DSC190: Machine Learning with Few Labels

Meta learning

Zhiting Hu

Lecture 16, November 18, 2021

UC San Diego
HALICIOĞLU DATA SCIENCE INSTITUTE

Outline

- Meta learning (65mins)

- 2 Paper presentations (15 mins)
  - **Harsha Jagarlamudi:** ML-MG: Multi-label Learning with Missing Labels Using a Mixed Graph

Recap: Problem Settings

Multi-Task Learning

Solve multiple tasks $\mathcal{T}_1, \cdots, \mathcal{T}_T$ at once.

$$\min_{\theta} \sum_{i=1}^{T} \mathcal{L}_i(\theta, \mathcal{D}_i)$$

Transfer Learning

Solve target task $\mathcal{T}_b$ after solving source task $\mathcal{T}_a$ by *transferring* knowledge learned from $\mathcal{T}_a$

The Meta-Learning Problem

Given data from $\mathcal{T}_1, \ldots, \mathcal{T}_n$, quickly solve new task $\mathcal{T}_{\text{test}}$

In transfer learning and meta-learning:
generally impractical to access prior tasks

In all settings: tasks must share structure.

Slide courtesy: Finn, Stanford CS 330, Fall 2021

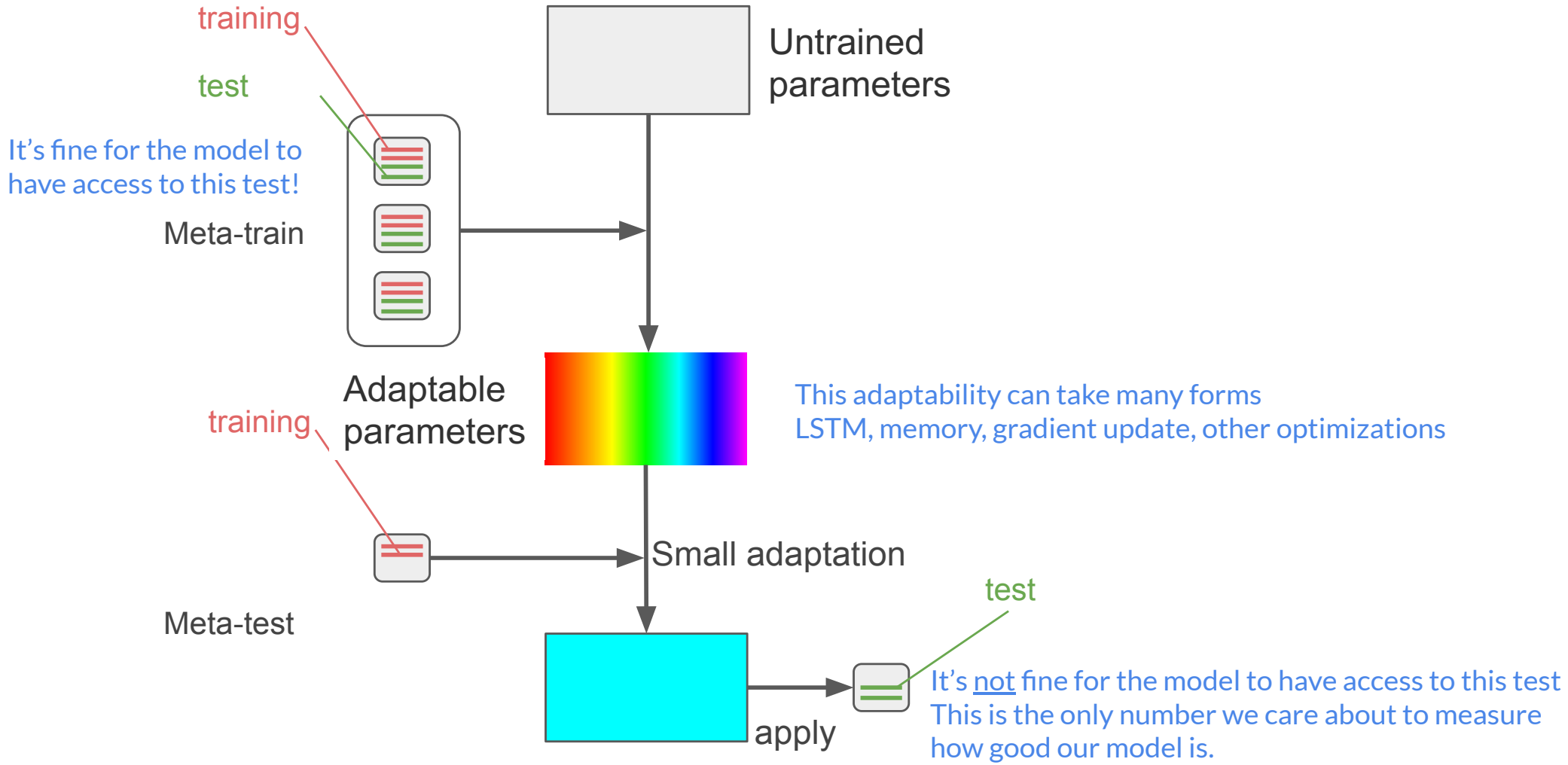
Recap: Meta-learning for few-shot learning

Given 1 example of 5 classes:

Classify new examples

Slide courtesy: Finn, Stanford CS 330, Fall 2021

Recap: Meta-Learning Methods



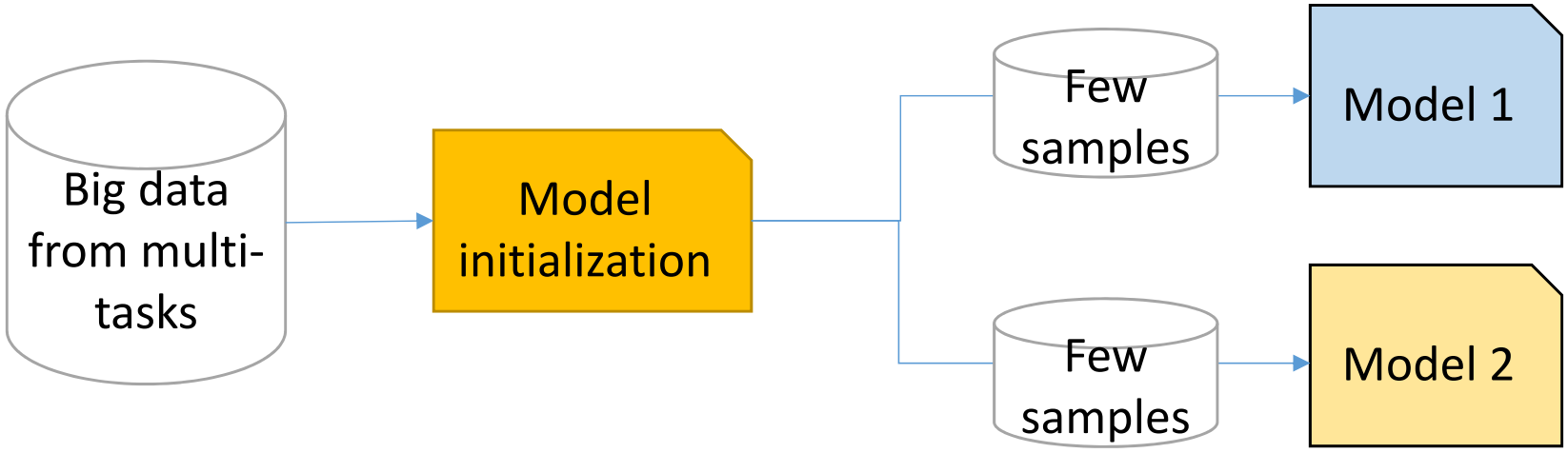
Slide courtesy: Andreas, MIT 6.884

Meta-Learning Methods

- Initialization based methods
  - Learning how to initialize the model for the new task
- Black-box adaptation methods
- Non-parametric methods

Recap: Model-Agnostic Meta Learning (MAML)

- Goal: train a model that can be fast adapted to different tasks via few shots
- MAML idea: directly optimize for an initial representation that can be effectively fine-tuned from a small number of examples



Finn et al. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. ICML 2017.

Slide courtesy: Zhang, SJTU CS420, 2019

Recap: Fine-tuning

Fine-tuning

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

pre-trained parameters

training data for new task

Finn et al. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. ICML 2017.

Slide courtesy: Finn, Stanford CS 330, Fall 2021

Recap: Model-Agnostic Meta Learning (MAML)

Fine-tuning
[test-time]

pre-trained parameters

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

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 et al. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. ICML 2017.

Slide courtesy: Finn, Stanford CS 330, Fall 2021

Algorithm 2 MAML for Few-Shot Supervised Learning

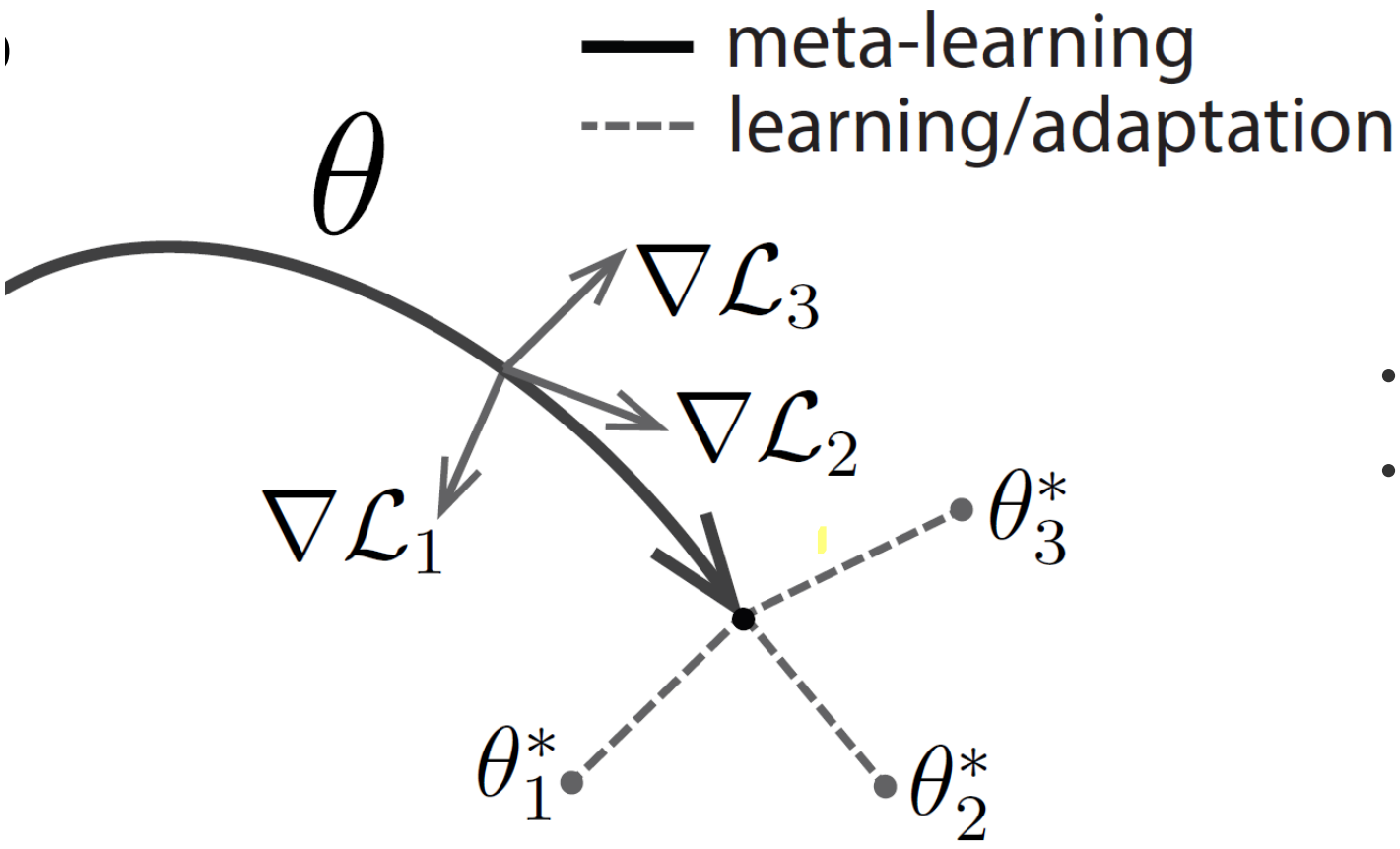
Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters

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 et al. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. ICML 2017.

Recap: Model-Agnostic Meta Learning (MAML)

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



- This brings up **second-order** derivatives
- Supported by standard deep learning libraries such as PyTorch/TensorFlow

Slide courtesy: Zhang, SJTU CS420, 2019

Recall Lecture#7: Meta-learning data weights/augmentation

- Can we learn ϕ_i automatically?

$$\min_{\theta} - \mathbb{E}_{x_i \sim \mathcal{D}} \left[\phi_i \log p_\theta(x_i) \right]$$

- Training set $\mathcal{D}$, a held-out "validation" set $\mathcal{D}_v$
- Intuition: after training the model θ on the weighted data, the model gets better performance on the validation set

$$\theta' = \operatorname{argmin}_{\theta} - \mathbb{E}_{x_i \sim \mathcal{D}} \left[\phi_i \log p_\theta(x_i) \right]$$

  - θ' is a function of ϕ, i.e., $\theta' = \theta'(\phi)$

$$\phi' = \operatorname{argmin}_{\phi} - \mathbb{E}_{x_i \sim \mathcal{D}_v} \left[\log p_{\theta'(\phi)}(x_i) \right]$$

Ren et al., "Learning to reweight examples for robust deep learning"

Hu et al., "Learning Data Manipulation for Augmentation and Weighting"

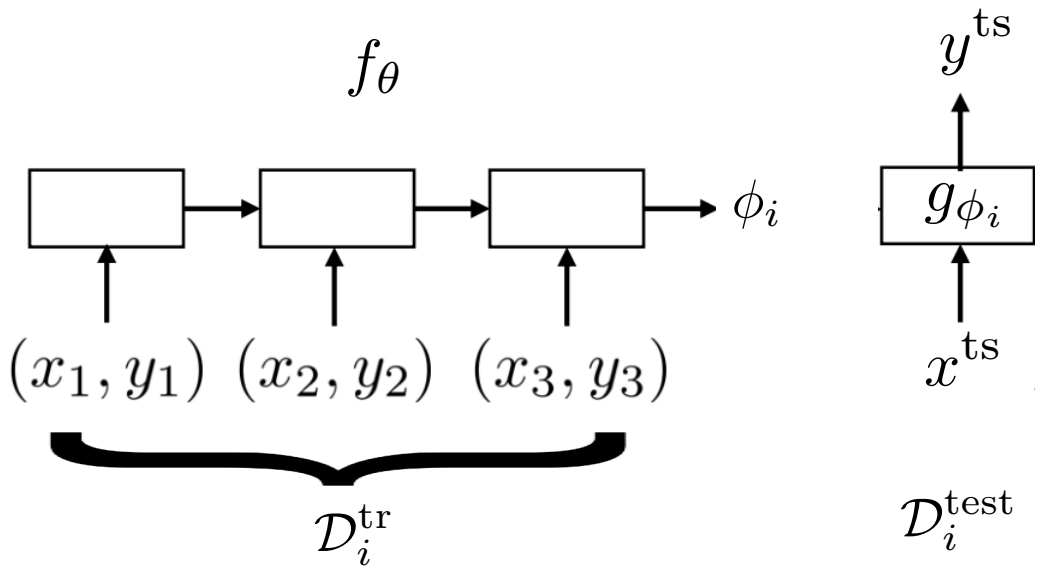
Meta-Learning Methods

- Initialization based methods
  - Learning how to initialize the model for the new task
- Black-box adaptation methods
- Non-parametric methods

Black-Box Adaptation

Key idea: Train a neural network to represent $\phi_i = f_\theta(\mathcal{D}_i^{\text{tr}})$

Predict test points with $\mathbf{y}^{\text{ts}} = g_{\phi_i}(\mathbf{x}^{\text{ts}})$

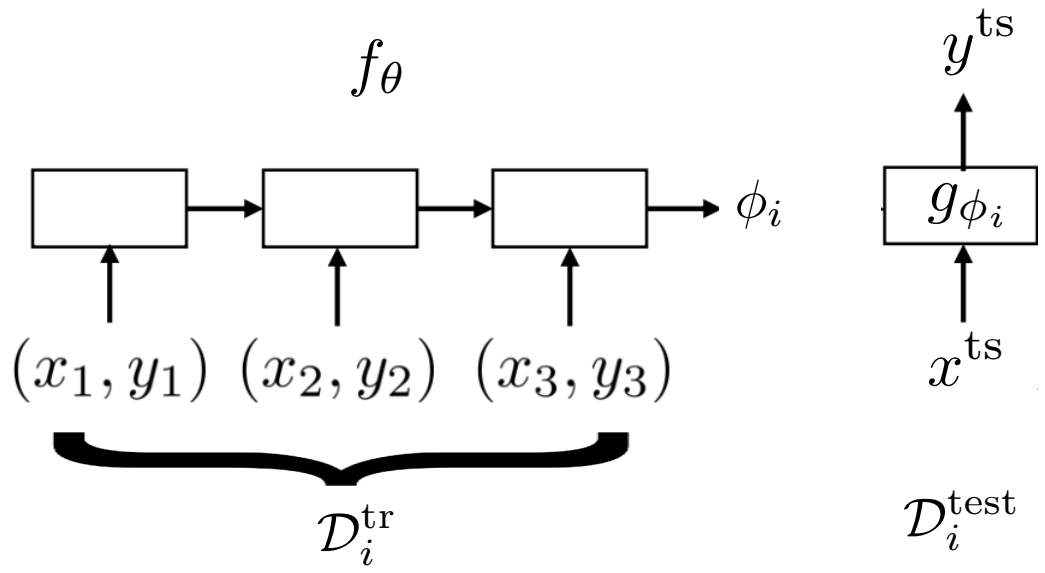


Slide courtesy: Finn, Stanford CS 330, Fall 2021

Black-Box Adaptation

Key idea: Train a neural network to represent $\phi_i = f_\theta(\mathcal{D}_i^{\text{tr}})$

Predict test points with $\mathbf{y}^{\text{ts}} = g_{\phi_i}(\mathbf{x}^{\text{ts}})$



Train with standard supervised learning!

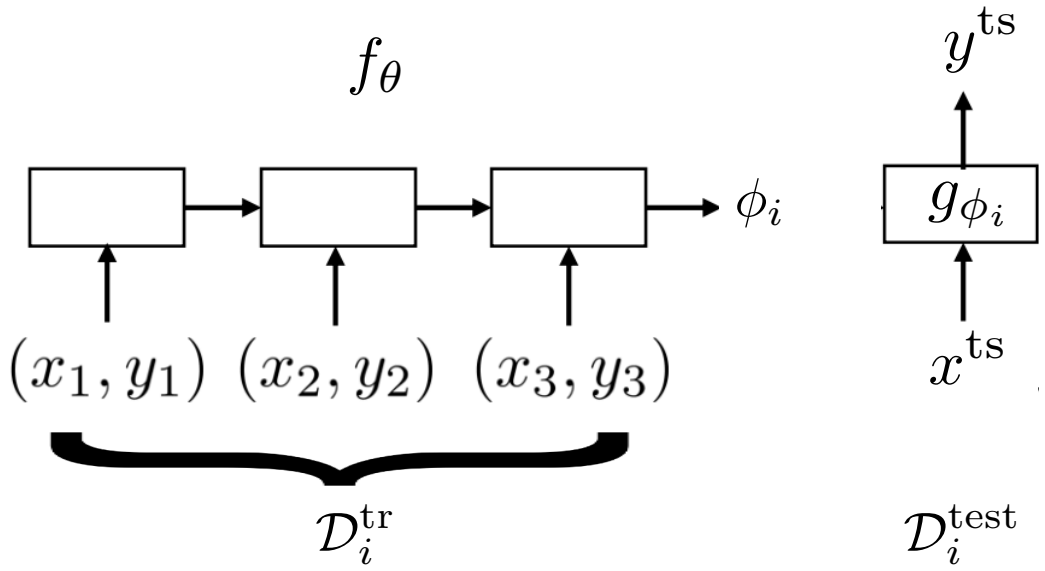
$$\max_\theta \sum_{\mathcal{T}_i} \underbrace{\sum_{(x,y)\sim\mathcal{D}_i^{\text{test}}} \log g_{\phi_i}(y|x)}_{\mathcal{L}(\phi_i, \mathcal{D}_i^{\text{test}})}$$

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

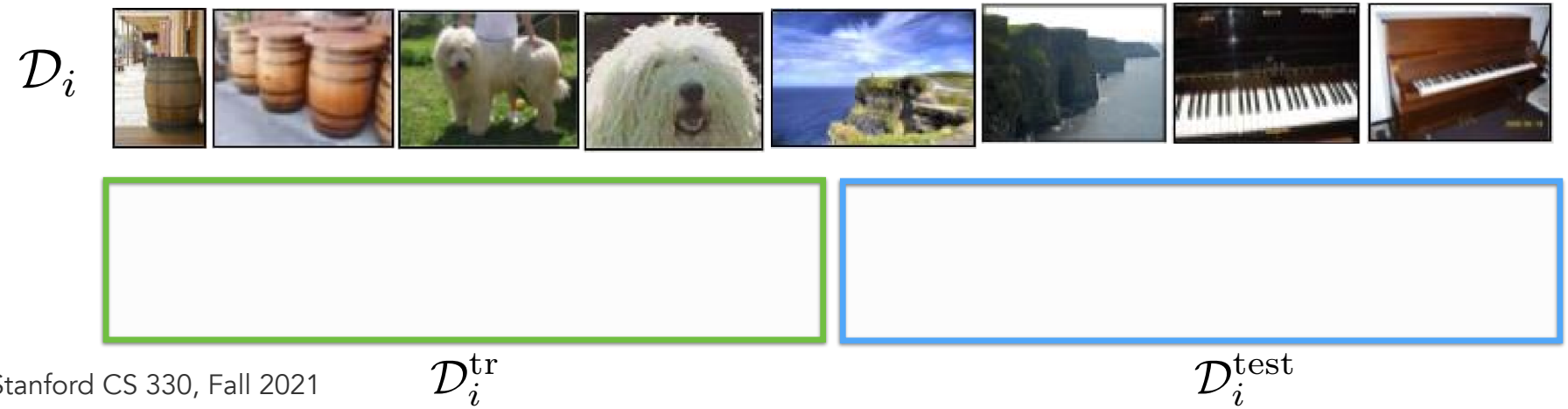
Slide courtesy: Finn, Stanford CS 330, Fall 2021

Black-Box Adaptation

Key idea: Train a neural network to represent $\phi_i = f_\theta(\mathcal{D}_i^{\text{tr}})$.



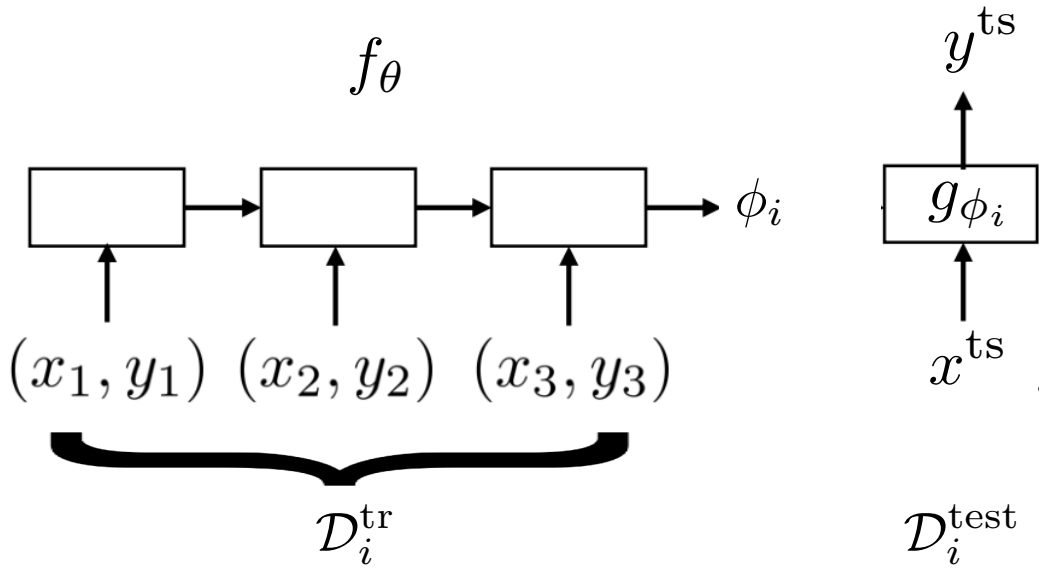
1. Sample task $\mathcal{T}_i$ *(or mini batch of tasks)*
2. Sample disjoint datasets $\mathcal{D}_i^{\text{tr}}, \mathcal{D}_i^{\text{test}}$ from $\mathcal{D}_i$



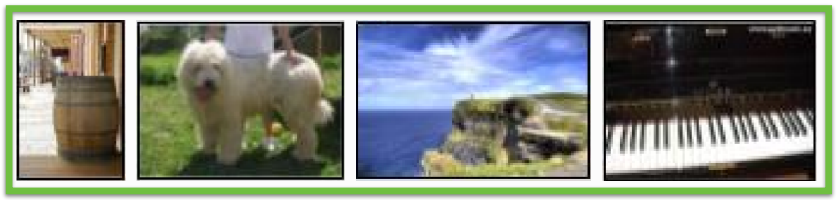
Slide courtesy: Finn, Stanford CS 330, Fall 2021

Black-Box Adaptation

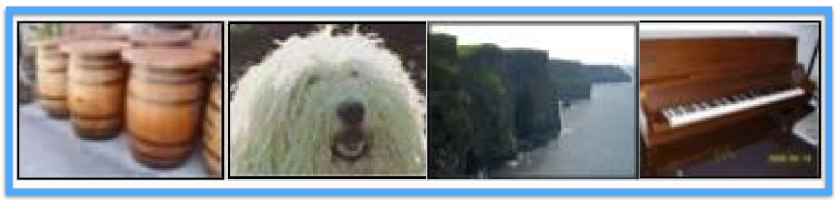
Key idea: Train a neural network to represent $\phi_i = f_\theta(\mathcal{D}_i^{\text{tr}})$.



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 $\nabla_\theta \mathcal{L}(\phi_i, \mathcal{D}_i^{\text{test}})$



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



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

Slide courtesy: Finn, Stanford CS 330, Fall 2021

Black-Box Adaptation

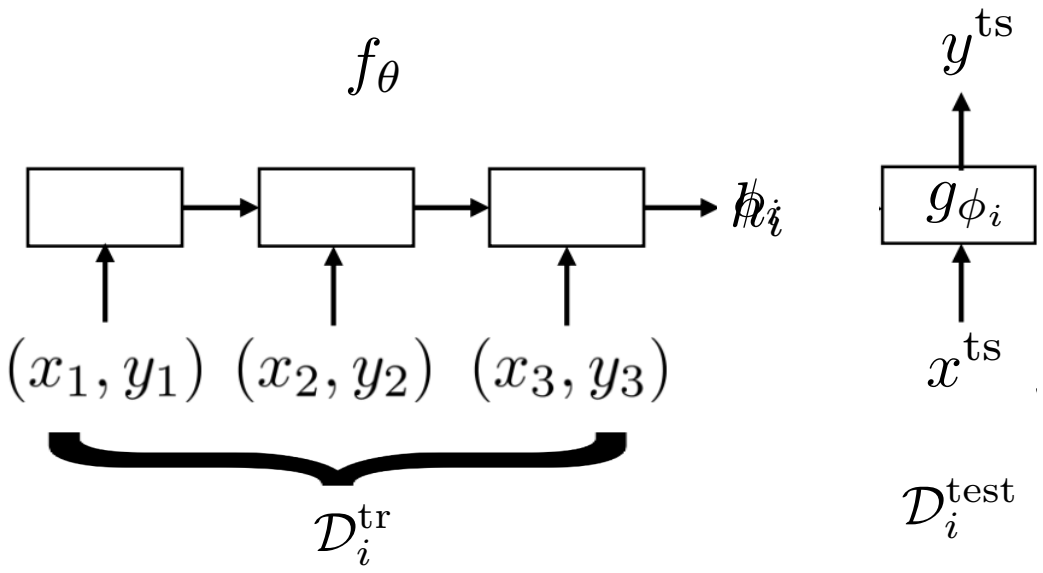
Key idea: Train a neural network to represent $\phi_i = f_\theta(\mathcal{D}_i^{\text{tr}})$.

Challenge

Outputting all neural net parameters does not seem scalable?

Idea: Do not need to output **all** parameters of neural net, only sufficient statistics

(Santoro et al. MANN, Mishra et al. SNAIL)

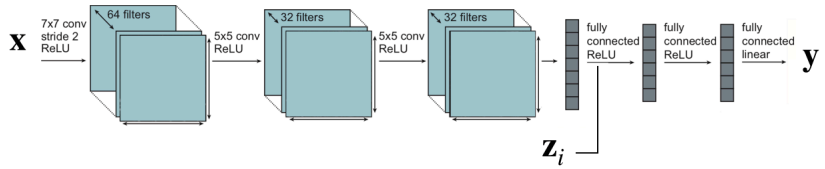


low-dimensional vector h_i

represents contextual task information

$$\phi_i = \{h_i, \theta_g\}$$

recall:



general form: $y^{\text{ts}} = f_\theta(\mathcal{D}_i^{\text{tr}}, x^{\text{ts}})$

What architecture should we use for f_θ**?**

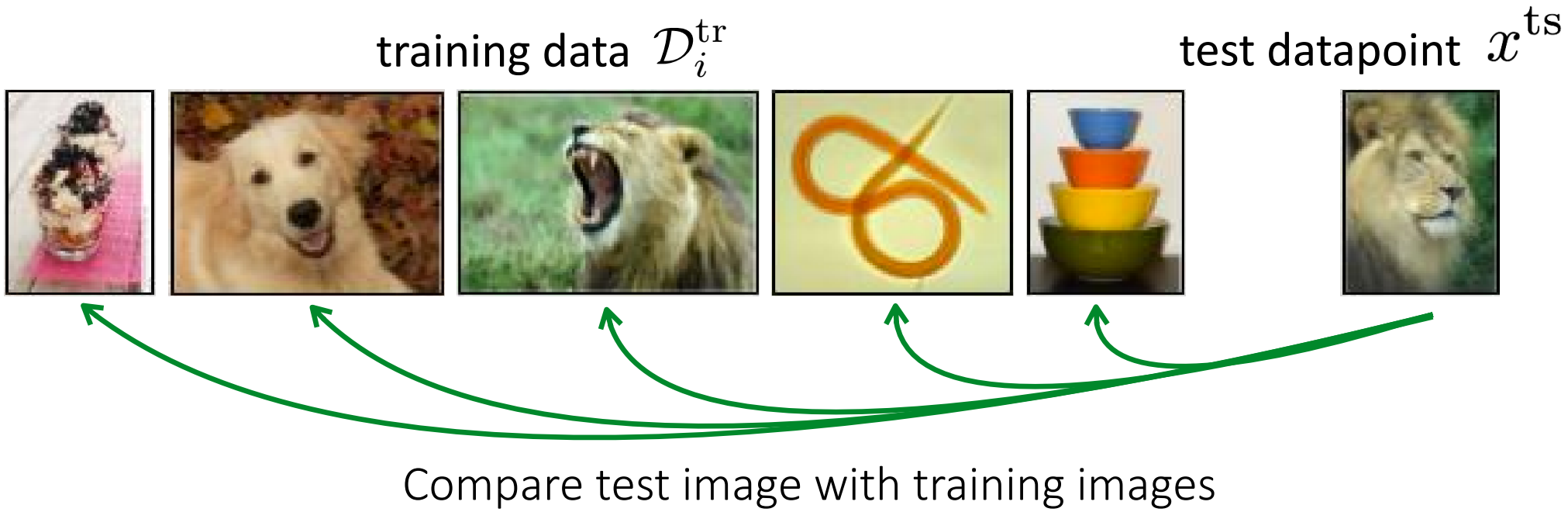
Slide courtesy: Finn, Stanford CS 330, Fall 2021

Meta-Learning Methods

- Initialization based methods
  - Learning how to initialize the model for the new task
- Black-box adaptation methods
- Non-parametric methods

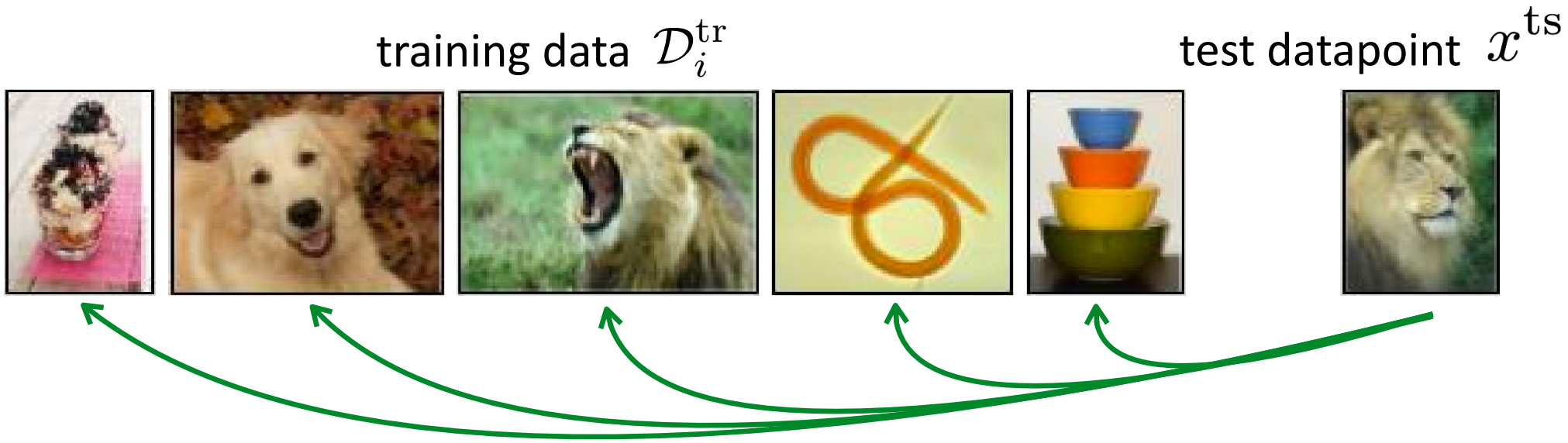
Non-parametric methods

Key Idea: Use non-parametric learner.



Non-parametric methods

Key Idea: Use non-parametric learner.



Compare test image with training images

In what space do you compare? With what distance metric?

ℓ_2 distance in pixel space?

Slide courtesy: Finn, Stanford CS 330, Fall 2021

In what space do you compare? With what distance metric?

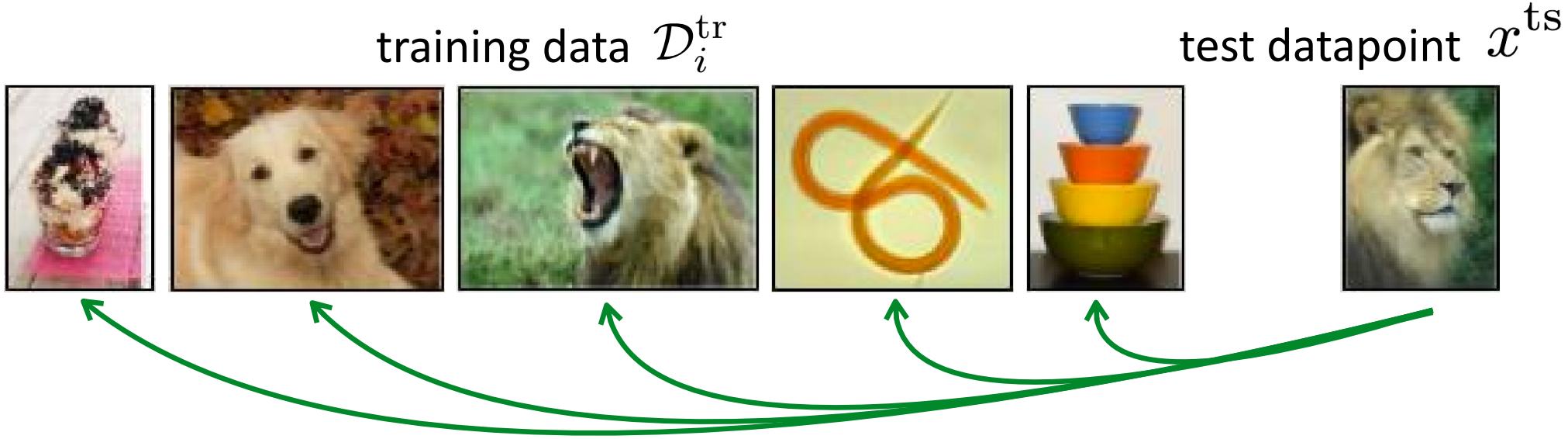
ℓ_2 distance in pixel space?

Slide courtesy: Finn, Stanford CS 330, Fall 2021

Zhang et al. (arXiv 1801.03924)

Non-parametric methods

Key Idea: Use non-parametric learner.



Compare test image with training images

In what space do you compare? With what distance metric?

~~ℓ_2 distance in pixel space?~~

Question: What distance metric would you use instead?

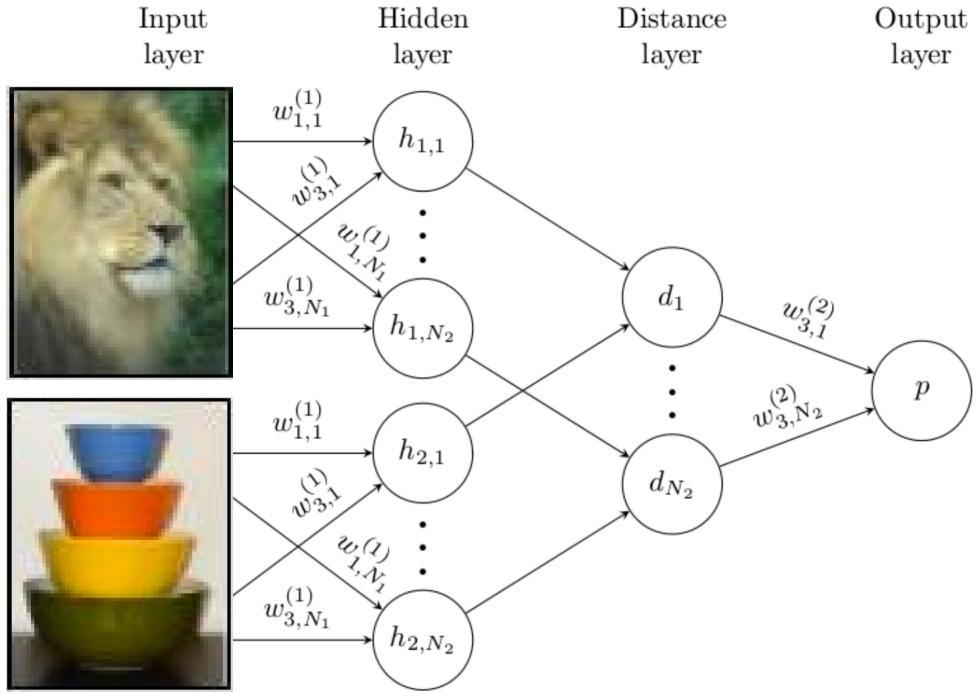
Idea: Learn to compare using meta-training data

Slide courtesy: Finn, Stanford CS 330, Fall 2021

Non-parametric methods

Key Idea: Use non-parametric learner.

train Siamese network to predict whether or not two images are the same class



label

0

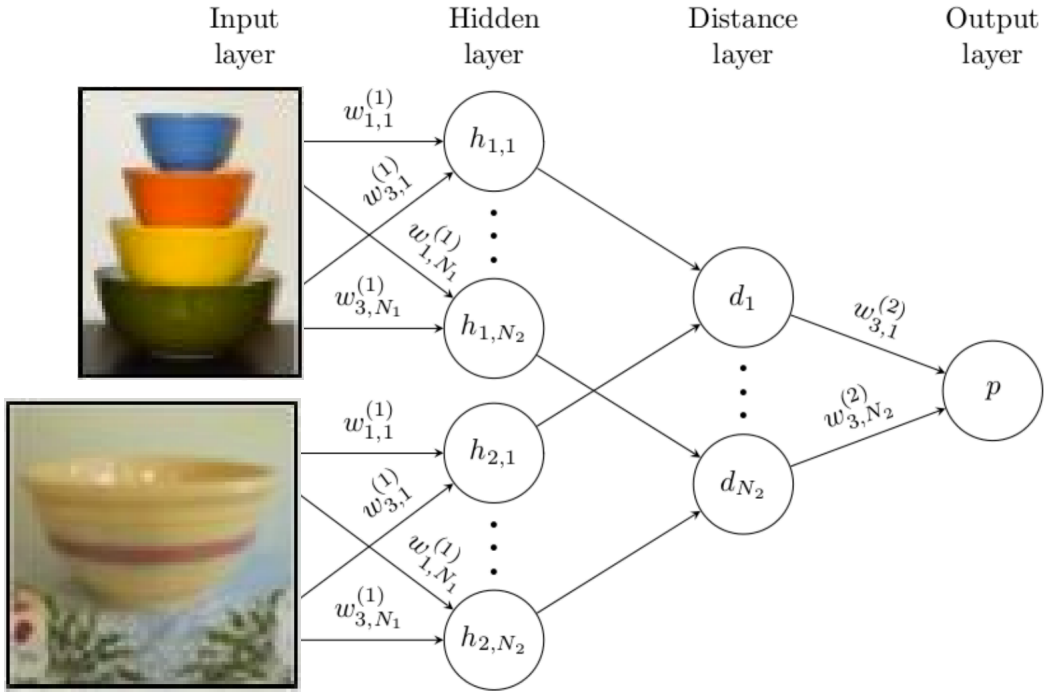
Slide courtesy: Finn, Stanford CS 330, Fall 2021

Koch et al., ICML '15

Non-parametric methods

Key Idea: Use non-parametric learner.

train Siamese network to predict whether or not two images are the same class



label

1

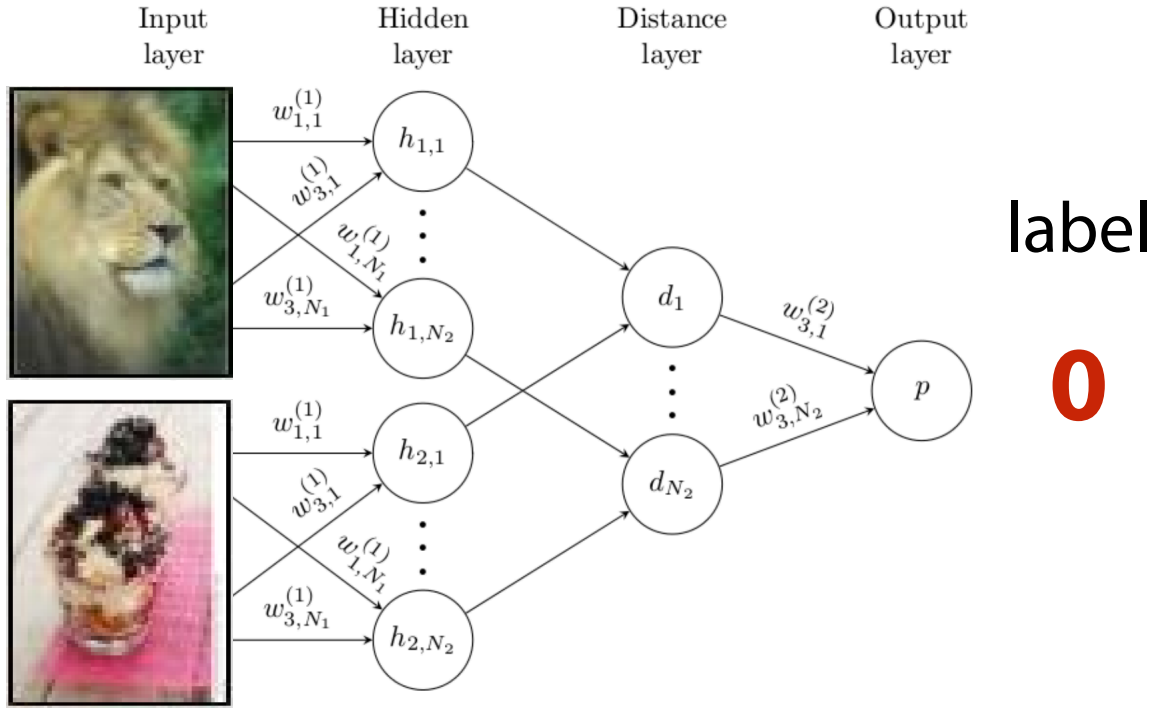
Slide courtesy: Finn, Stanford CS 330, Fall 2021

Koch et al., ICML '15

Non-parametric methods

Key Idea: Use non-parametric learner.

train Siamese network to predict whether or not two images are the same class



label

0

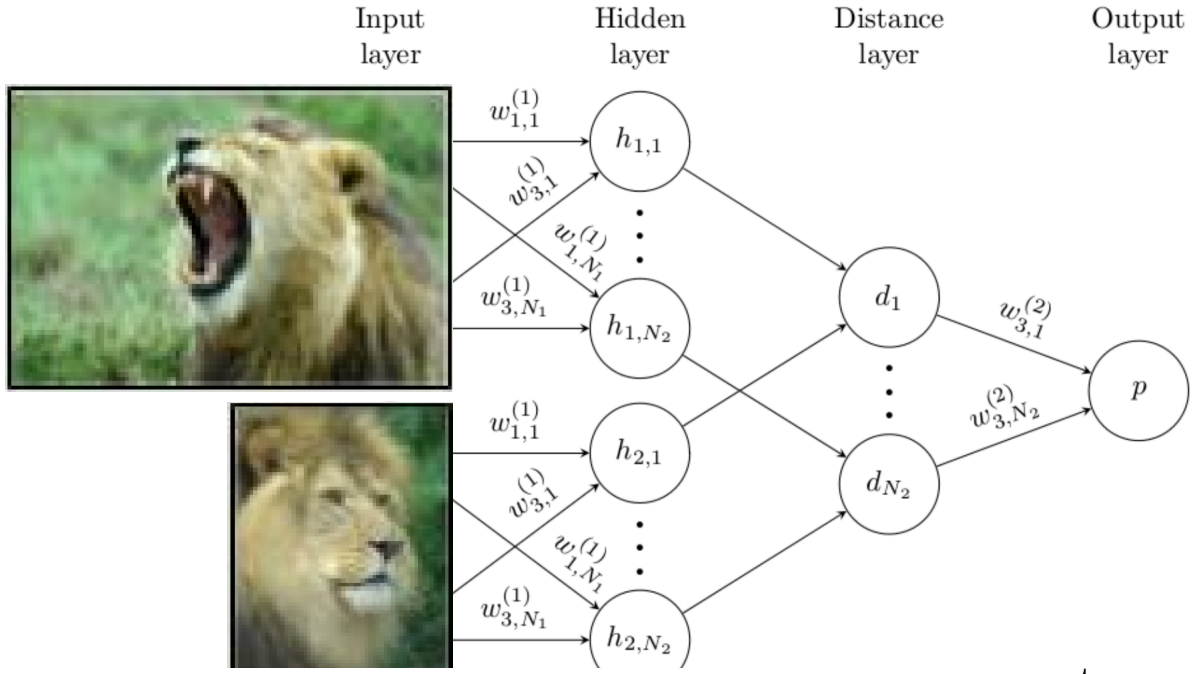
Slide courtesy: Finn, Stanford CS 330, Fall 2021

Koch et al., ICML '15

Non-parametric methods

Key Idea: Use non-parametric learner.

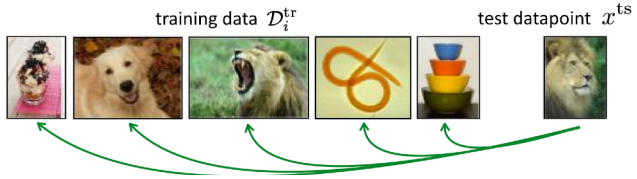
train Siamese network to predict whether or not two images are the same class



label

1

Meta-test time: compare image $\mathbf{x}_{\text{test}}$ to each image in $\mathcal{D}_j^{\text{tr}}$



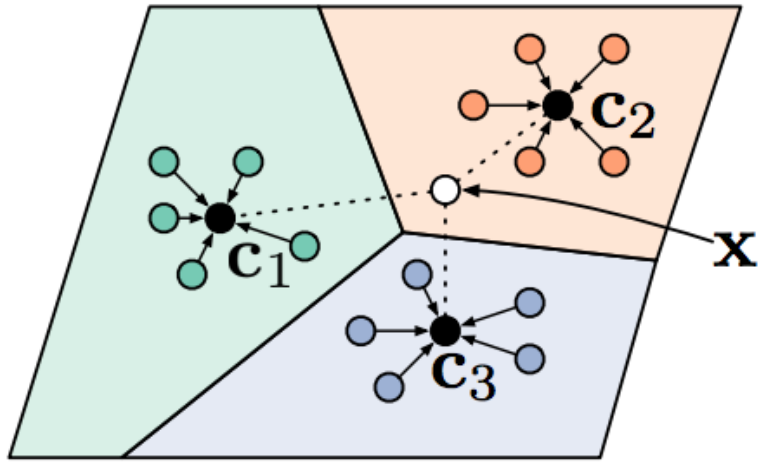
Slide courtesy: Finn, Stanford CS 330, Fall 2021

Koch et al., ICML '15

Non-parametric methods

Key Idea: Use non-parametric learner.

Prototypes: aggregate class information to create a prototypical embedding



$$\mathbf{c}_n = \frac{1}{K} \sum_{(x,y) \in \mathcal{D}_i^{\text{tr}}} \mathbb{1}(y = n) f_\theta(x)$$

$$p_\theta(y = n|x) = \frac{\exp(-d\left(f_\theta(x), \mathbf{c}_n\right))}{\sum_{n'} \exp(d(f_\theta(x), \mathbf{c}_{n'}))}$$

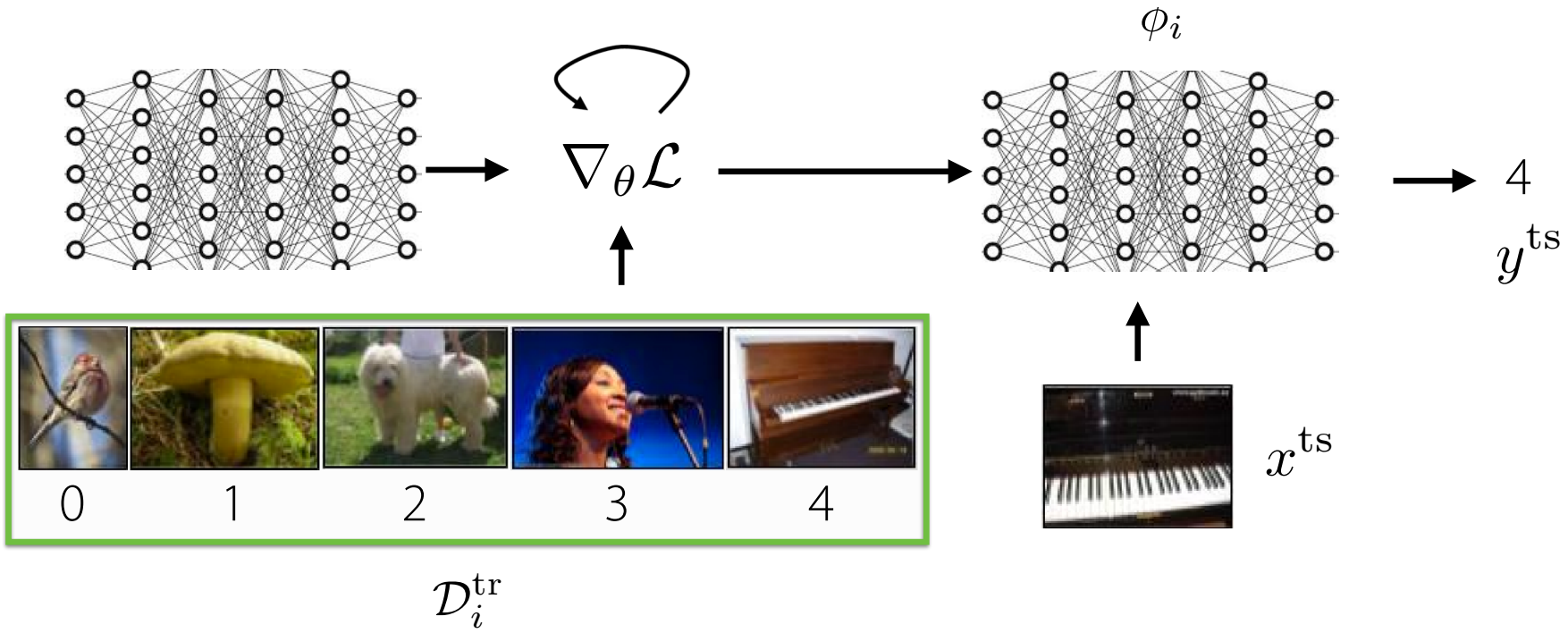
d: Euclidean, or cosine distance

Slide courtesy: Finn, Stanford CS 330, Fall 2021

Snell et al. Prototypical Networks, NeurIPS '17

Summary: Meta-Learning Methods

- Initialization based methods



Key idea: embed optimization inside the inner learning process

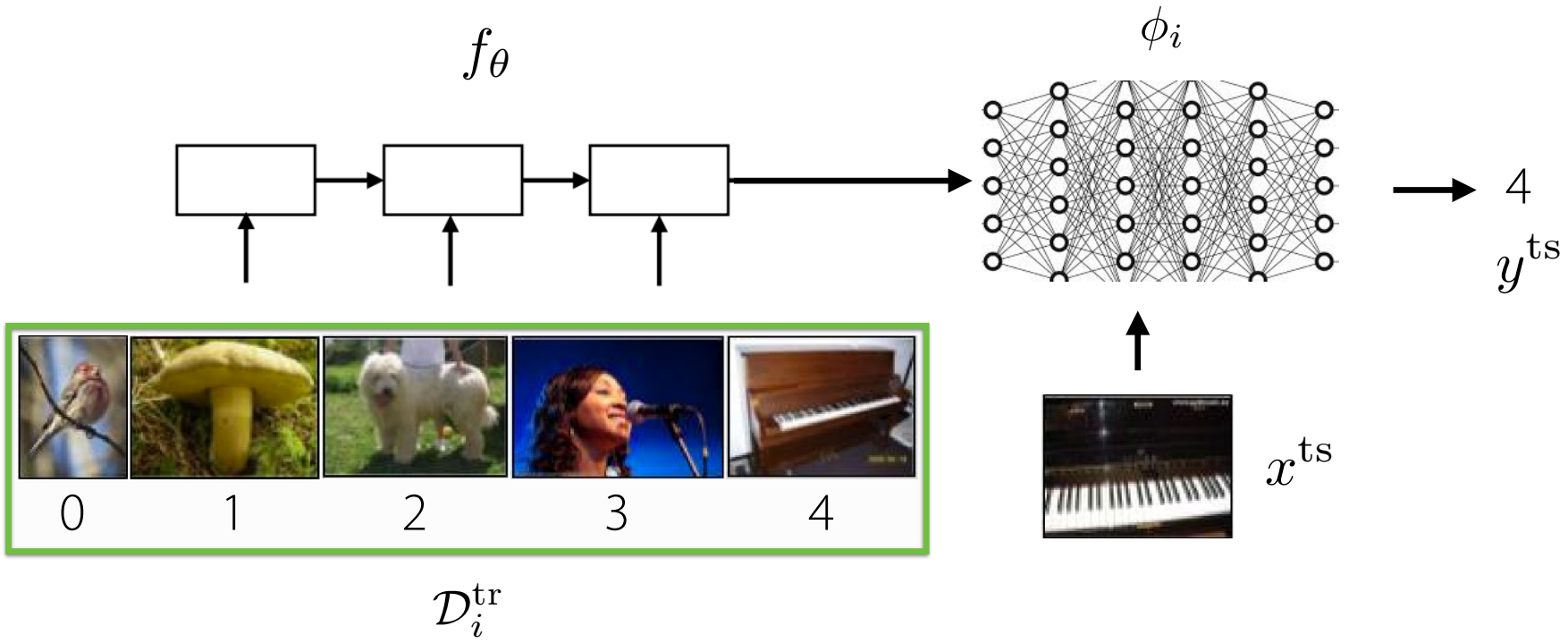
+ **structure** of **optimization** embedded into meta-learner

- typically requires **second-order optimization**

Slide courtesy: Finn, Stanford CS 330, Fall 2021

Summary: Meta-Learning Methods

- Black-box adaptation methods



Key idea: parametrize learner as a neural network

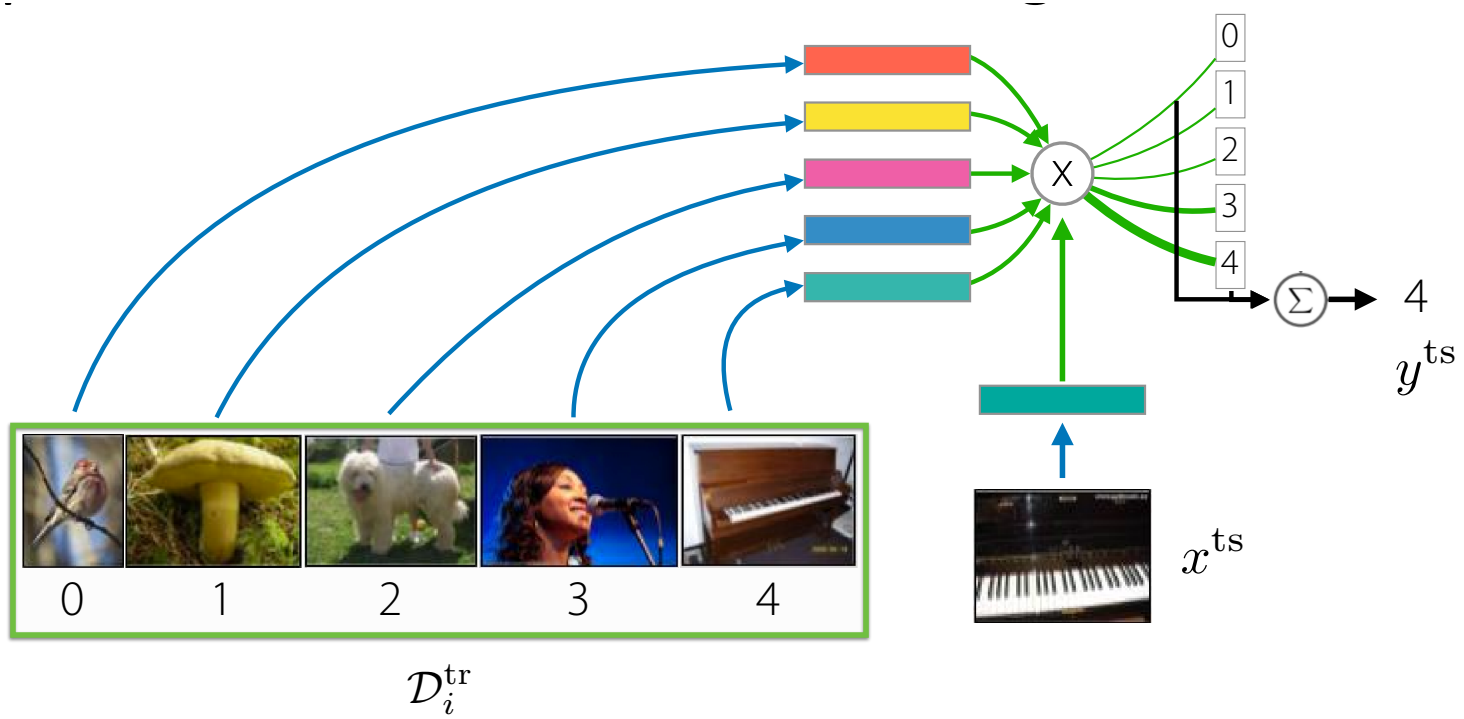
Slide courtesy: Finn, Stanford CS 330, Fall 2021

+ expressive

- challenging optimization problem

Summary: Meta-Learning Methods

- Non-parametric methods



Key idea: *non-parametric learner* with *parametric* embedding / distance
(e.g. kNN to examples/prototypes)

+ easy to optimize, computationally fast

- largely restricted to classification

Slide courtesy: Finn, Stanford CS 330, Fall 2021

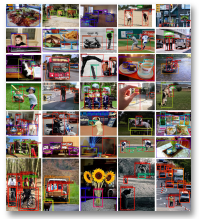
Key Takeaways

- Learning with multiple tasks:
  - Multi-task learning, transfer learning, mete-learning
- Meta-learning problem setting
- Meta-learning methods
  - Initialization-based methods
  - Black-box adaptation
  - Non-parametric methods

Recap: ML solutions given few data (labels)

- How can we make more efficient use of the data?
  - Clean but small-size
  - Noisy
  - Out-of-domain

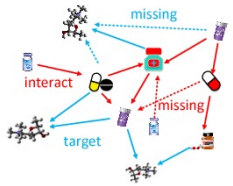
- Can we incorporate other types of experiences in learning?



Data examples

Type-2 diabetes is 90% more common than type-1

Rules/Constraints

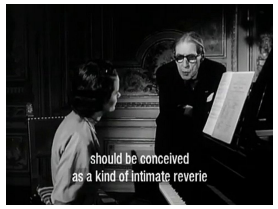


Knowledge graphs

Rewards

Auxiliary agents

Adversaries



Master classes

…

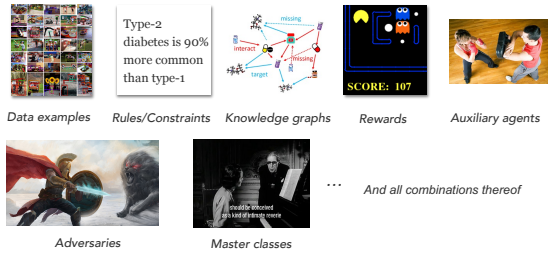
And all combinations thereof

Machine learning solutions given few data (labels)

- (1) How can we make more efficient use of the data?
  - Clean but small-size, Noisy, Out-of-domain, …
- Algorithms
  - **Supervised learning**: MLE, maximum entropy principle
  - **Unsupervised learning**: EM, variational inference, VAEs
    - Key concept: ELBO
  - **Self-supervised learning**: successful instances, e.g., BERT, GPT-3, contrastive learning, applications on image/video/text
  - **Distant/weakly supervised learning**: successful instances
  - **Data manipulation:** augmentation, re-weighting, curriculum learning, …
  - **Meta-learning**

Machine learning solutions given few data (labels)

- (2) Can we incorporate other types of experiences in learning?
  - Learning from auxiliary models, e.g., adversarial models:
    - Generative adversarial learning (GANs and variants)
  - Learning from structured knowledge
    - Weakly supervised learning, Posterior regularization
  - Learning from rewards
    - Reinforcement learning: MDP, Bellman equation
    - policy gradient, Q-learning
  - Learning in dynamic environment *(not covered)*
    - Online learning, lifelong/continual learning, …



Questions?