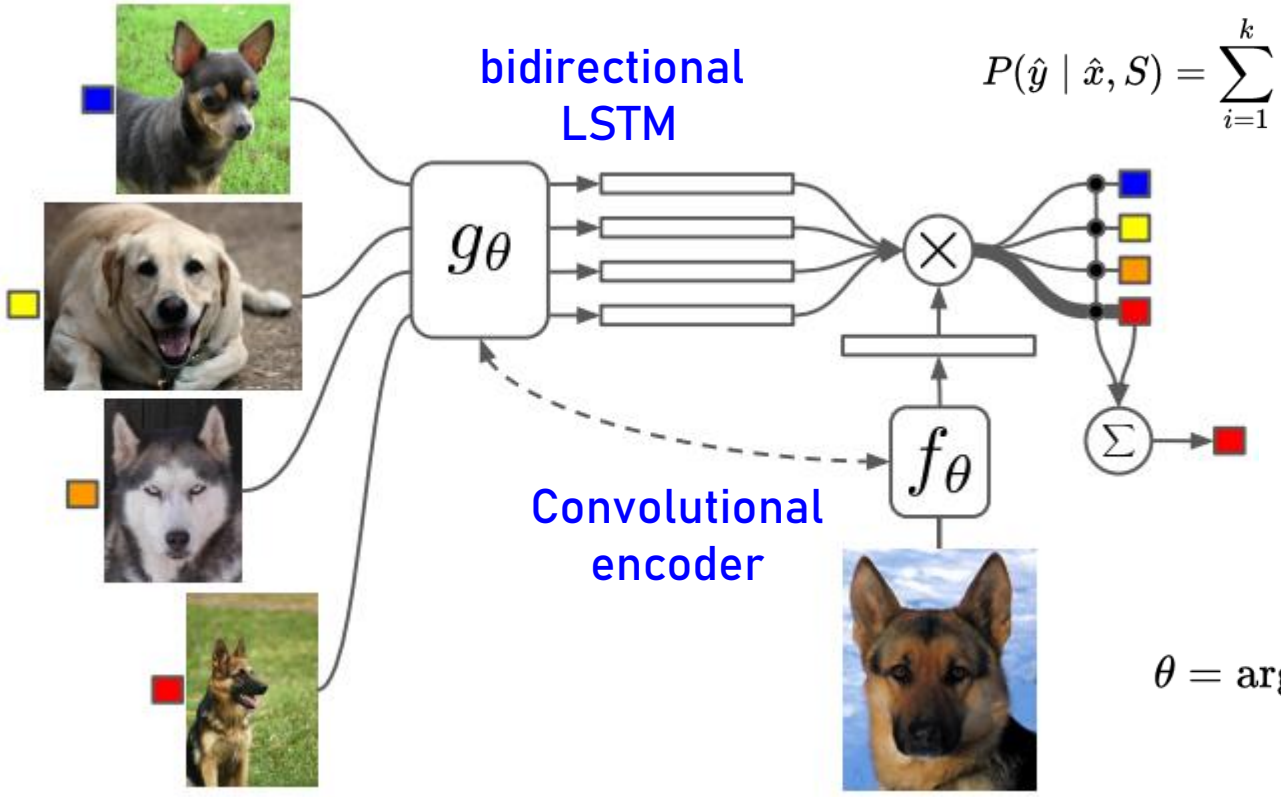
"*A simple machine learning principle: test and train conditions must match*"

Vinyals et al., Matching Networks for One-Shot Learning

Koch, Gregory, et al. "Siamese neural networks for one-shot image recognition." *ICML deep learning workshop*, 2015.

Non-parametric approach

- Matching Networks
  - Can we make a fully end-to-end differentiable nearest neighbor classifier?



$$P(\hat{y} \mid \hat{x}, S) = \sum_{i=1}^{k} a(\hat{x}, x_i) y_i$$

Cosine distance

$$\theta = \arg\max_{\theta} E_{L \sim T}\left[E_{S \sim L, B \sim L}\left[\sum_{(x,y) \in B} \log P_\theta(y \mid x, S)\right]\right]$$

Vinyals, Oriol, et al. "Matching networks for one shot learning." *Advances in neural information processing systems*. 2016.

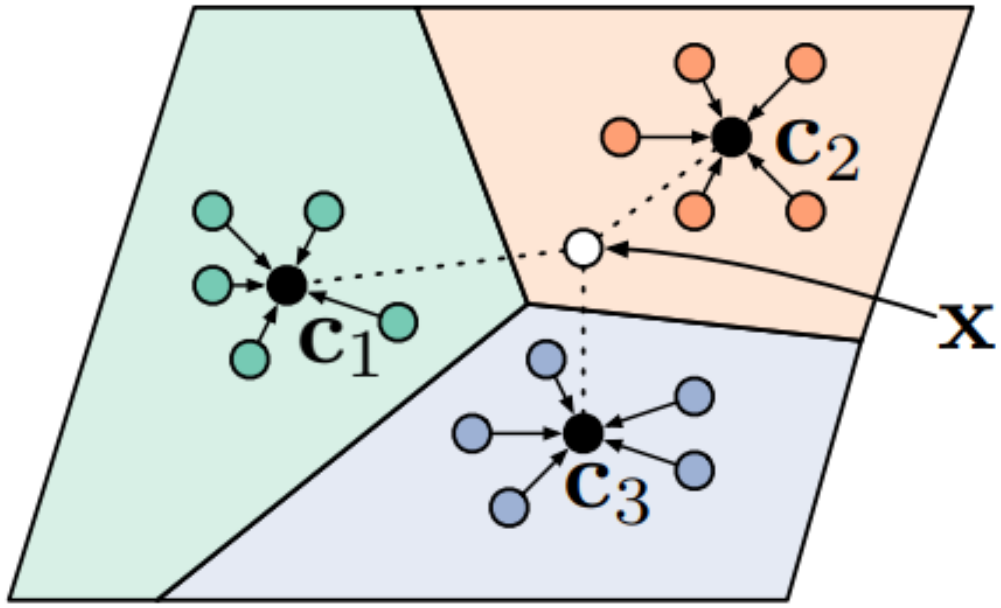
Non-parametric approach

- Prototypical Networks
  - Can we aggregate class information to create a prototypical embedding?

$$\mathbf{c}_k = \frac{1}{|S_k|} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_\phi(\mathbf{x}_i)$$

$$p_\phi(y = k \mid \mathbf{x}) = \frac{\exp(-d(f_\phi(\mathbf{x}), \mathbf{c}_k))}{\sum_{k'} \exp(-d(f_\phi(\mathbf{x}), \mathbf{c}_{k'}))}$$

d: Euclidian distance



Snell, Jake, Kevin Swersky, and Richard Zemel. "Prototypical networks for few-shot learning." *Advances in neural information processing systems*. 2017.

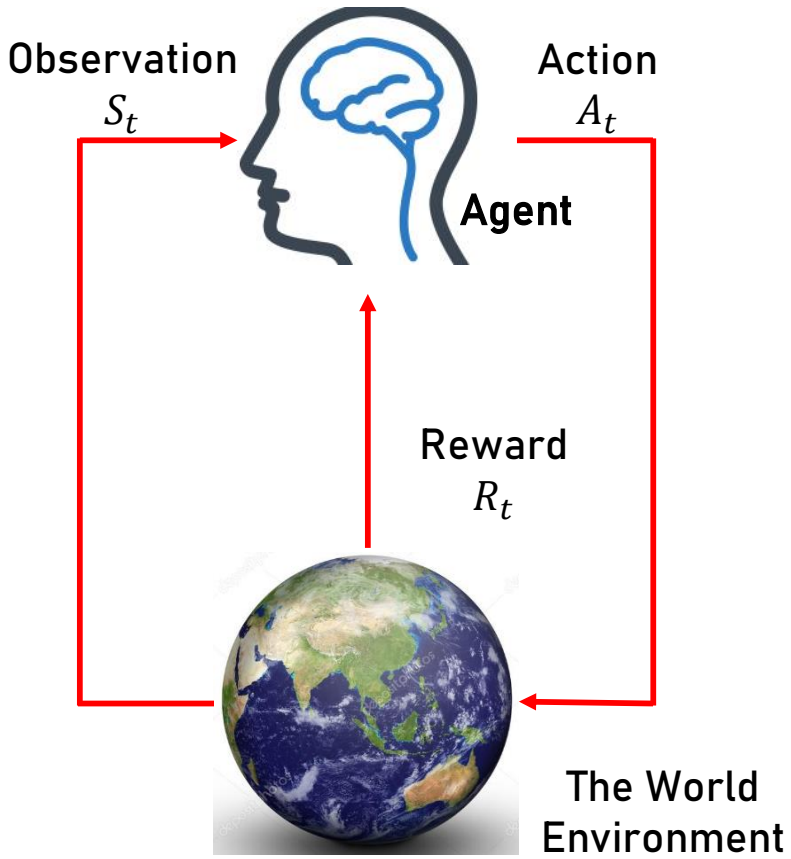
Non-parametric approach: Further Readings

- Koch, Gregory, Richard Zemel, and Ruslan Salakhutdinov. "Siamese neural networks for one-shot image recognition." *ICML deep learning workshop*. Vol. 2. 2015.
- Vinyals, Oriol, et al. "Matching networks for one shot learning." *Advances in neural information processing systems*. 2016.
- Snell, Jake, Kevin Swersky, and Richard Zemel. "Prototypical networks for few-shot learning." *Advances in neural information processing systems*. 2017.
- Sung, Flood, et al. "Learning to compare: Relation network for few-shot learning." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.
- Allen, Kelsey R., et al. "Infinite mixture prototypes for few-shot learning." *arXiv preprint arXiv:1902.04552* (2019).
- Garcia, Victor, and Joan Bruna. "Few-shot learning with graph neural networks." *arXiv preprint arXiv:1711.04043* (2017).

Meta Reinforcement Learning

Reinforcement Learning

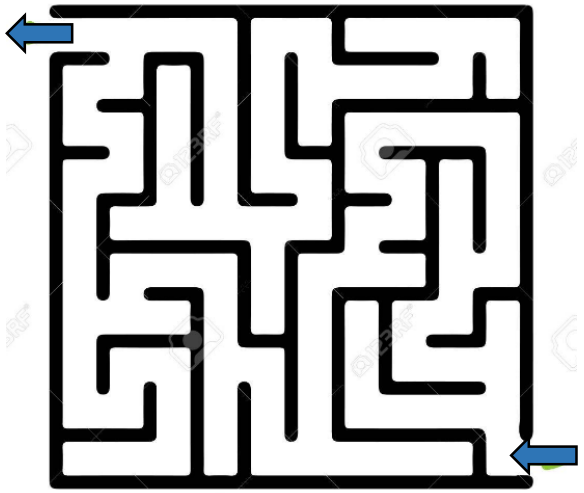
- Reinforcement Learning



- At each time step t the Agent
  - Executes action A_t
  - Receives observation S_t
  - Receives scalar reward R_t
- The environment
  - Receives action A_t
  - Emits observation S_{t+1}
  - Emits scalar reward R_{t+1}
- t increments at env. step

Reinforcement Learning

- Reinforcement Learning

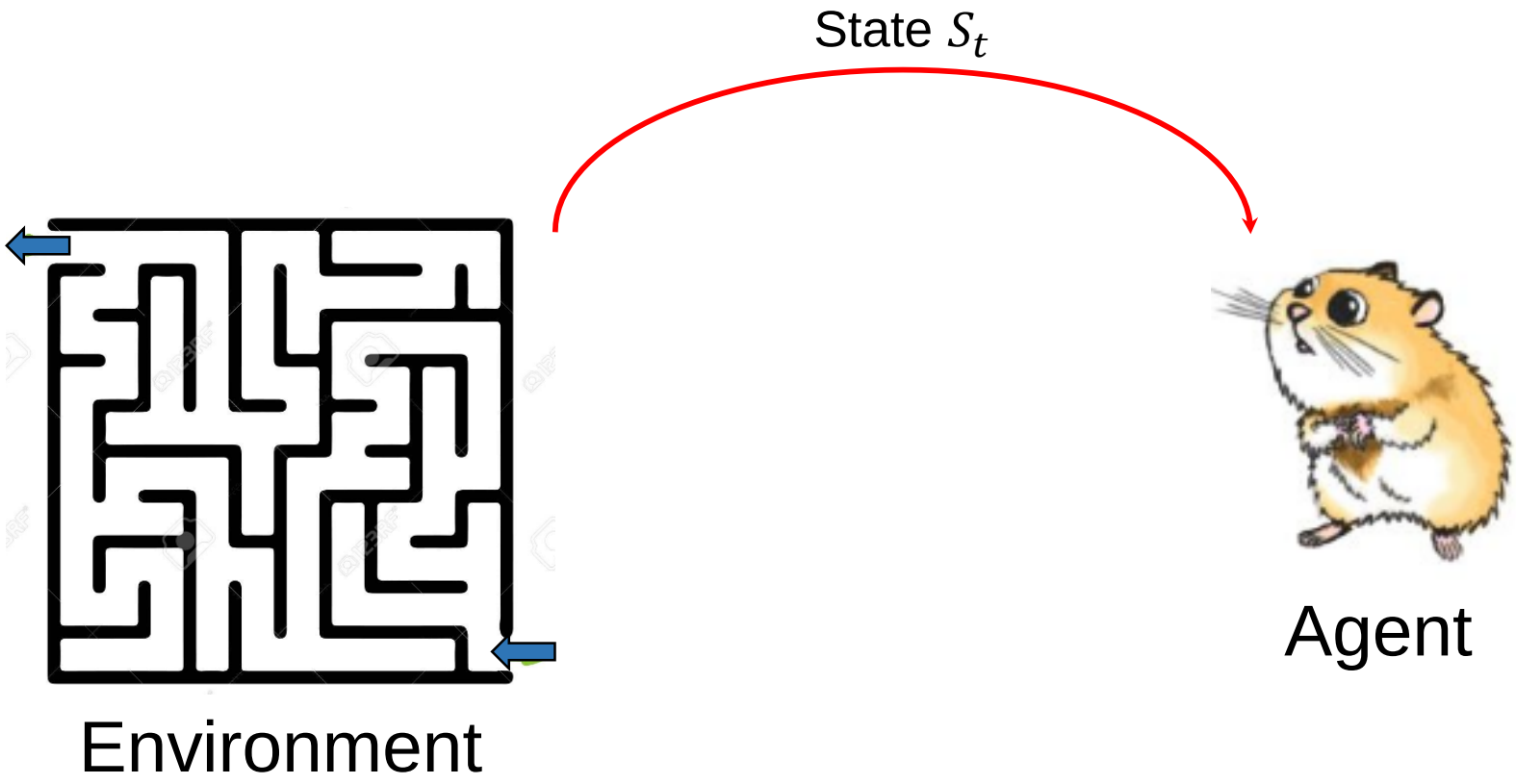


Environment

Agent

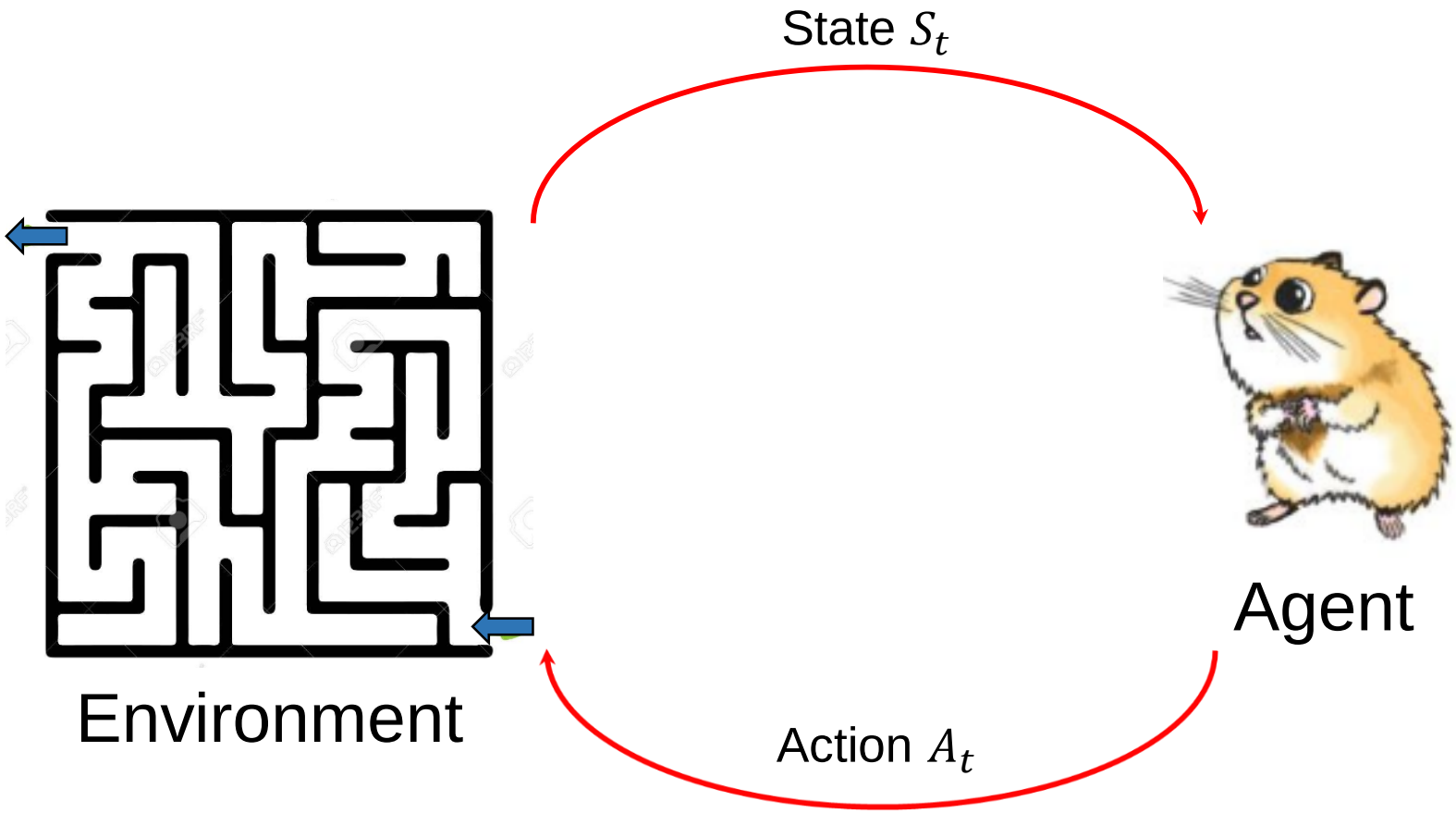
Reinforcement Learning

- Reinforcement Learning



Reinforcement Learning

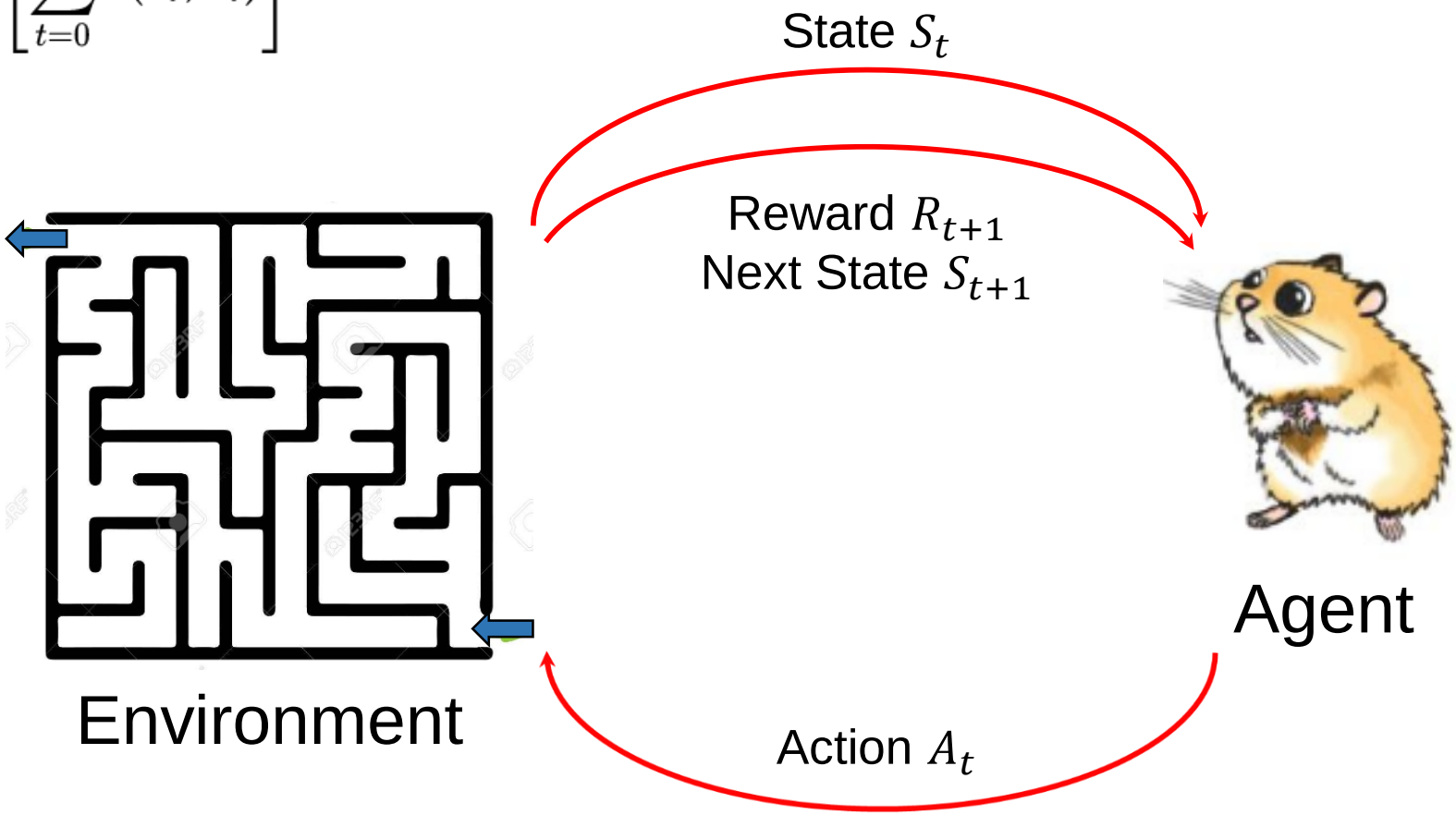
- Reinforcement Learning



Reinforcement Learning

- Reinforcement Learning

$$\theta^{\star} = \arg\max_{\theta} \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T} r(s_t, a_t)\right]$$



Reinforcement Learning

- Markov Decision Processes(MDPs)
  - Mathematical formulation of the RL problem

- MDPs defined by $(\mathcal{S}, \mathcal{A}, \mathbb{P}, \mathcal{R}, \gamma)$
  - $\mathcal{S}$: State set
  - $\mathcal{A}$: Action set
  - $\mathbb{P}$: Transition set
  - $\mathcal{R}$: Reward
  - γ : Discount factor

Reinforcement Learning

- Markov Decision Processes(MDPs)
  - Mathematical formulation of the RL problem

- MDPs defined by $(\mathcal{S}, \mathcal{A}, \mathbb{P}, \mathcal{R}, \gamma)$
  - $\mathcal{S}$: State set
  - $\mathcal{A}$: Action set
  - $\mathbb{P}$: Transition set
  - $\mathcal{R}$: Reward
  - γ : Discount factor

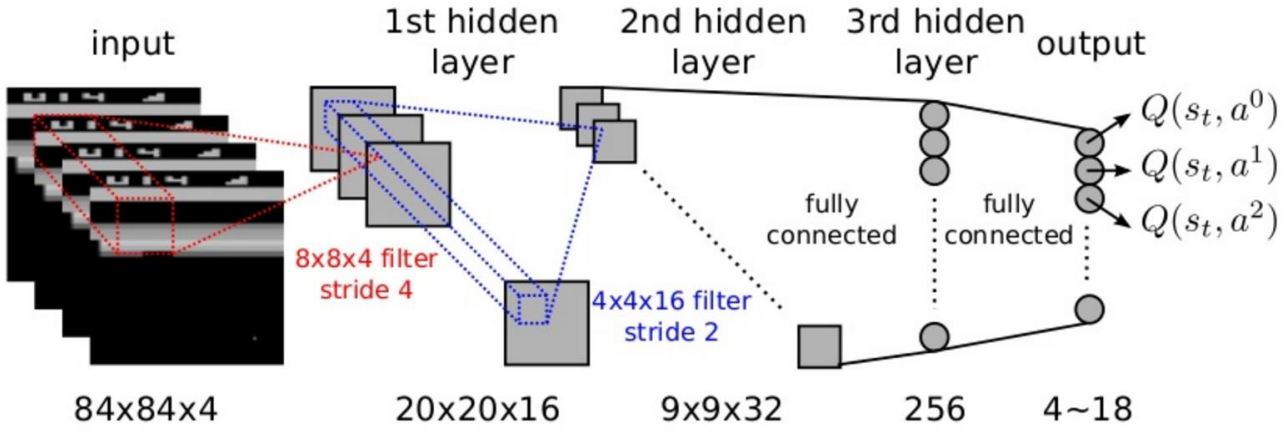
Goal: Find Optimal policy

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \underbrace{\mathbb{E}\left[\sum_{t \geq 0} \gamma^t r_t \mid \pi\right]}_{\text{Value function}} \text{ with } s_0 \sim p(s_0), a_0 \sim \pi(\cdot \mid s_t), s_{t+1} \sim \mathbb{P}(\cdot \mid s_t, a_t)$$

- (Stochastic) Policies
  - Probability of agent' action $\pi(a \mid s) = \mathbb{P}[a_t = a \mid s_t = s]$

Problem Definition

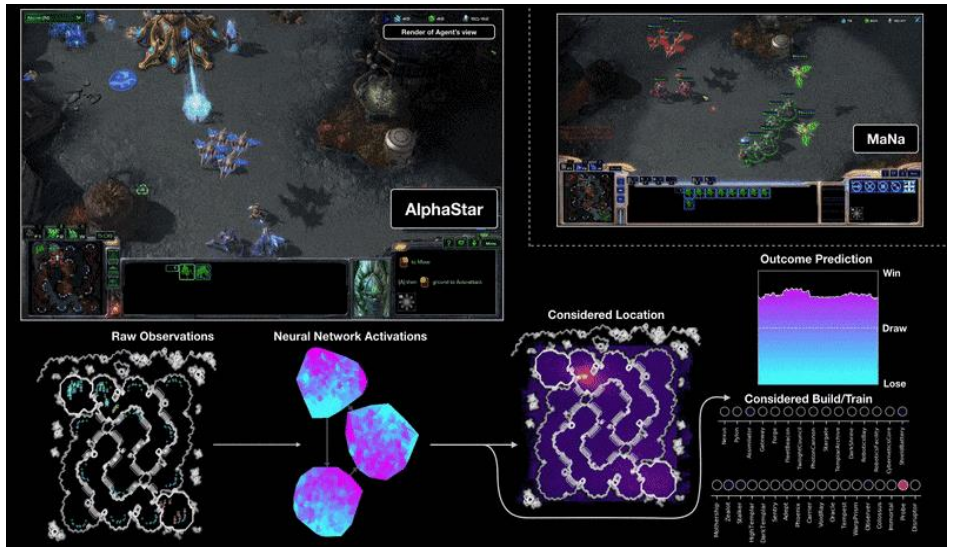
- What's wrong with Reinforcement Learning?



DQN

AlphaGo

OpenAI Five

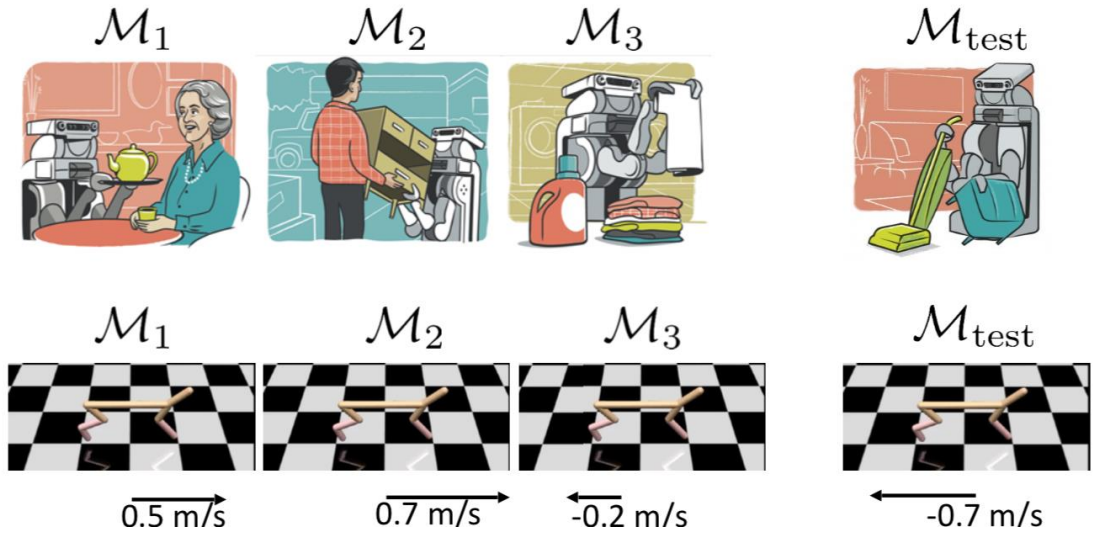


AlphaStar

Problem Definition

- What's wrong with Reinforcement Learning?

MDP 1 MDP 2 MDP 3

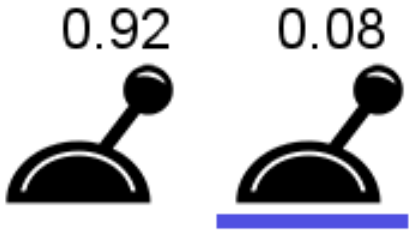


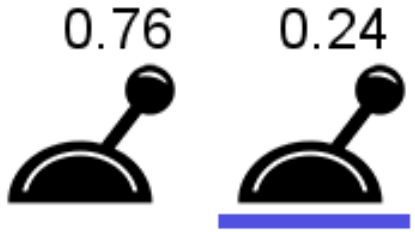
Can we meta-learn reinforcement learning algorithms that are much more efficient?

"Hula Beach", "Never grow up", "The Sled" – by artist Matt Spangler, *mattspangler.com*

Problem Definition

- What's wrong with Reinforcement Learning?





Reinforcement Learning

Meta Reinforcement Learning

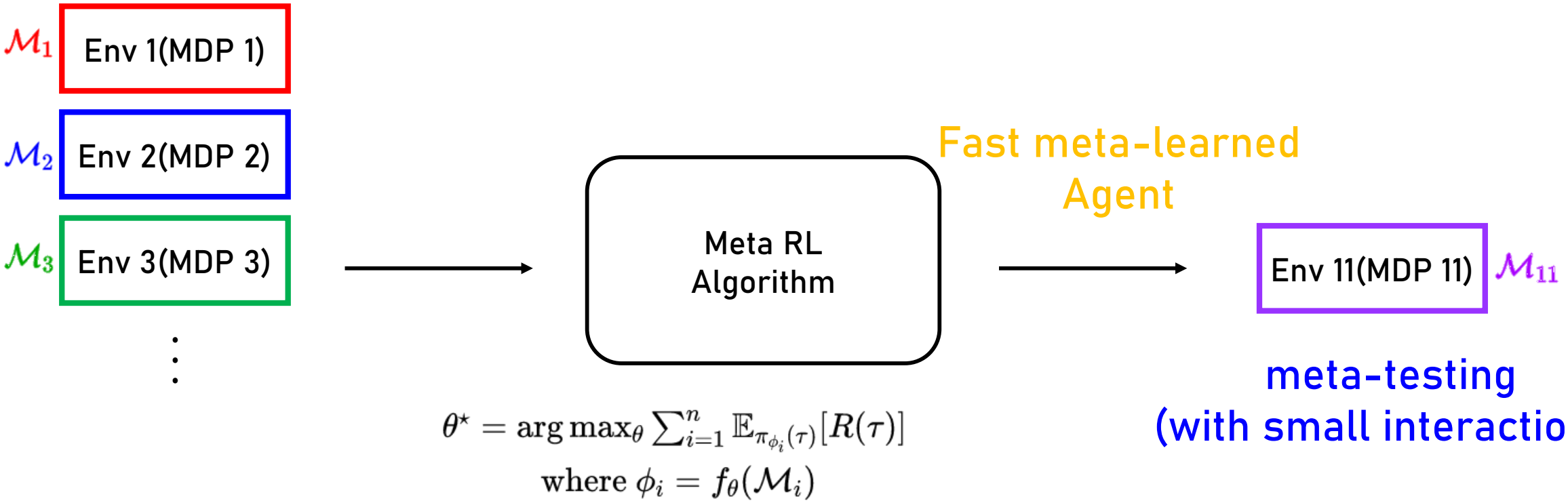
Trial: 1

Trial: 1

https://bit.ly/bandit-meta-rl-ex

Meta Reinforcement Learning

meta-training(10 env)



$$\theta^\star = \arg\max_\theta \sum_{i=1}^{n} \mathbb{E}_{\pi_{\phi_i}(\tau)}[R(\tau)]$$
$$\text{where } \phi_i = f_\theta(\mathcal{M}_i)$$

meta-testing
(with small interactions)

Meta-RL Algorithms

- Recurrent policies appraoch
  - Duan, Yan, et al. "RL^2: Fast reinforcement learning via slow reinforcement learning." *arXiv preprint arXiv:1611.02779* (2016).
  - Mishra, Nikhil, et al. "A simple neural attentive meta-learner." *arXiv preprint arXiv:1707.03141* (2017).

- Optimization-based approach
  - Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." *arXiv preprint arXiv:1703.03400* (2017).
  - Nichol, Alex, Joshua Achiam, and John Schulman. "On first-order meta-learning algorithms." *arXiv preprint arXiv:1803.02999* (2018).

- POMDPs-based approach
  - Rakelly, Kate, et al. "Efficient off-policy meta-reinforcement learning via probabilistic context variables." *International conference on machine learning*. 2019.
  - Humplik, Jan, et al. "Meta reinforcement learning as task inference." *arXiv preprint arXiv:1905.06424* (2019).

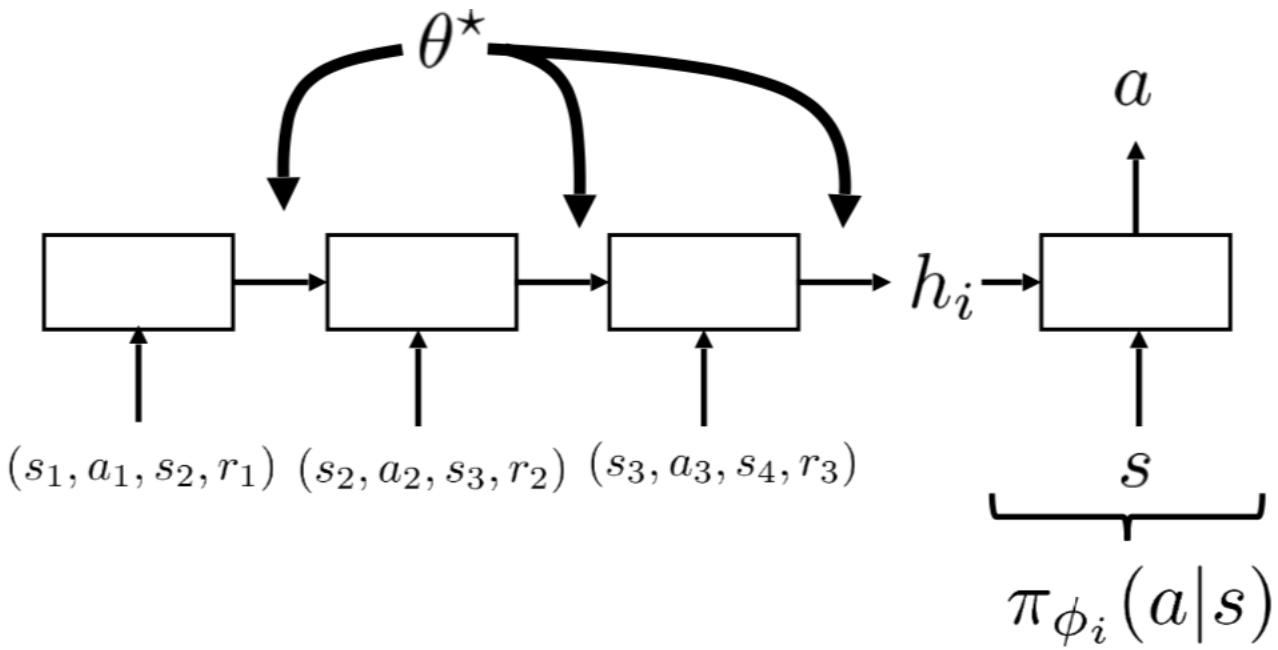
Meta-RL Algorithms

- Recurrent policies appraoch
  - Duan, Yan, et al. "RL^2: Fast reinforcement learning via slow reinforcement learning." *arXiv preprint arXiv:1611.02779* (2016).
  - Mishra, Nikhil, et al. "A simple neural attentive meta-learner." *arXiv preprint arXiv:1707.03141* (2017).
- Optimization-based approach
  - Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." *arXiv preprint arXiv:1703.03400* (2017).
  - Nichol, Alex, Joshua Achiam, and John Schulman. "On first-order meta-learning algorithms." *arXiv preprint arXiv:1803.02999* (2018).
- POMDPs-based approach
  - Rakelly, Kate, et al. "Efficient off-policy meta-reinforcement learning via probabilistic context variables." *International conference on machine learning*. 2019.
  - Humplik, Jan, et al. "Meta reinforcement learning as task inference." *arXiv preprint arXiv:1905.06424* (2019).

Recurrent policies approach

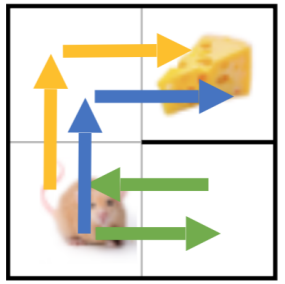
Recurrent policies approach

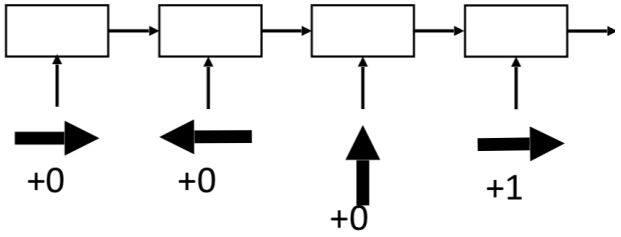
- General approach: Just train a recurrent network!



$$\theta^\star = \arg\max_\theta \sum_{i=1}^{n} \mathbb{E}_{\pi_{\phi_i}(\tau)}[R(\tau)]$$

$$\text{where } \phi_i = f_\theta(\mathcal{M}_i)$$



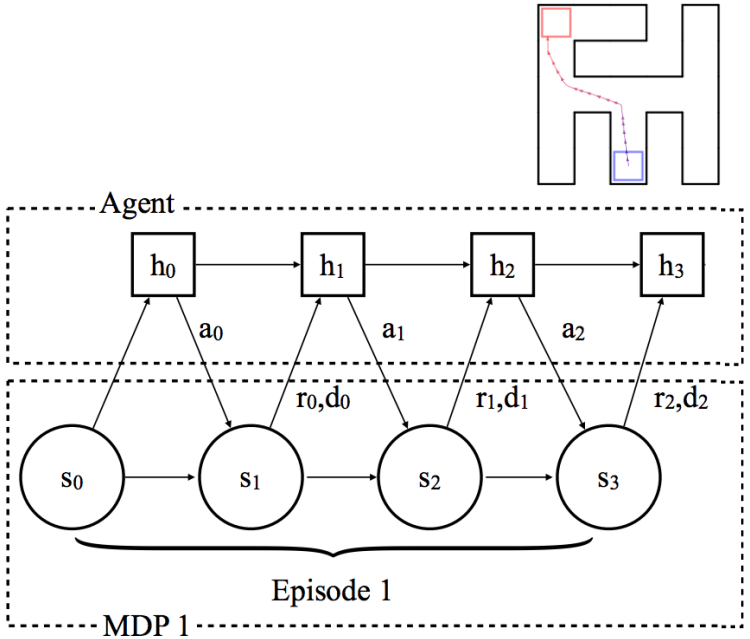


RNN hidden state is not reset between episodes!

Finn, Chelsea, Sergey Levine. “Meta-Learning: from Few-Shot Learning to Rapid Reinforcement Learning." ICML Tutorial 2019

Recurrent policies approach

- RL^2: Fast Reinforcement Learning via Slow Reinforcement Learning



RL Algorithm: TRPO+GAE

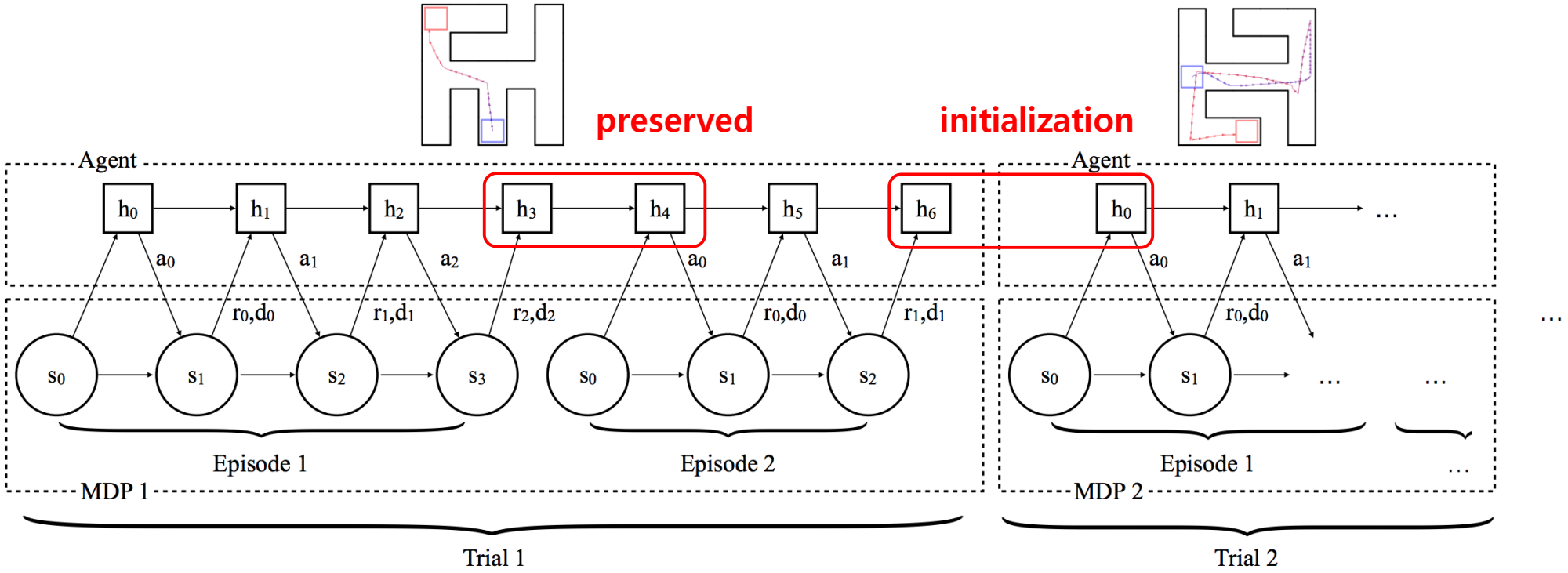
(Vanilla) Reinforcement Learning: Optimize episode by episode

$$\eta(\pi_\theta) = \mathbb{E}_\tau \left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t) \right], \text{ where } \tau = (s_0, a_0, \ldots)$$

Duan, Yan, et al. "RL^2: Fast reinforcement learning via slow reinforcement learning." ICLR 2016.

Recurrent policies approach

- RL^2: Fast Reinforcement Learning via Slow Reinforcement Learning



RL Algorithm: TRPO+GAE

Meta Reinforcement Learning: Optimize trial by trial

$$\max_{\theta} \mathbb{E}_{M} \mathbb{E}_{\tau_M^{(k)}} \left[\sum_{k=1}^{K} R\left(\tau_M^{(k)}\right) \mid \text{RLagent}_{\theta} \right]$$

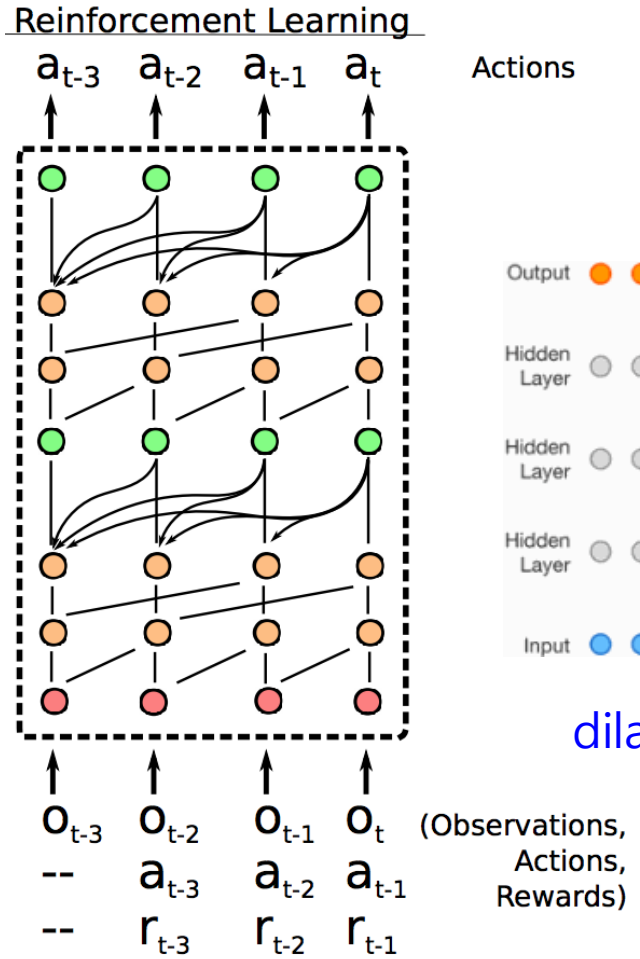
M: Sample environment

$\tau_M^{(k)}$: k'th episode in environment M

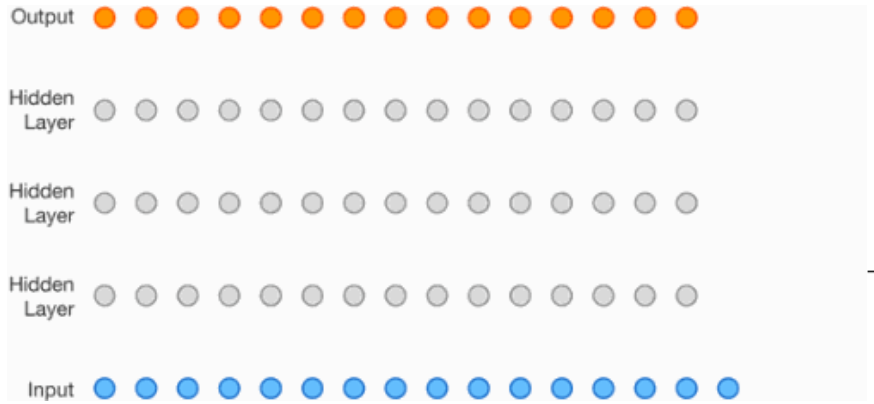
Duan, Yan, et al. "RL^2: Fast reinforcement learning via slow reinforcement learning." ICLR 2016.

Recurrent policies approach

- A Simple Neural AttentIve Meta-Learner(SNAIL)



Like RL^2 but:
replace the LSTM with dilated temporal convolution
(like wavenet) + attention



dilated temporal convolution

function DENSEBLOCK(inputs, dilation rate R, number of filters D):
    xf, xg = CausalConv(inputs, R, D), CausalConv(inputs, R, D)
    activations = tanh(xf) * sigmoid(xg)
    return concat(inputs, activations)

function TCBLOCK(inputs, sequence length T, number of filters D):
    for i in $1, \ldots, \lceil \log_2 T \rceil$ **do**
        inputs = DenseBlock(inputs, 2^i, D)
    return inputs

function ATTENTIONBLOCK(inputs, key size K, value size V):
    keys, query = affine(inputs, K), affine(inputs, K)
    logits = matmul(query, transpose(keys))
    probs = CausallyMaskedSoftmax(logits / $\sqrt{K}$)
    values = affine(inputs, V)
    read = matmul(probs, values)
    return concat(inputs, read)

Mishra, Nikhil, et al. "A simple neural attentive meta-learner." ICLR 2018 & https://www.programmersought.com/article/1715257628/

Recurrent policies approach: Further Readings

- Botvinick, Matthew, et al. "Reinforcement learning, fast and slow." *Trends in cognitive sci ences* 23.5 (2019): 408-422.
- Wang, Jane X., et al. "Learning to reinforcement learn." *arXiv preprint arXiv:1611.05763* (2016).
- Duan, Yan, et al. "RL^2: Fast reinforcement learning via slow reinforcement learning." *arXiv preprint arXiv:1611.02779* (2016).
- Mishra, Nikhil, et al. "A simple neural attentive meta-learner." *arXiv preprint arXiv:1707.03141* (2017).
- Ritter, Samuel, et al. "Been there, done that: Meta-learning with episodic recall." *arXiv preprint arXiv:1805.09692* (2018).

Optimization-based approach

Optimization-based approach

- Model-Agnostic Meta-Learning
  - Also can be applied in Reinforcement Learning!

pre-trained parameters

Fine-tuning [test-time]

$$\phi \leftarrow \boldsymbol{\theta} - \alpha \nabla_\theta \mathcal{L}(\theta, \mathcal{D}^{\text{tr}})$$

training data for new task

Meta-learning

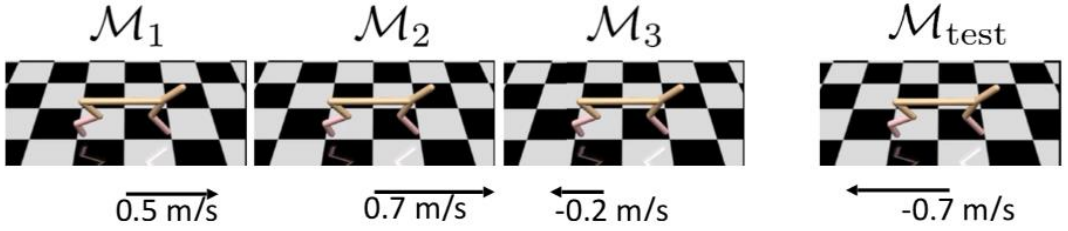
$$\min_\theta \sum_{\text{task } i} \mathcal{L}(\theta - \alpha \nabla_\theta \mathcal{L}(\theta, \mathcal{D}_i^{\text{tr}}), \mathcal{D}_i^{\text{ts}})$$

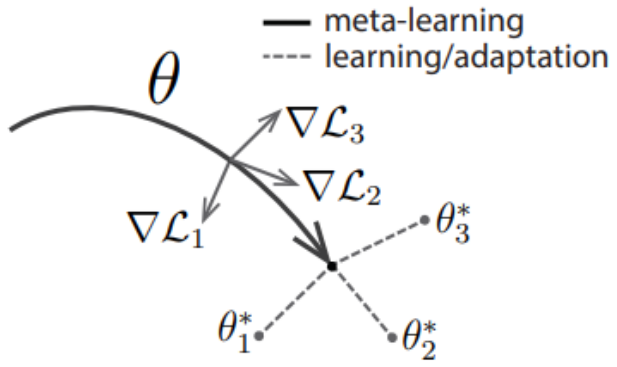
Key idea: Over many tasks, learn parameter vector θ that transfers via fine-tuning

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta-Learning
  - Also can be applied in Reinforcement Learning!





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

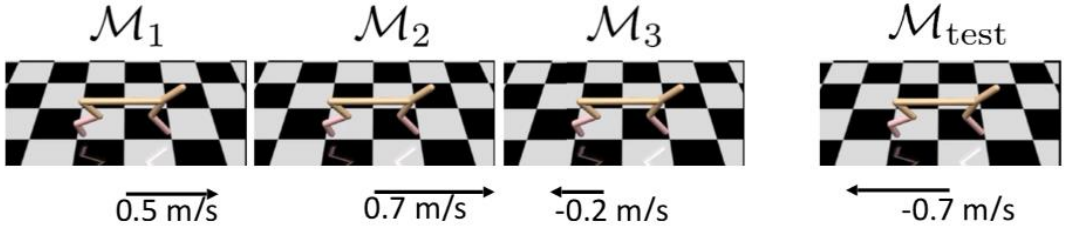
Algorithm 3 MAML for Reinforcement Learning

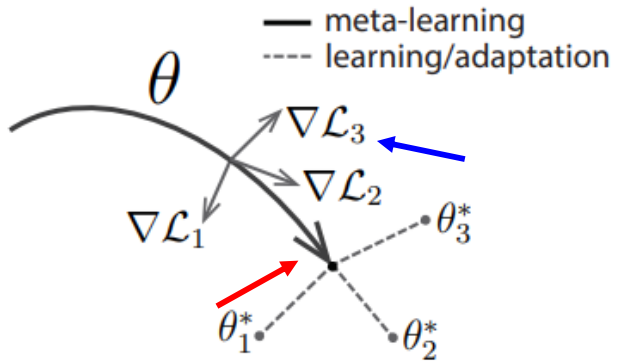
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K trajectories $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{a}_1, ...\mathbf{x}_H)\}$ 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**

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta-Learning
  - Also can be applied in Reinforcement Learning!





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

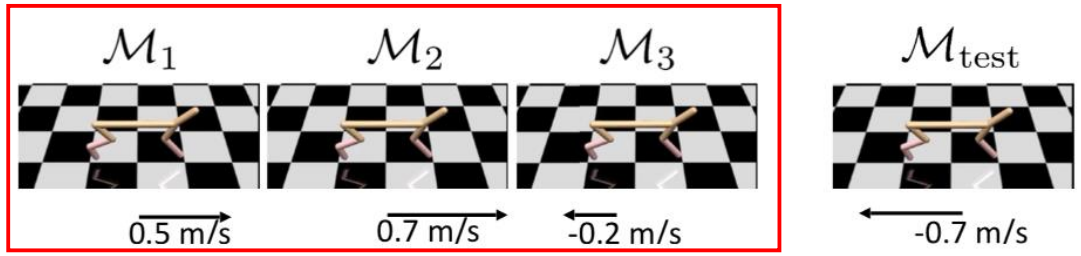
Algorithm 3 MAML for Reinforcement Learning

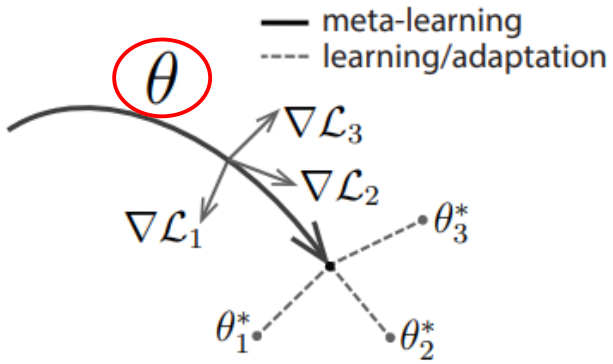
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ — meta-initialization(outer-loop)
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$ — adaptation(inner-loop)
4: **for all** $\mathcal{T}_i$ **do**
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**

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta-Learning
  - Also can be applied in Reinforcement Learning!





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

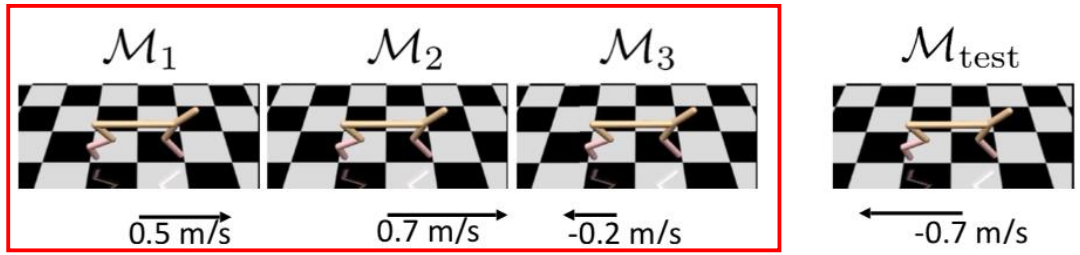
Algorithm 3 MAML for Reinforcement Learning

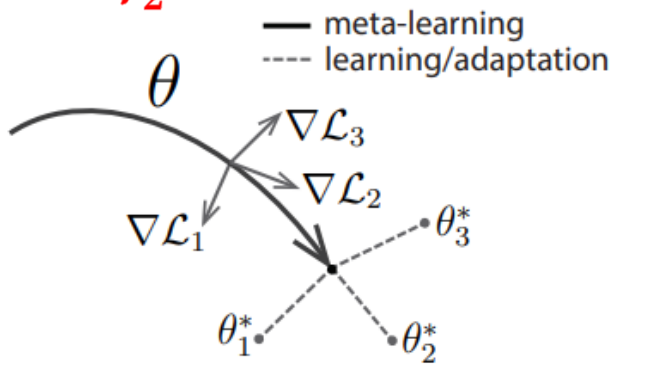
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K trajectories $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{a}_1, ...\mathbf{x}_H)\}$ 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**

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta-Learning
  - Also can be applied in Reinforcement Learning!





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

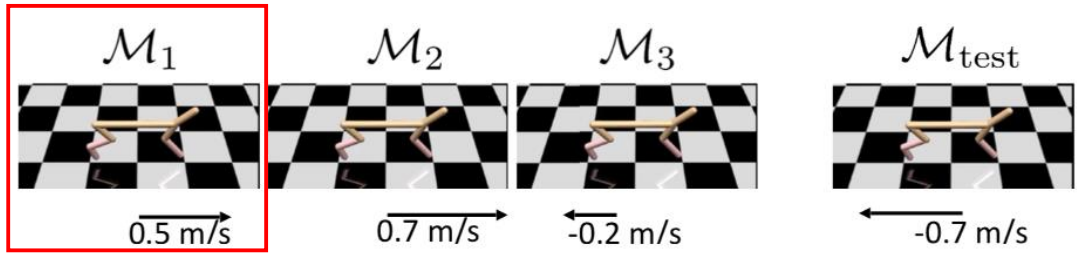
Algorithm 3 MAML for Reinforcement Learning

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**

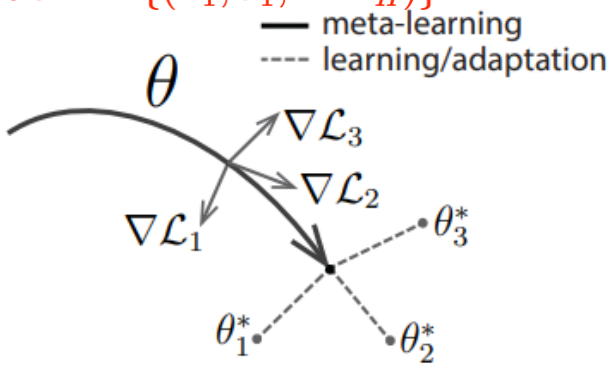
Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta-Learning
  - Also can be applied in Reinforcement Learning!



Roll out $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{a}_1, \ldots \mathbf{x}_H)\}$



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

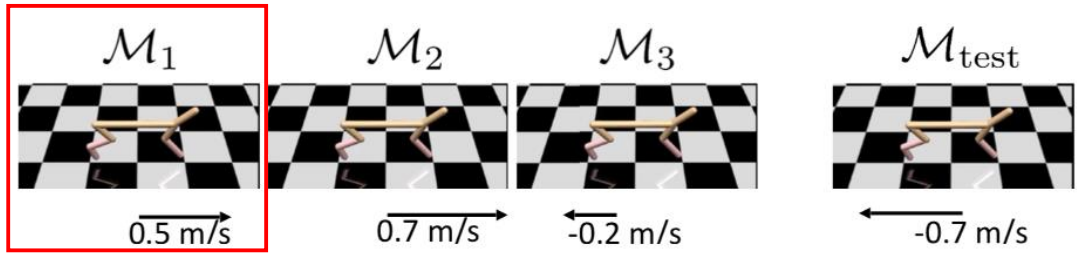
Algorithm 3 MAML for Reinforcement Learning

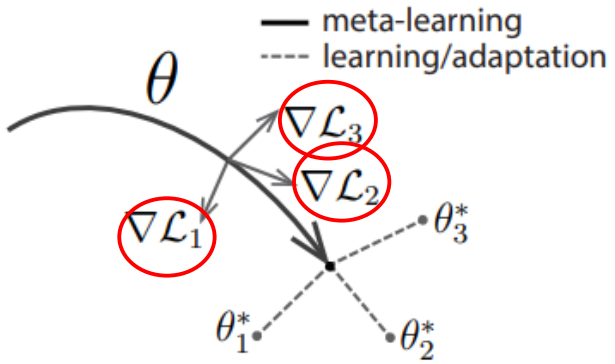
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K trajectories $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{a}_1, \ldots \mathbf{x}_H)\}$ 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 $\mathcal{D}_i' = \{(\mathbf{x}_1, \mathbf{a}_1, \ldots \mathbf{x}_H)\}$ 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**

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta-Learning
  - Also can be applied in Reinforcement Learning!





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

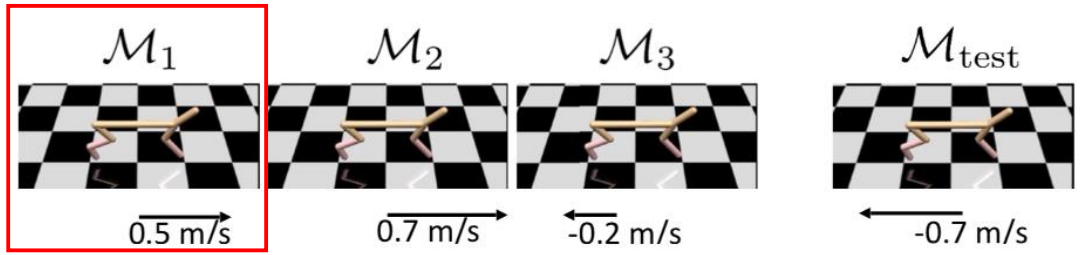
Algorithm 3 MAML for Reinforcement Learning

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**

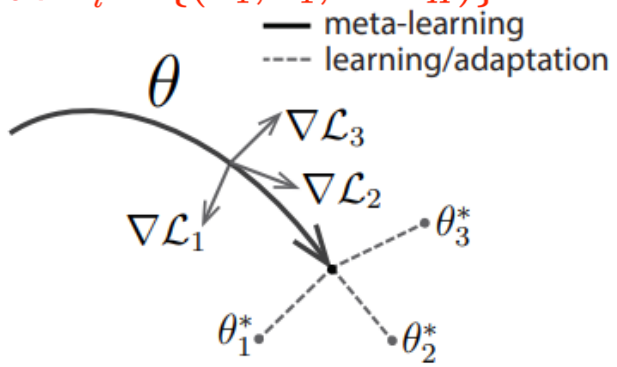
Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta-Learning
  - Also can be applied in Reinforcement Learning!



Roll out $\mathcal{D}_i' = \{(\mathbf{x}_1, \mathbf{a}_1, \ldots \mathbf{x}_H)\}$



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

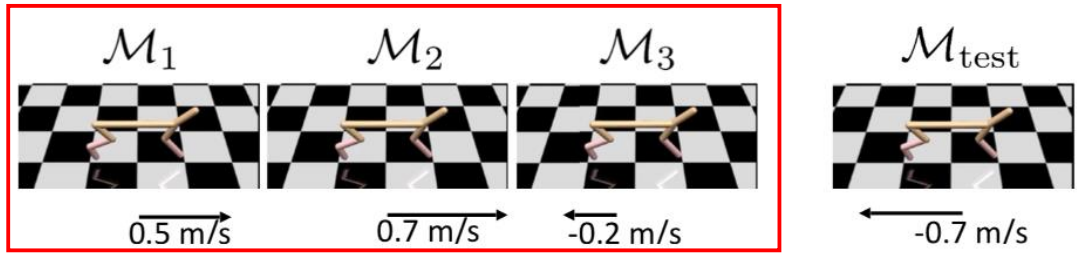
Algorithm 3 MAML for Reinforcement Learning

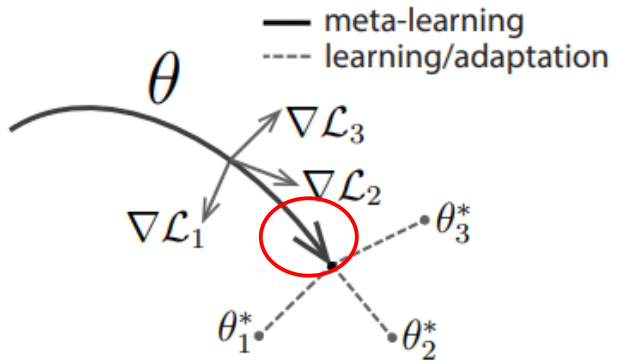
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K trajectories $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{a}_1, \ldots \mathbf{x}_H)\}$ 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 $\mathcal{D}_i' = \{(\mathbf{x}_1, \mathbf{a}_1, \ldots \mathbf{x}_H)\}$ 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**

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta-Learning
  - Also can be applied in Reinforcement Learning!





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

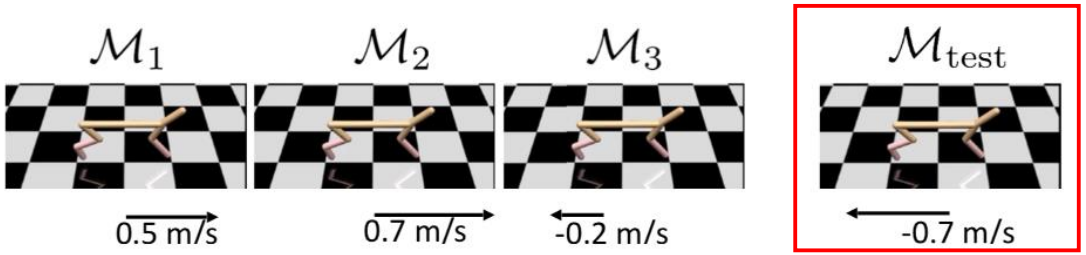
Algorithm 3 MAML for Reinforcement Learning

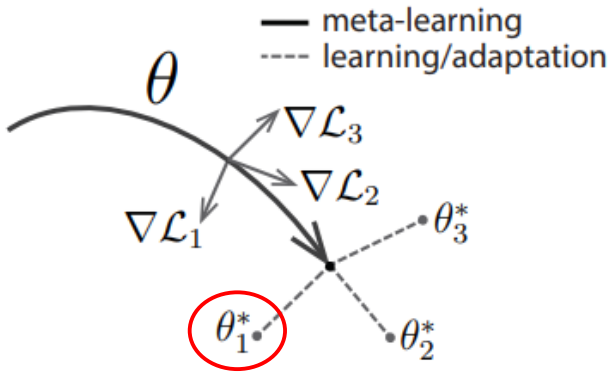
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K trajectories $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{a}_1, ...\mathbf{x}_H)\}$ 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**

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

Optimization-based approach

- Model-Agnostic Meta-Learning
  - Also can be applied in Reinforcement Learning!





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

Fine Tuning(Adaptation)

$$\theta_1^* \leftarrow \boldsymbol{\theta} - \alpha \nabla_\theta \mathcal{L}(\theta, \mathcal{D}_{test})$$

Just 1 or few gradient steps

Evaluation

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." ICML 2017

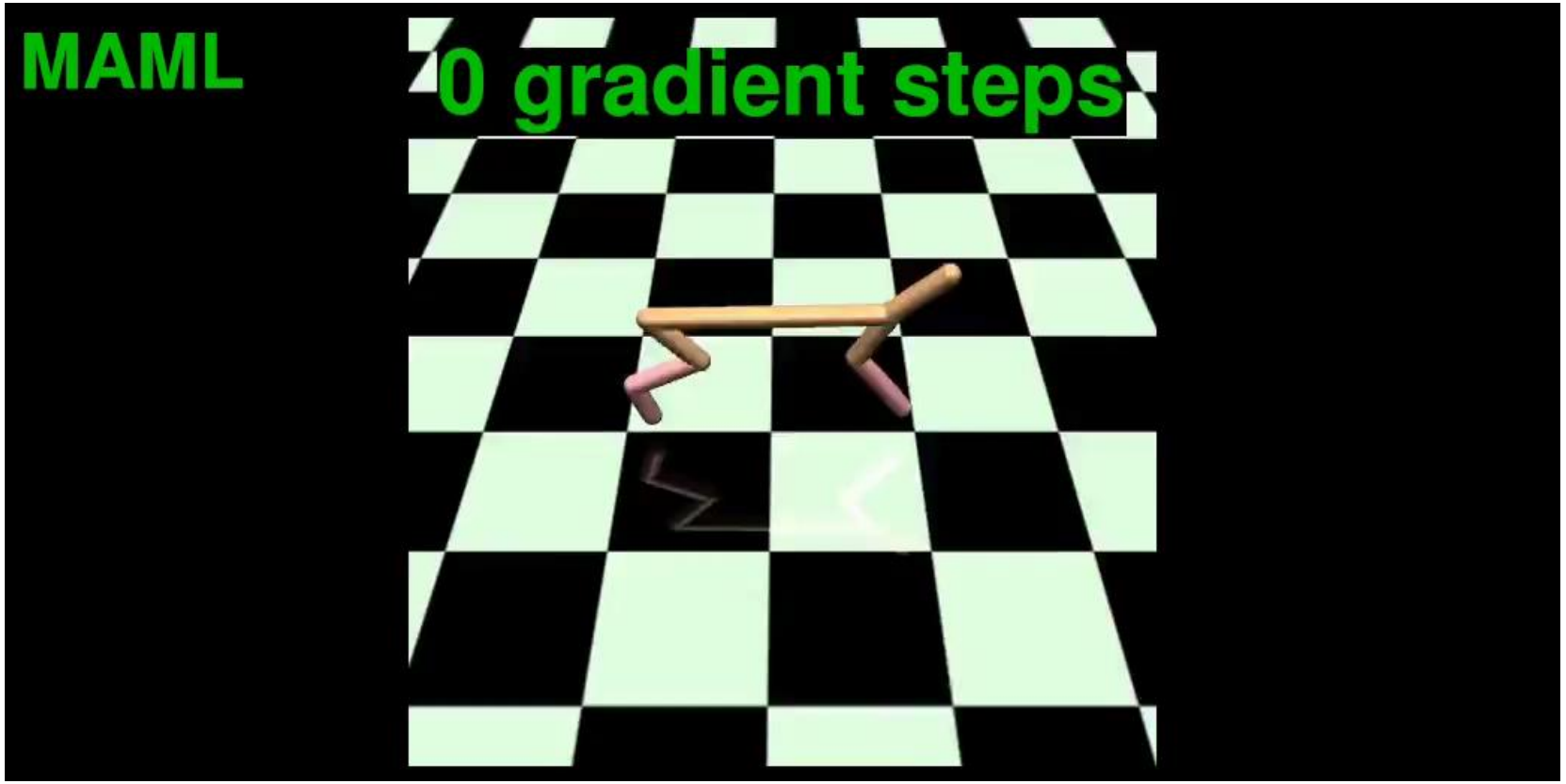
Optimization-based approach

- Model-Agnostic Meta-Learning
  - MAML-RL Demo(meta-test)

https://sites.google.com/view/maml

Optimization-based approach

- Model-Agnostic Meta-Learning
  - MAML-RL Demo(meta-test)



https://sites.google.com/view/maml

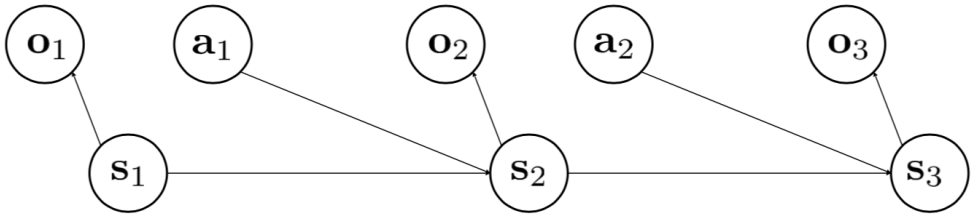
Optimization-based approach: Further Readings

- Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." *arXiv preprint arXiv:1703.03400* (2017).
- Nichol, Alex, Joshua Achiam, and John Schulman. "On first-order meta-learning algorithms." *arXiv preprint arXiv:1803.02999* (2018).
- Xu, Zhongwen, Hado P. van Hasselt, and David Silver. "Meta-gradient reinforcement learning." *Advances in neural information processing systems*. 2018.
- Houthooft, Rein, et al. "Evolved policy gradients." *Advances in Neural Information Processing Systems*. 2018.
- Salimans, Tim, et al. "Evolution strategies as a scalable alternative to reinforcement learning." *arXiv preprint arXiv:1703.03864* (2017).
- Rothfuss, Jonas, et al. "Promp: Proximal meta-policy search." *arXiv preprint arXiv:1810.06784* (2018).
- Fernando, Chrisantha, et al. "Meta-learning by the baldwin effect." *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 2018.

POMDPs-based approach

POMDPs-based approach

- What is the Partially Observable Markov Decision Processes(POMDPs)?
- A POMDP is an MDP with hidden states(hidden Markov model with actions) $(S, A, O, P, R, Z, \gamma)$
- A POMDP is tuple
  - S is a finite set of states
  - A is a finite set of actions
  - O is a finite set of observations
  - P is a state transition probability matrix: $P^a_{ss'} = \mathbb{P}\left[S_{t+1} = s' \mid S_t = s, A_t = a\right]$
  - R is a reward function: $R^a_S = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a]$
  - Z is an observation function: $Z^a_{s'o} = \mathbb{P}\left[O_{t+1} = o \mid S_{t+1} = s', A_t = a\right]$
  - is a discount factor: $\gamma \in [0, 1]$



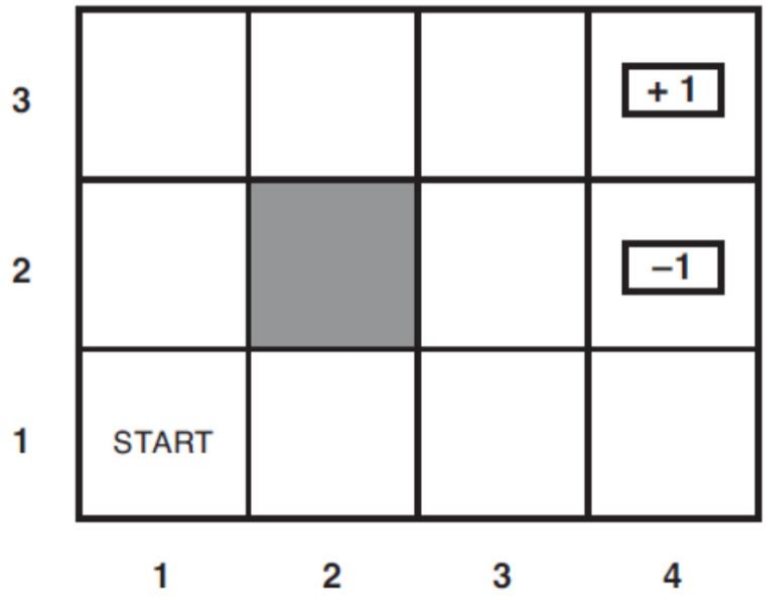
- A History H_t is a sequence of actions, observations, and rewards

$$H_t = A_o, O_1, R_1, \ldots, A_{t-1}, O_t, R_t$$

POMDPs-based approach

- A belief state $b(h)$ is a probability distribution over states, conditioned on history h

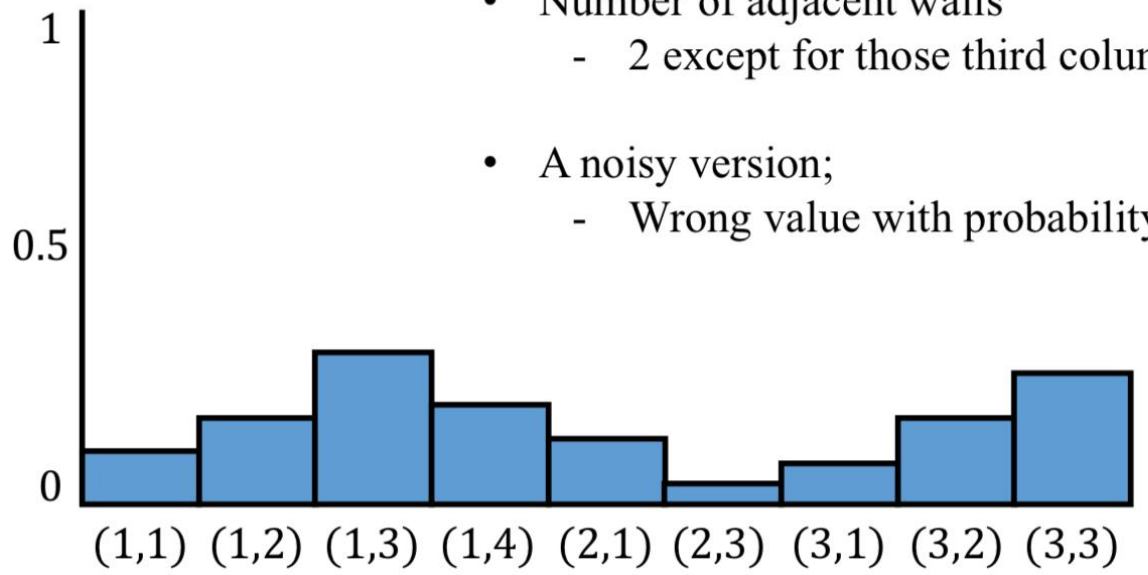
$$b(h) = \left(\mathbb{P}\left[S_t = s^1 \mid H_t = h\right], \ldots, \mathbb{P}\left[S_t = s^n \mid H_t = h\right]\right)$$



Environment

Example of observation;

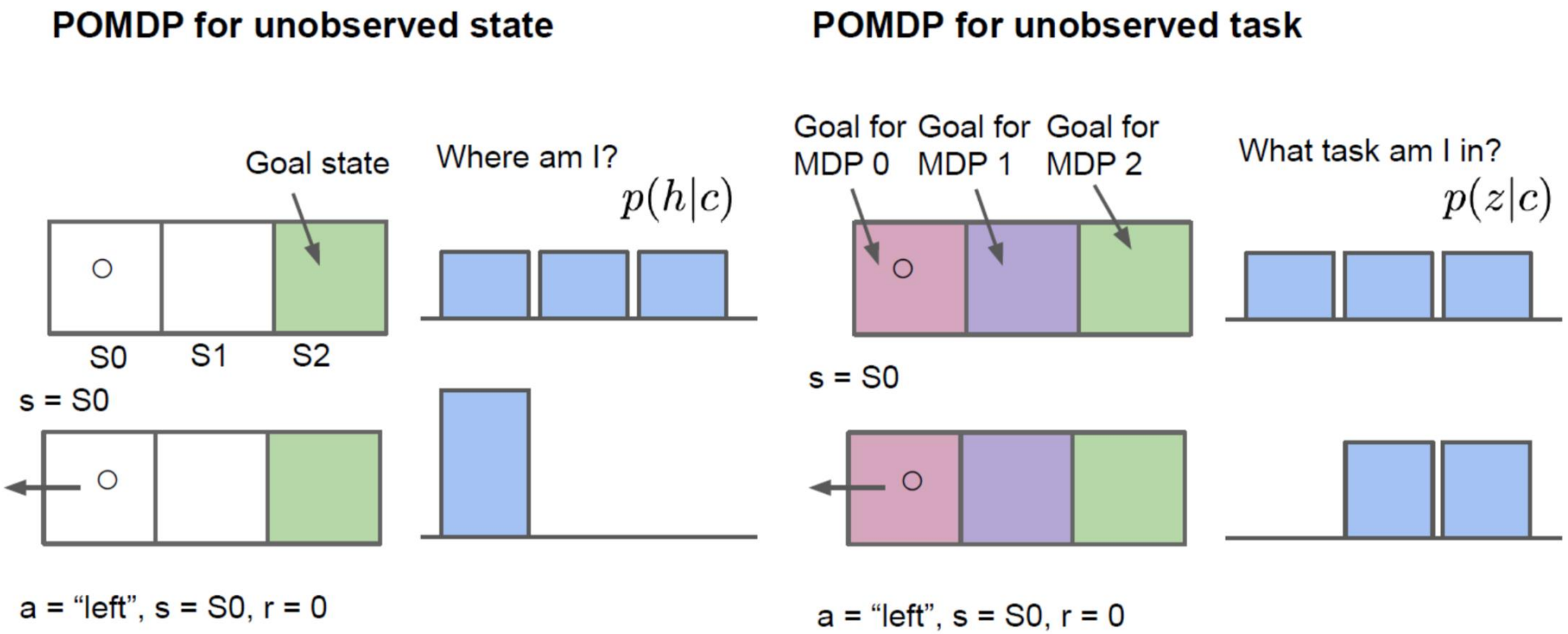
- Number of adjacent walls
  - 2 except for those third column(1)
- A noisy version;
  - Wrong value with probability 0.1



Belief state $b(h)$

POMDPs-based approach

- Meta Reinforcement Learning as POMDPs?
- Model belief over latent task variables



POMDPs-based approach

- Meta Reinforcement Learning as Partially Observable RL
  - Control-Inference duality problem
  - $\pi_\theta(a \mid s, z), \quad z_t \sim p(z_t \mid s_{1:t}, a_{1:t}, r_{1:t})$
  - $Z \rightarrow$ everything needed to solve the task

- So, what does it mean?
  - Learning a task = inferring z
  - Encapsulated information policy $\pi_\theta(a|s,z)$ must solve current task

- General algorithm
  1. Sample $z \sim \hat{p}(z_t \mid s_{1:t}, a_{1:t}, r_{1:t})$ ← some approximate posterior (e.g. variational inference)
  2. Act according to $\pi_\theta(a|s,z)$ to collect more data ← act as though z was correct

POMDPs-based approach

- Meta Reinforcement Learning as Task Inference
  - Optimal agent satisfies,

$$\max_{\pi} \sum_{w} p(w) \sum_{\tau_0 0:\infty} p_\pi(\tau_{0:\infty} \mid w) \left[\sum_{t=0}^{\infty} \gamma^t r_t \right]$$

  - does not have access to task label w
  - but assume to have access to past observed interactions with the task(e.g. LSTM, GRU)

- Belief states
  - The agent itself has no access to w, the posterior satisfies(Appendix A)

$$p(w \mid \tau_{0:t}) \propto p(w) p_0(x_0 \mid w) \prod_{t'=0}^{t-1} P(x_{t'+1} \mid x_{t'}, a_{t'}, w) R(r_{t'} \mid x_{t'}, a_{t'}, x_{t'+1}), w)$$

$$p(w \mid \tau_{0:t}) \propto p(w) p_0(x_0 \mid w) \prod_{t'=0}^{t-1} P(r_{t'}, x_{t'+1} \mid x_{t'}, a_{t'}, w)$$

Can use off-policy algorithm!

Humplik, Jan, et al. "Meta reinforcement learning as task inference." *arXiv* 2019

POMDPs-based approach

- An agent consists of two modules;
  - The optimal meta-learner only needs to make decisions based on current state and the current belief $b_t(w)$

$$\pi(a_t|\tau_{0:t}) \equiv \pi(a_t|x_t, b_t)$$

1. $\pi(a_t|s_t, \hat{b}_t)$: the policy dependent on the current state and (an approximate representation of) posterior

2. The belief module: learns to output an approximate representation $\hat{b}_t$

But how can we estimate the $\hat{b}_t$?

Humplik, Jan, et al. "Meta reinforcement learning as task inference." *arXiv* 2019

POMDPs-based approach

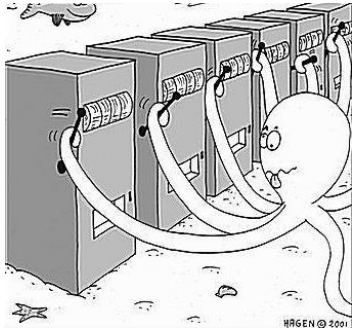
- Different type of task description for Auxiliary supervised learning
  1. Task description
     - Train belief module $b_\theta(h_t|\tau_{0:t})$ to directly task description(e.g. goal position): $h_t = w$
  2. Task embeddings
     - We can define k task and indexing these by $\{1,2,\dots,k\}$ (one-hot or pre-trained), so we can train belief module to predict index of i^w task w: $h_t = i^w$

- Train a belief module as supervised learning
  - Train belief module $b_\theta(h_t|\tau_{0:t})$ to predict task information in a supervised way to minimizing $\mathbb{E}_{p(h_t|\tau_{0:t})}[-\log b_\theta(h_t \mid \tau_{0:t})]$
  - So, the target distribution is $p(h_t \mid \tau_{0:t})$, we can minimize $\mathbb{KL}(p(h_t \mid \tau_{0:t}) \| b_\theta(h_t \mid \tau_{0:t}))$

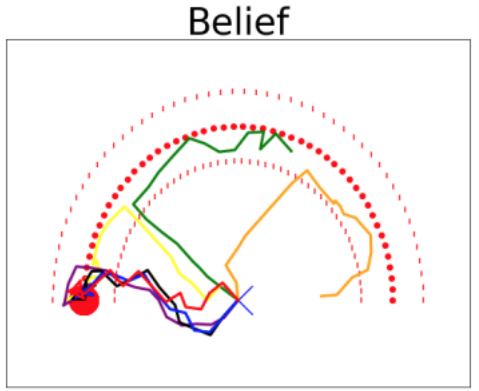
Humplik, Jan, et al. "Meta reinforcement learning as task inference." *arXiv* 2019

POMDPs-based approach

- Some examples of task description(experiments)
  1. Multi-armed bandit
     - A vector of arm probabilities

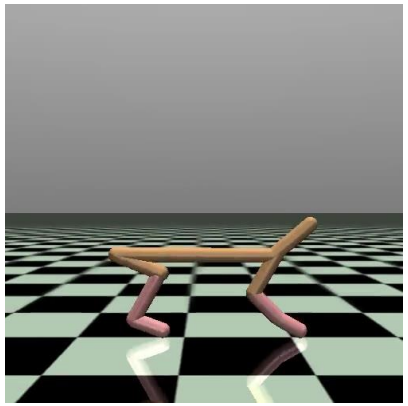


  2. Semi-circle
     - An angle of semi-circle(e.g. $\max = 2\pi$)



  3. Cheetah velocity
     - v_{target}

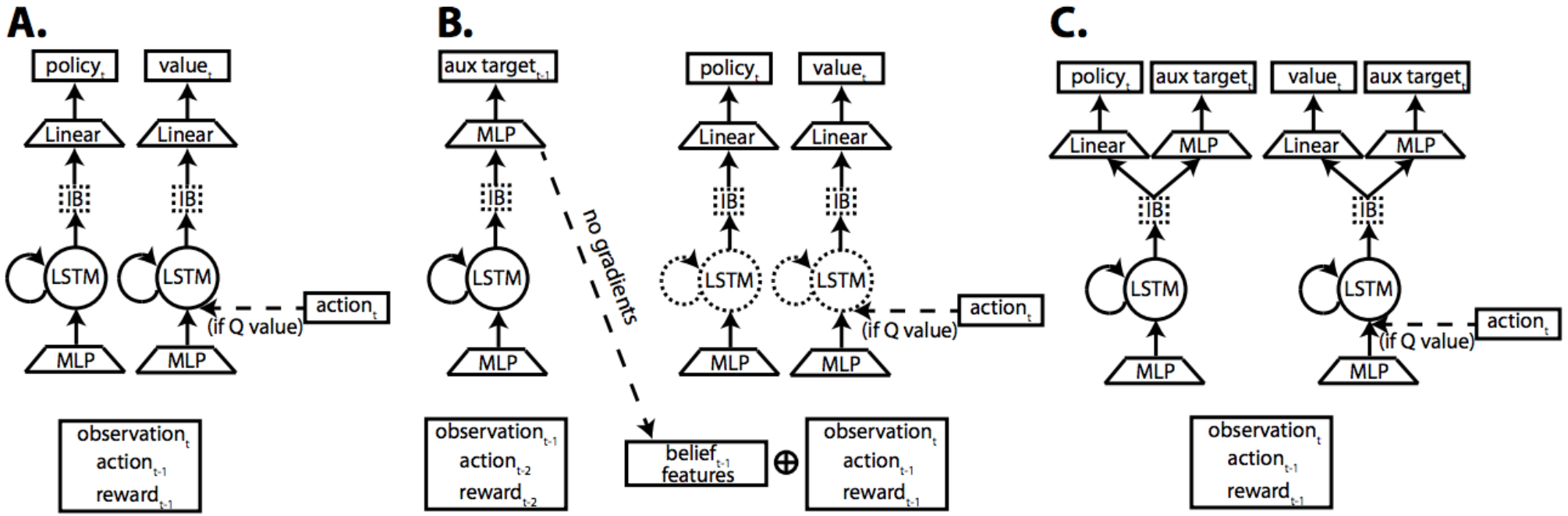
$$r(v) = \max\left(1 - \left|\frac{v}{v_{\text{target}}} - 1\right|, 0\right)$$



Humplik, Jan, et al. "Meta reinforcement learning as task inference." *arXiv* 2019

POMDPs-based approach

- Meta Reinforcement Learning as Task Inference



A. A baseline LSTM agent
B. A belief network agent
C. Auxiliary head agent

- LSTMs and information bottleneck(IB) is optional
- RL algorithms
  - SVG(0)(Nicolas, Heess, et al., 2015) as off-policy
  - PPO(John, Schulman, et al., 2017) as on-policy

Humplik, Jan, et al. "Meta reinforcement learning as task inference." *arXiv* 2019

POMDPs-based approach: Further Readings

- Osband, Ian, Daniel Russo, and Benjamin Van Roy. "(More) efficient reinforcement learning via posterior sampling." *Advances in Neural Information Processing Systems*. 2013.
- Rakelly, Kate, et al. "Efficient off-policy meta-reinforcement learning via probabilistic context variables." *International conference on machine learning*. 2019.
- Humplik, Jan, et al. "Meta reinforcement learning as task inference." *arXiv preprint arXiv:1905.06424* (2019).
- Zintgraf, Luisa, et al. "Variational Task Embeddings for Fast Adaption in Deep Reinforcement Learning." *International Conference on Learning Representations Workshop on Structure & Priors in Reinforcement Learning*. 2019.
- Zintgraf, Luisa, et al. "VariBAD: A Very Good Method for Bayes-Adaptive Deep RL via Meta-Learning." *arXiv preprint arXiv:1910.08348* (2019).

Thank you for your attention!