Meta-Learning

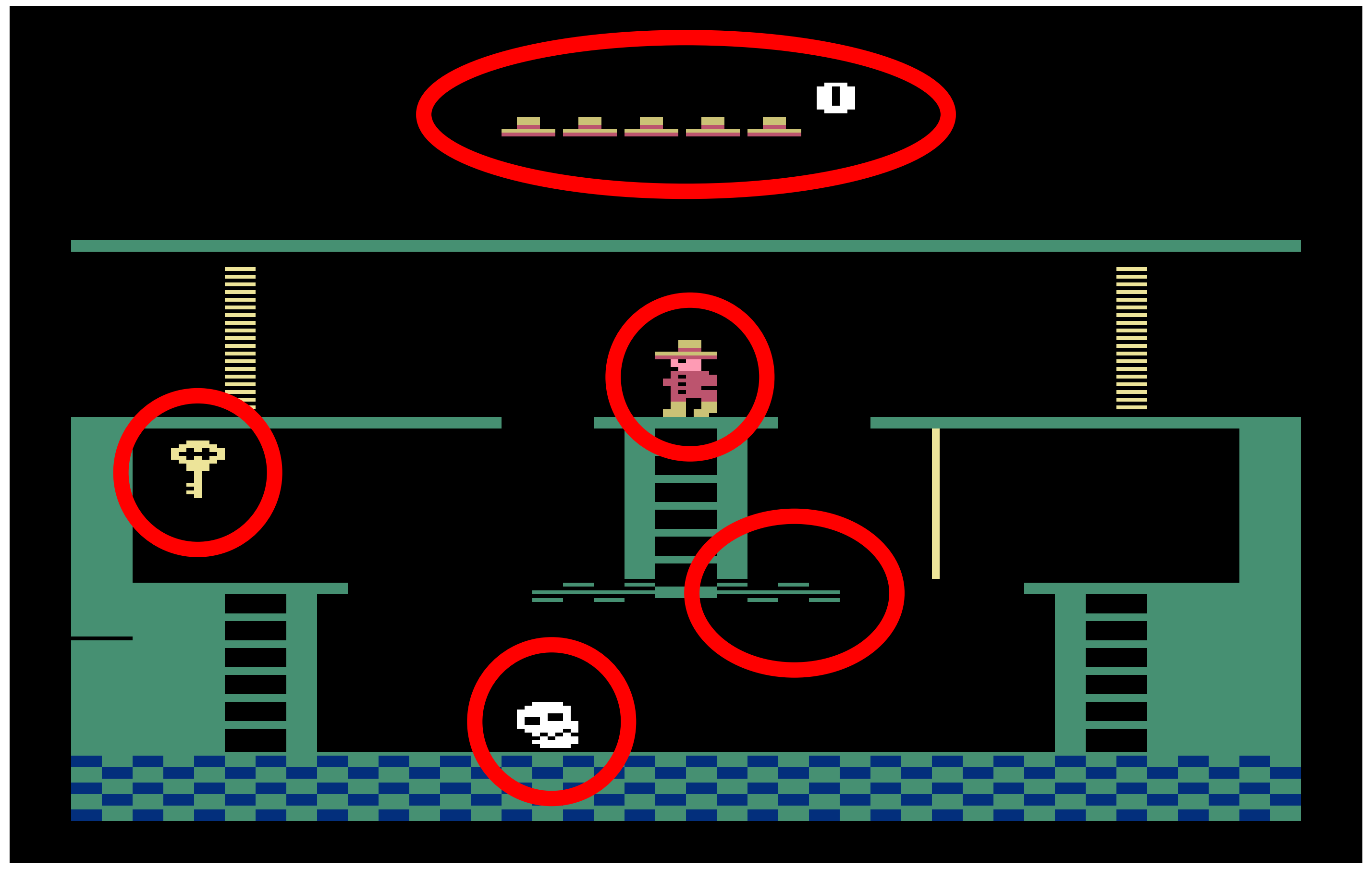
John Canny

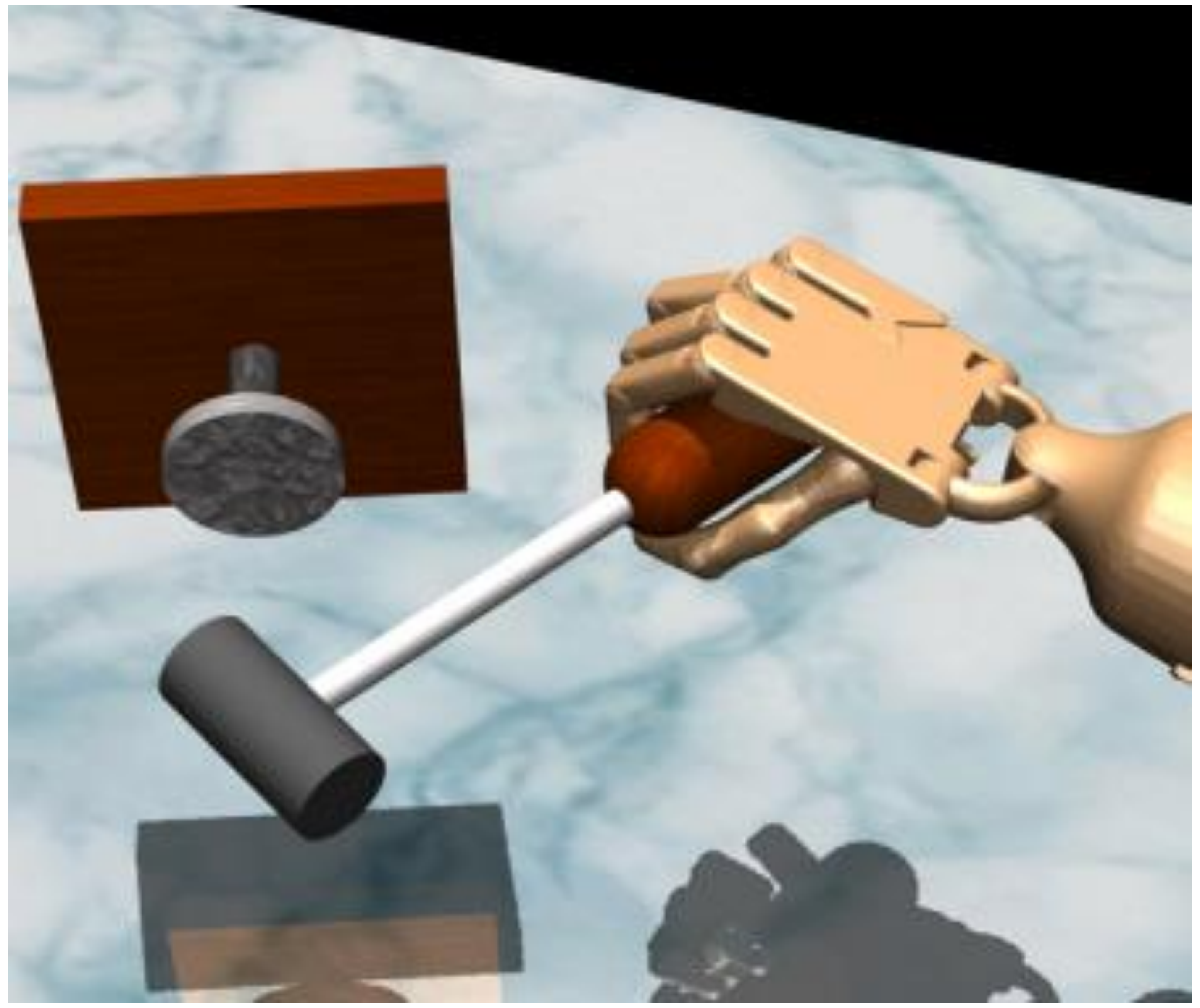
Spring 2019

Lecture 23 of CS182/282A: Designing, Visualizing and Understanding Deep Neural Networks

Most slides by Chelsea Finn, 2018

Last Time: Exploring with pseudo-counts





fit model $p_\theta(\mathbf{s})$ to all states $\mathcal{D}$ seen so far

take a step i and observe $\mathbf{s}_i$

fit new model $p_{\theta'}(\mathbf{s})$ to $\mathcal{D} \cup \mathbf{s}_i$

use $p_\theta(\mathbf{s}_i)$ and $p_{\theta'}(\mathbf{s}_i)$ to estimate $\hat{N}(\mathbf{s})$

set $r_i^+ = r_i + \mathcal{B}(\hat{N}(\mathbf{s}))$ ← "pseudo-count"

how to get $\hat{N}(\mathbf{s})$? use the equations

$$p_\theta(\mathbf{s}_i) = \frac{\hat{N}(\mathbf{s}_i)}{\hat{n}} \qquad p_{\theta'}(\mathbf{s}_i) = \frac{\hat{N}(\mathbf{s}_i) + 1}{\hat{n} + 1}$$

two equations and two unknowns!

$$\hat{N}(\mathbf{s}_i) = \hat{n} p_\theta(\mathbf{s}_i) \qquad \hat{n} = \frac{1 - p_{\theta'}(\mathbf{s}_i)}{p_{\theta'}(\mathbf{s}_i) - p_\theta(\mathbf{s}_i)} p_\theta(\mathbf{s}_i)$$

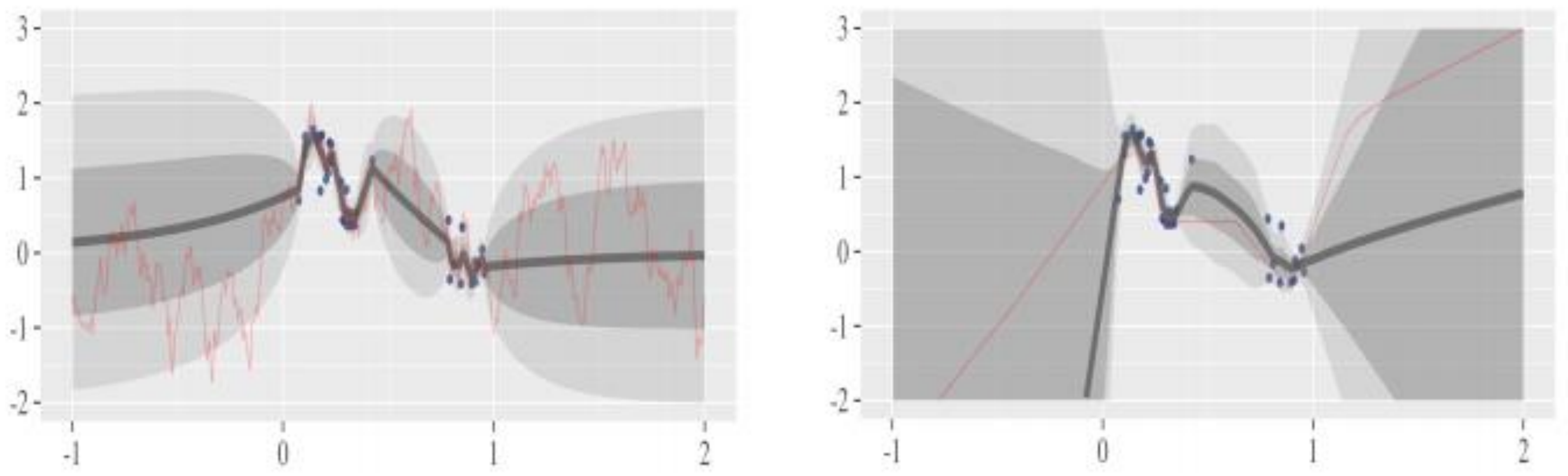
Bellemare et al. "Unifying Count-Based Exploration…"

Last Time: Bootstrap

Given a dataset $\mathcal{D}$, resample with replacement N times to get bootstrap datasets $\mathcal{D}_1, \dots, \mathcal{D}_N$.

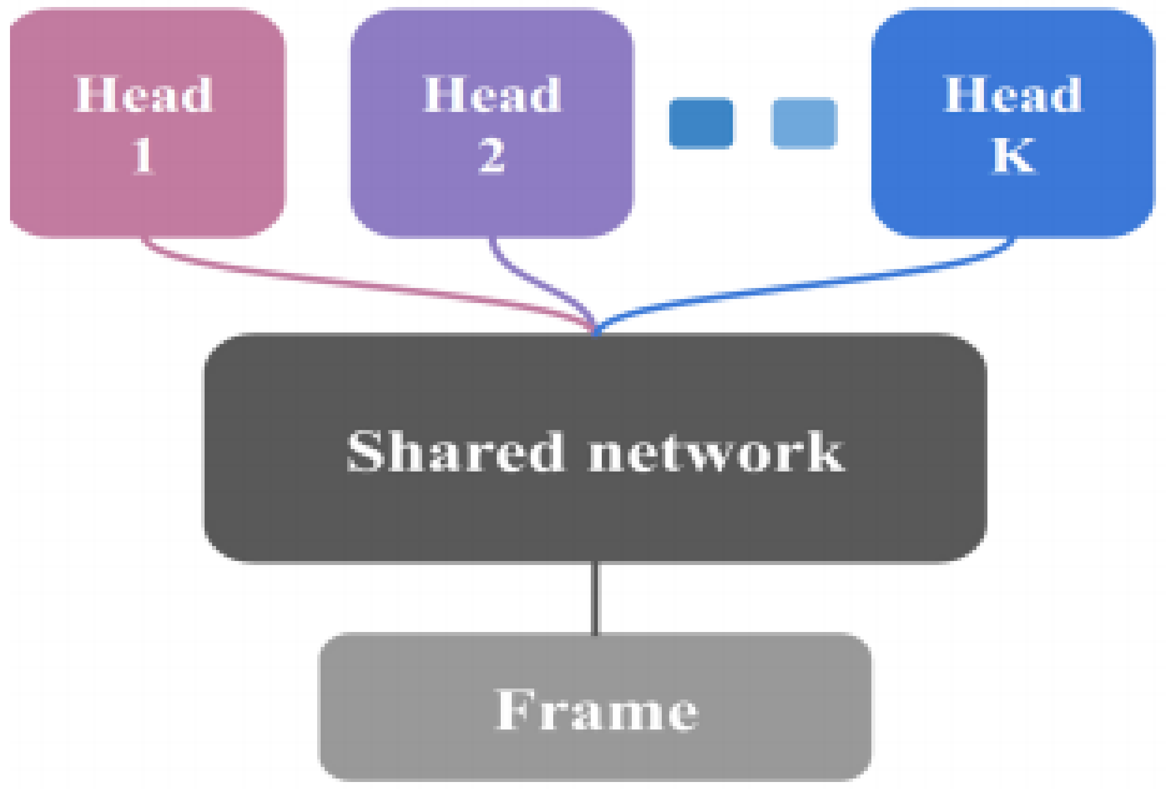
Train a model f_{θ_i} on $\mathcal{D}_i$.

To sample from $p(\theta)$, sample $i \in [1, \dots, N]$, use f_{θ_i}.



(b) Gaussian process posterior (c) Bootstrapped neural nets

But training N large neural networks is expensive, can we avoid it?

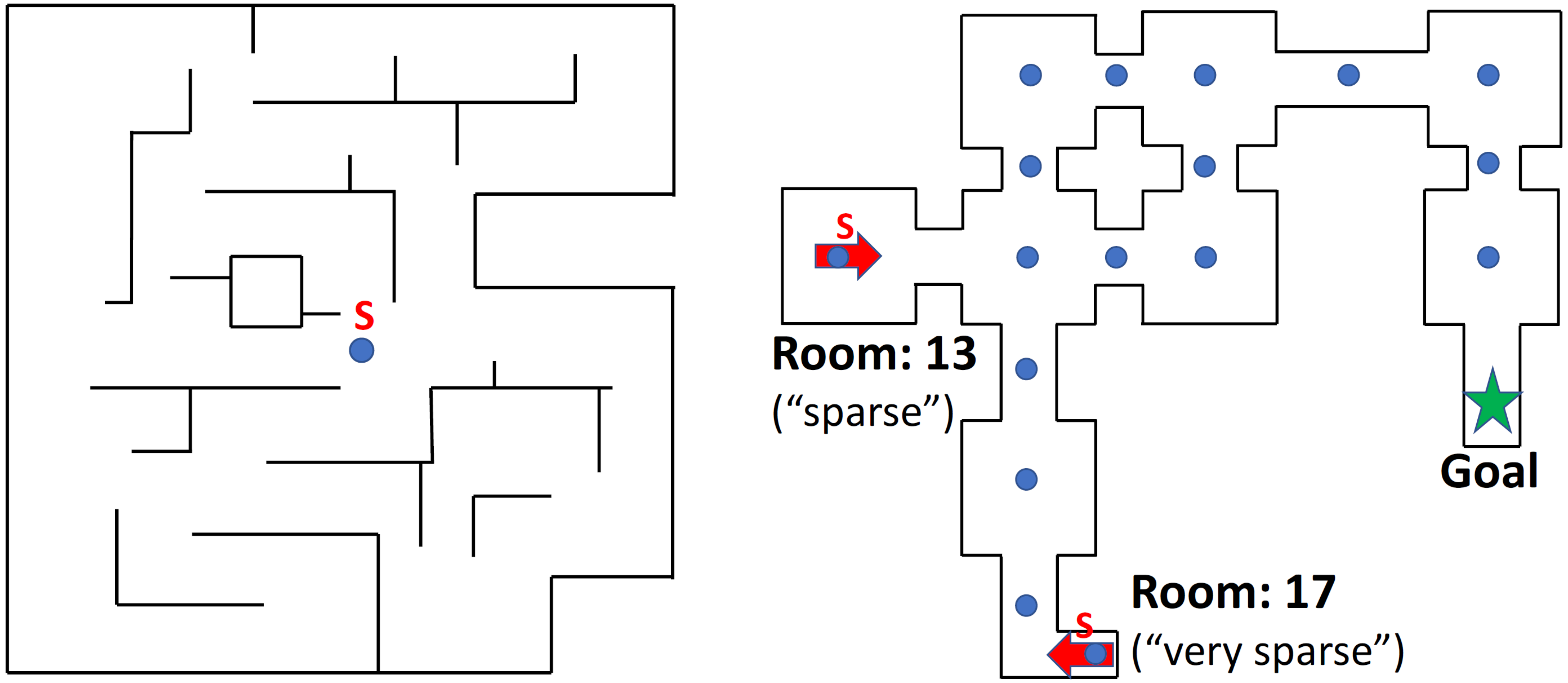


Osband et al. "Deep Exploration via Bootstrapped DQN"

Last Time: Curiosity-Driven Exploration

Pathak et al. 2017

Environments: Vizdoom



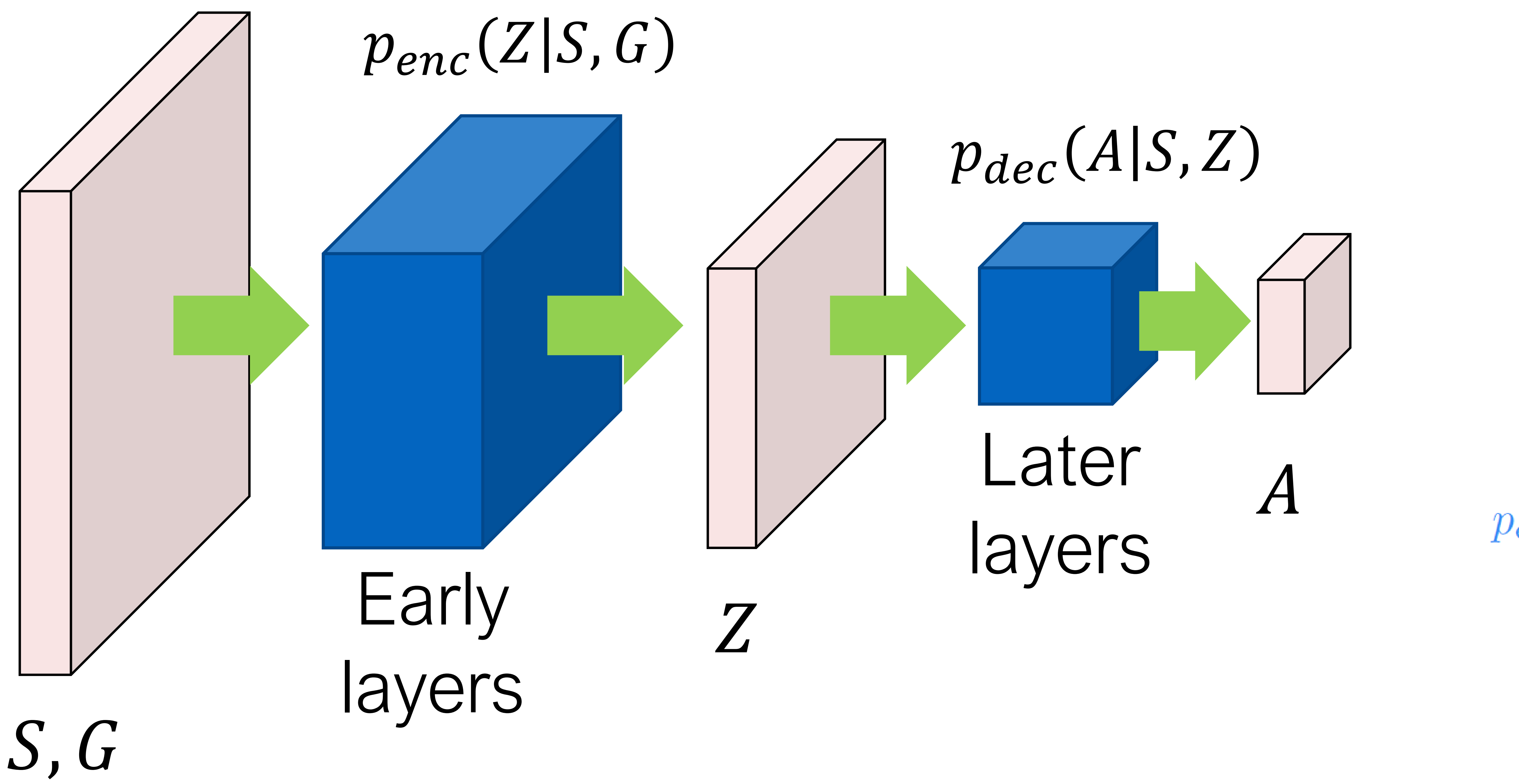
(a) Train Map Scenario (b) Test Map Scenario

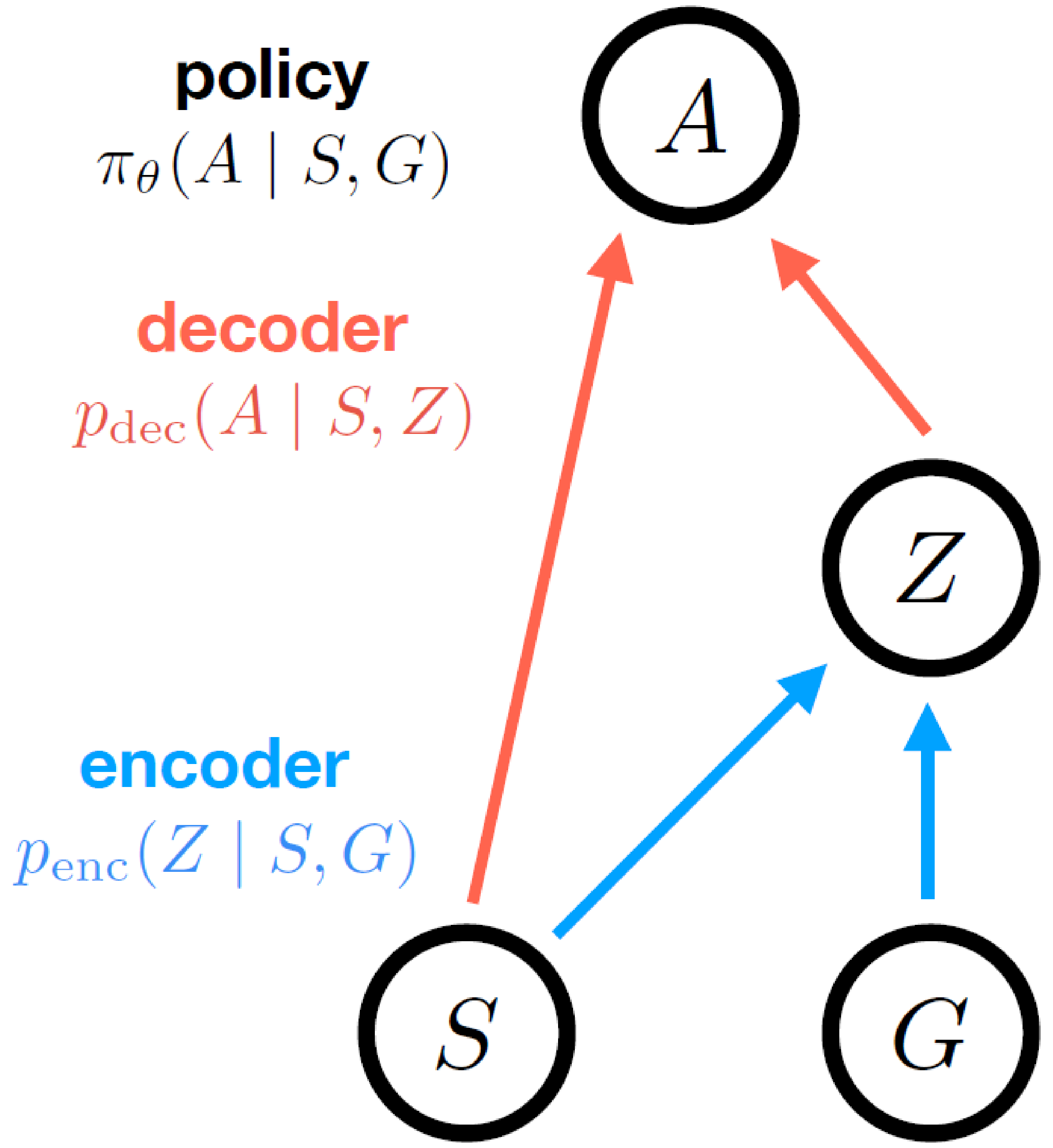
Model is trained initially without goal reward (left). Agent is randomly started on one of the blue dots in the "dense" case (right).

Last Time: InfoBot: Information Bottleneck for Reinforcement Learning

Minimize the Conditional Goal/Action Mutual Information, i.e. minimize the divergence between $\pi_\theta(A|G,S)$ and a default policy $\pi_0(A|S)$ that doesn't depend on G.
The policy has a bottleneck Z and satisfies:

$$\pi_\theta(A|G,S) = \sum_z p_{enc}(z|S,G)p_{dec}(A|S,z)$$





Course Logistics

- HW4 was due Friday.
- Make sure you completed (2nd) project checkin.
- Final Project Poster due Saturday 5/4
- Poster Session is Tuesday May 7th, 2-4pm.
- Course survey is open now, please fill it out.

This class:

1. Get large dataset
2. Get large compute (i.e. GPUs)
3. ?????
4. Profit!

Do neural networks need a large dataset?

Humans can learn with a very small amount of data. How?

Humans don't start from scratch.

Humans accumulate prior experience, acquire common sense.

How can we move away from training from scratch?
Can we develop systems that accumulate prior experience?

Idea: learn the structure underlying previous data/tasks/experiences; use that structure to quickly learn new tasks.

How? One way to do so is by learning to learn.

Disclaimer 1: meta-learning is still in its infancy (though it dates back to the 80/90s in the context of AI).
Disclaimer 2: Could teach an entire course on meta-learning.

Terminology

Learning to learn, **meta-learning**: overarching terms

AutoML: automating the learning process

- hyperparameter optimization
- architecture search

One-shot/few-shot learning: learning to learn from small amounts of data

Meta-reinforcement learning: learning a reinforcement learning algorithm

Meta-X learning: X=imitation, unsupervised, semi-supervised

This lecture: focus on few-shot learning

Outline

1. Applications of learning to learn
2. Problem formulation
3. Solution Classes:
    a) metric-learning approach
    b) direct black-box approach
    c) gradient-based approach
4. Open Questions / Problems

Outline

1. **Applications of learning to learn**
2. Problem formulation
3. Solution Classes:
    a) metric-learning approach
    b) direct black-box approach
    c) gradient-based approach
4. Open Questions / Problems

Application: Few-Shot Image Classification

Given 1 example of 5 classes:

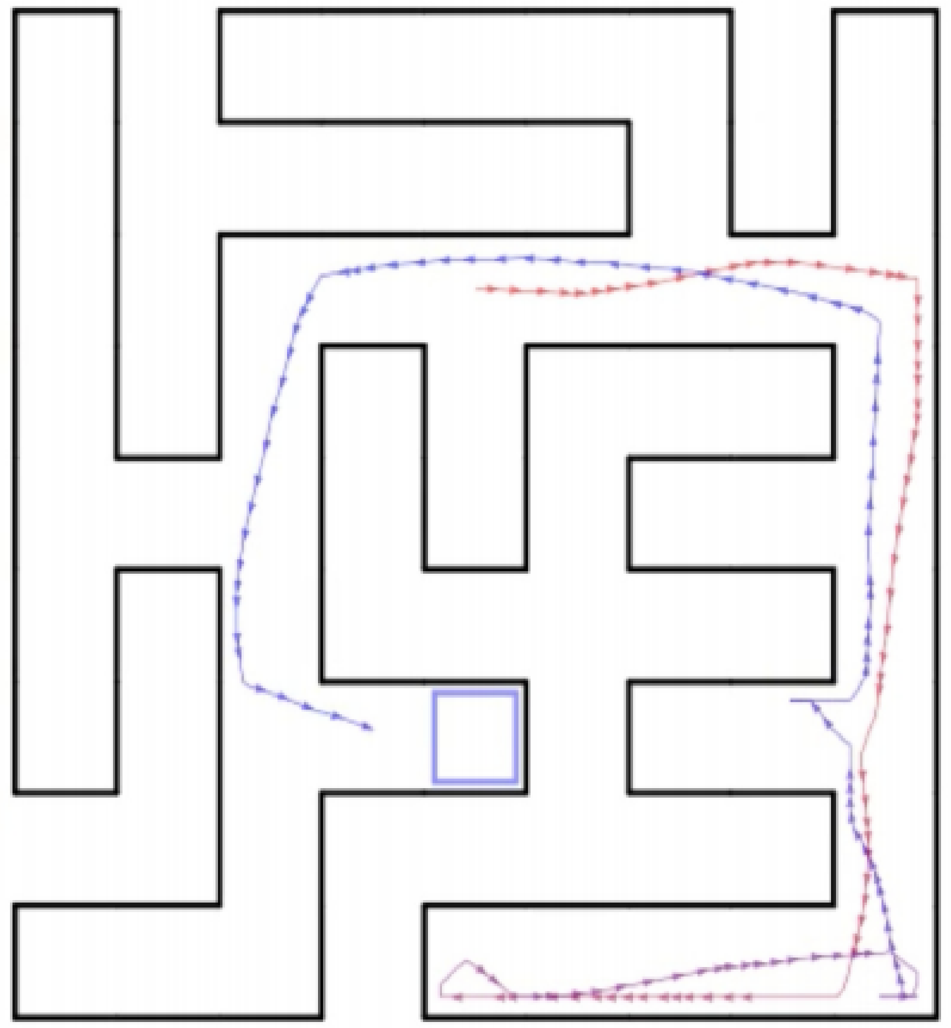
Classify new examples

training classes

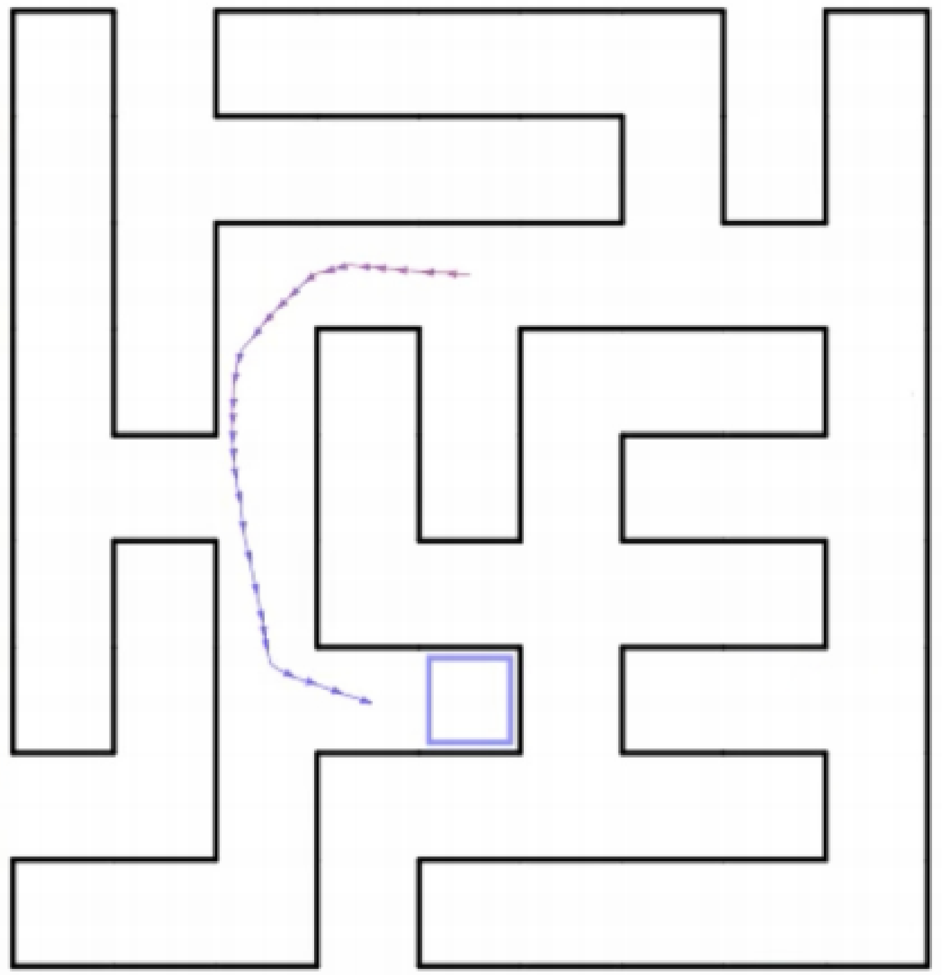
diagram adapted from Ravi & Larochelle ‘17

Application: Fast Reinforcement Learning

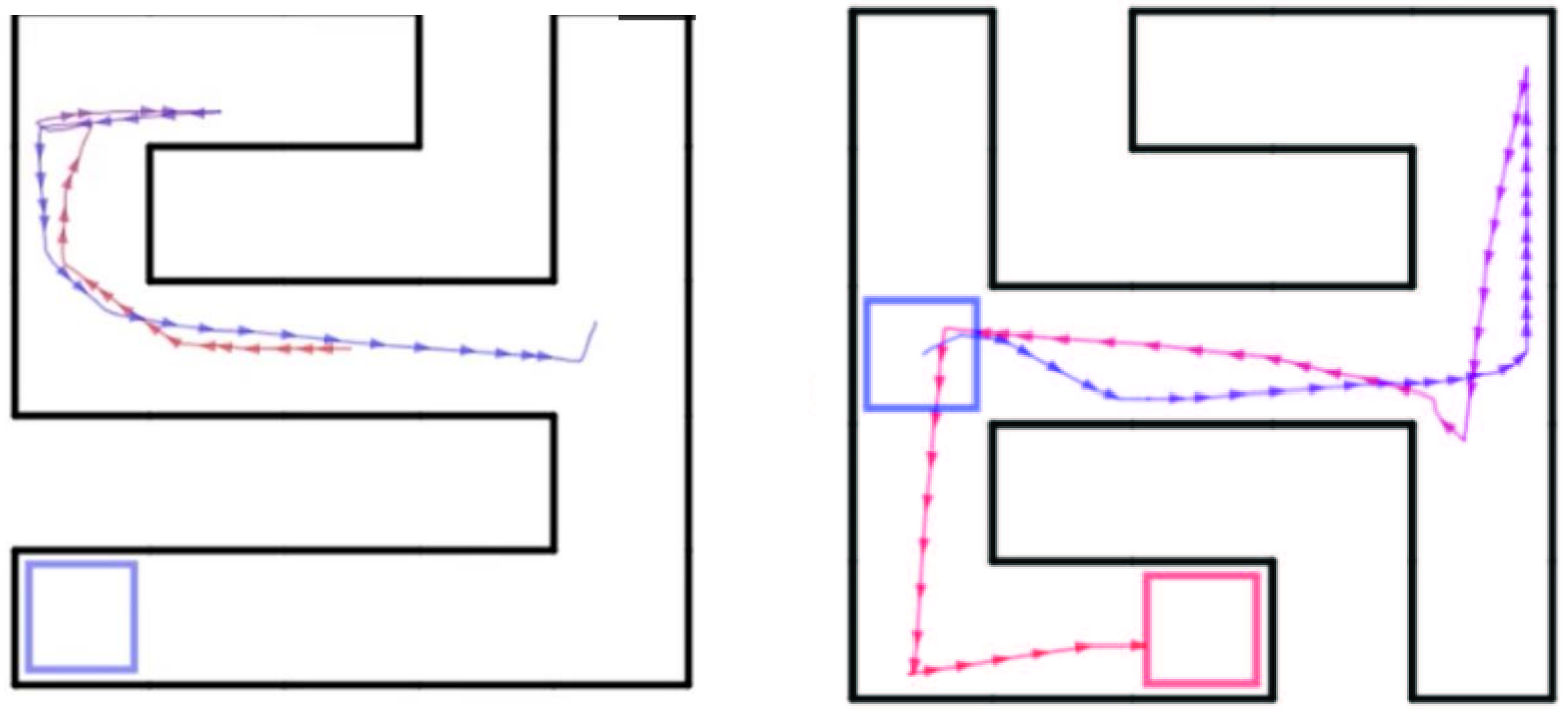
Given a small amount of experience



Learn to solve a task



By learning how to learn many other tasks:



. . .

diagram adapted from Duan et al. '17

Outline

1. Applications of learning to learn
2. **Problem formulation**
3. Solution Classes:
   a) metric-learning approach
   b) direct black-box approach
   c) gradient-based approach
4. Open Questions / Problems

The Meta-Learning Problem

Supervised Learning:

Inputs: $\mathbf{x}$ Outputs: $\mathbf{y}$ Data: $\{(\mathbf{x},\mathbf{y})_i\}$

$$\mathbf{y} = f(\mathbf{x};\theta)$$

Meta-Supervised Learning:

Inputs: $\underbrace{\mathcal{D}_{\text{train}}}_{\{(\mathbf{x},\mathbf{y})_{1:K}\}}$ $\mathbf{x}_{\text{test}}$ Outputs: $\mathbf{y}_{\text{test}}$ Data: $\{\mathcal{D}_i\}$

$\mathcal{D}_i : \{(\mathbf{x},\mathbf{y})_j\}$

$$\mathbf{y}_{\text{test}} = f(\mathcal{D}_{\text{train}}, \mathbf{x}_{\text{test}};\theta)$$

Why is this view useful?

Reduces the problem to the design & optimization of *f*.

Chelsea Finn, UC Berkeley

Application: Few-Shot Image Classification

Given 1 example of 5 classes:

training data $\mathcal{D}_{\text{train}}$

Classify new examples

test set $\mathbf{x}_{\text{test}}$

meta-training data

$\mathcal{T}_1$

$\mathcal{T}_2$

⋮ ⋮

training classes

diagram adapted from Ravi & Larochelle '17

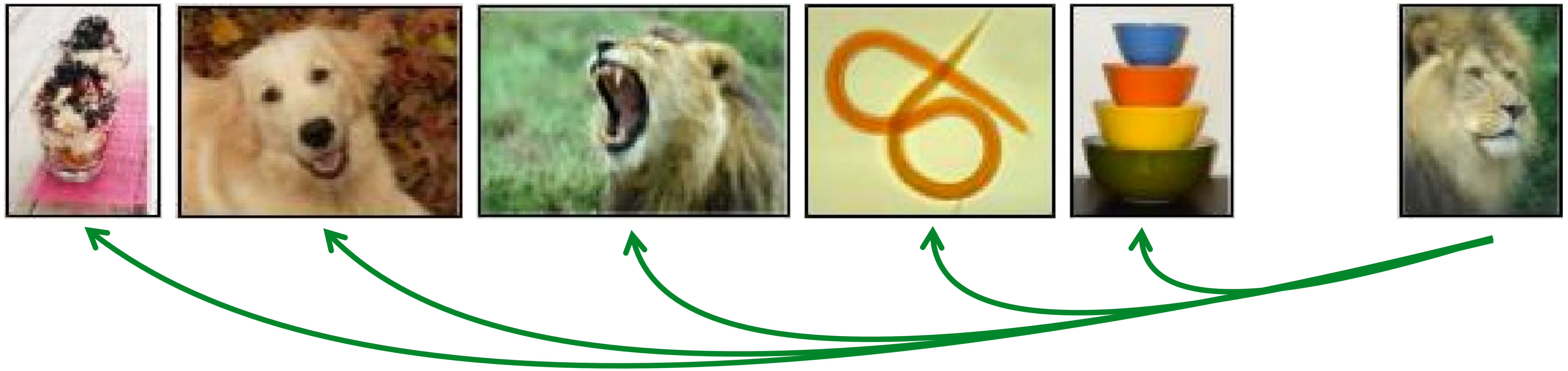
Outline

1. Applications of learning to learn
2. Problem formulation
3. **Solution Classes:**
   a) **metric-learning approach**
   b) direct black-box approach
   c) gradient-based approach
4. Open Questions / Problems

Metric Learning Approach

$\mathcal{D}_{\text{train}} \quad \mathbf{x}_{\text{test}} \longrightarrow \mathbf{y}_{\text{test}}$

training data $\mathcal{D}_{\text{train}}$ test set $\mathbf{x}_{\text{test}}$



Key idea: compare test image with training images

f := nearest neighbors

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

pixel space, l_2 distance?

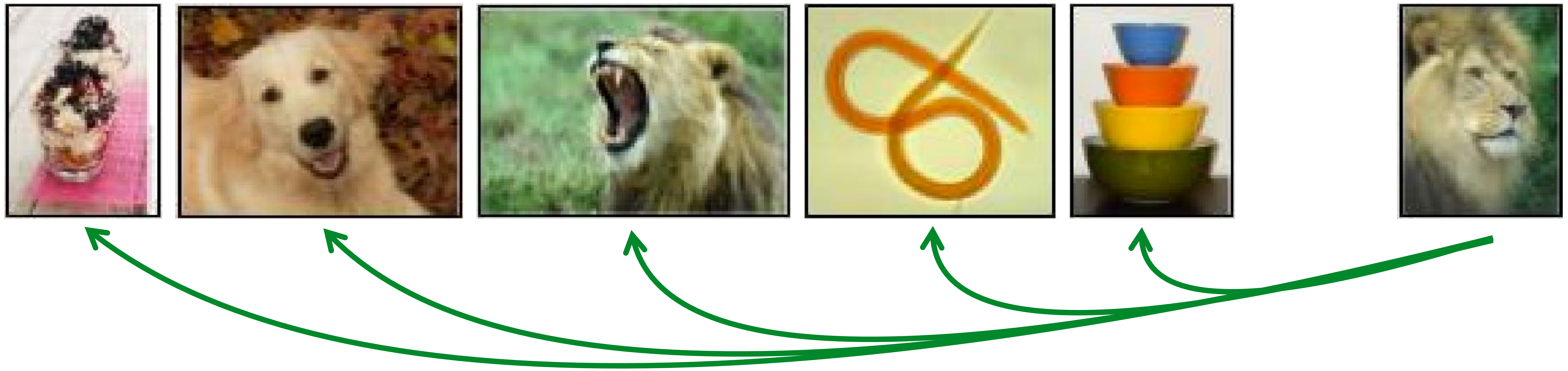
pixel space, l_2 distance?

Zhang et al. arXiv ‘18 (1801.03924)

Metric Learning Approach

$\mathcal{D}_{\text{train}} \quad \mathbf{x}_{\text{test}} \longrightarrow \mathbf{y}_{\text{test}}$

training data $\mathcal{D}_{\text{train}}$ test set $\mathbf{x}_{\text{test}}$



Key idea: compare test image with training images

f := nearest neighbors

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

<u>learn how to compare using data</u>

Metric Learning Example: Siamese Networks

Koch et al., ICML ‘15

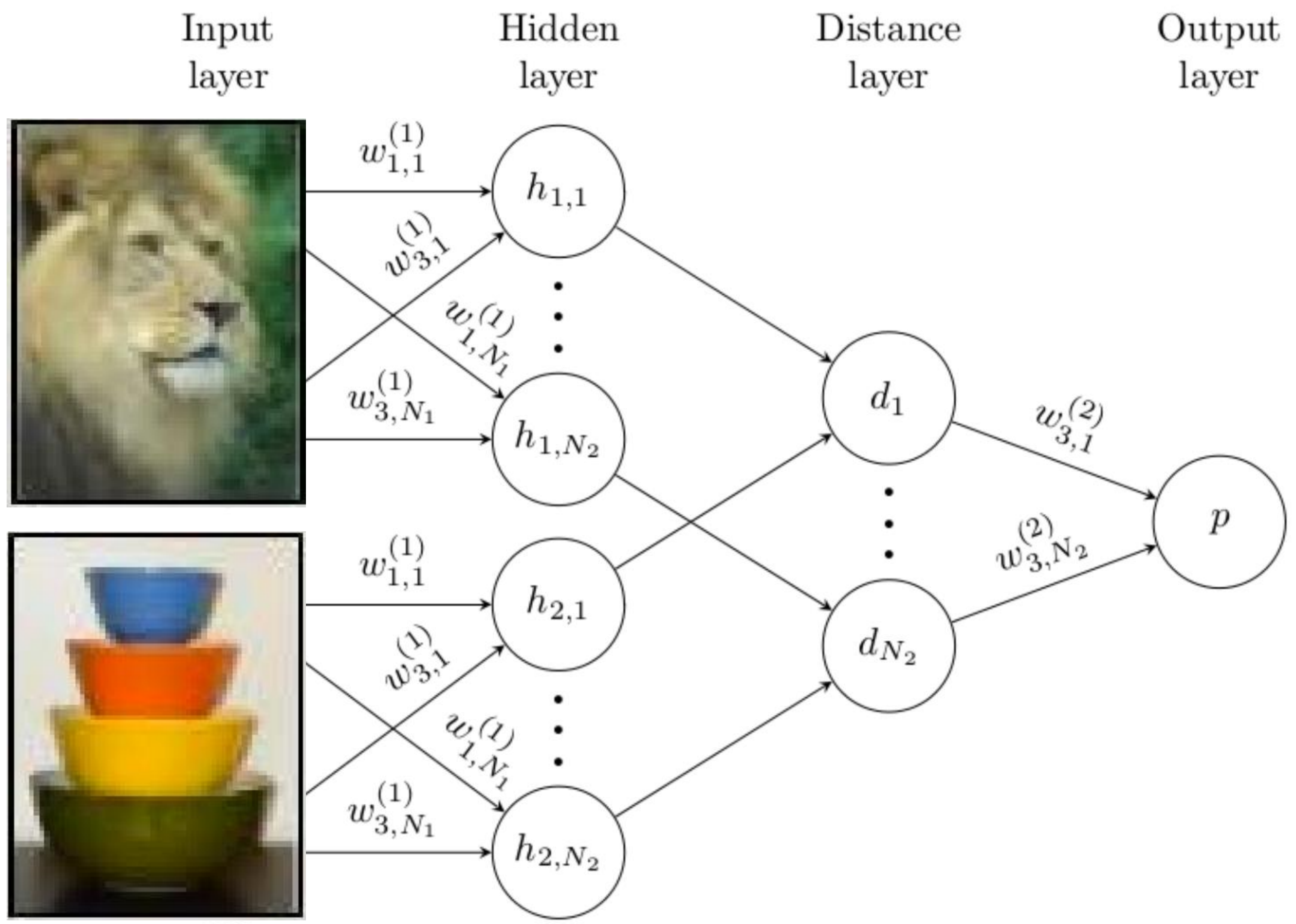
Siamese Neural Networks for One-shot Image Recognition

Gregory Koch GKOCH@CS.TORONTO.EDU
Richard Zemel ZEMEL@CS.TORONTO.EDU
Ruslan Salakhutdinov RSALAKHU@CS.TORONTO.EDU
Department of Computer Science, University of Toronto. Toronto, Ontario, Canada.

Siamese Networks for One-Shot Image Recognition

Koch et al., ICML '15

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



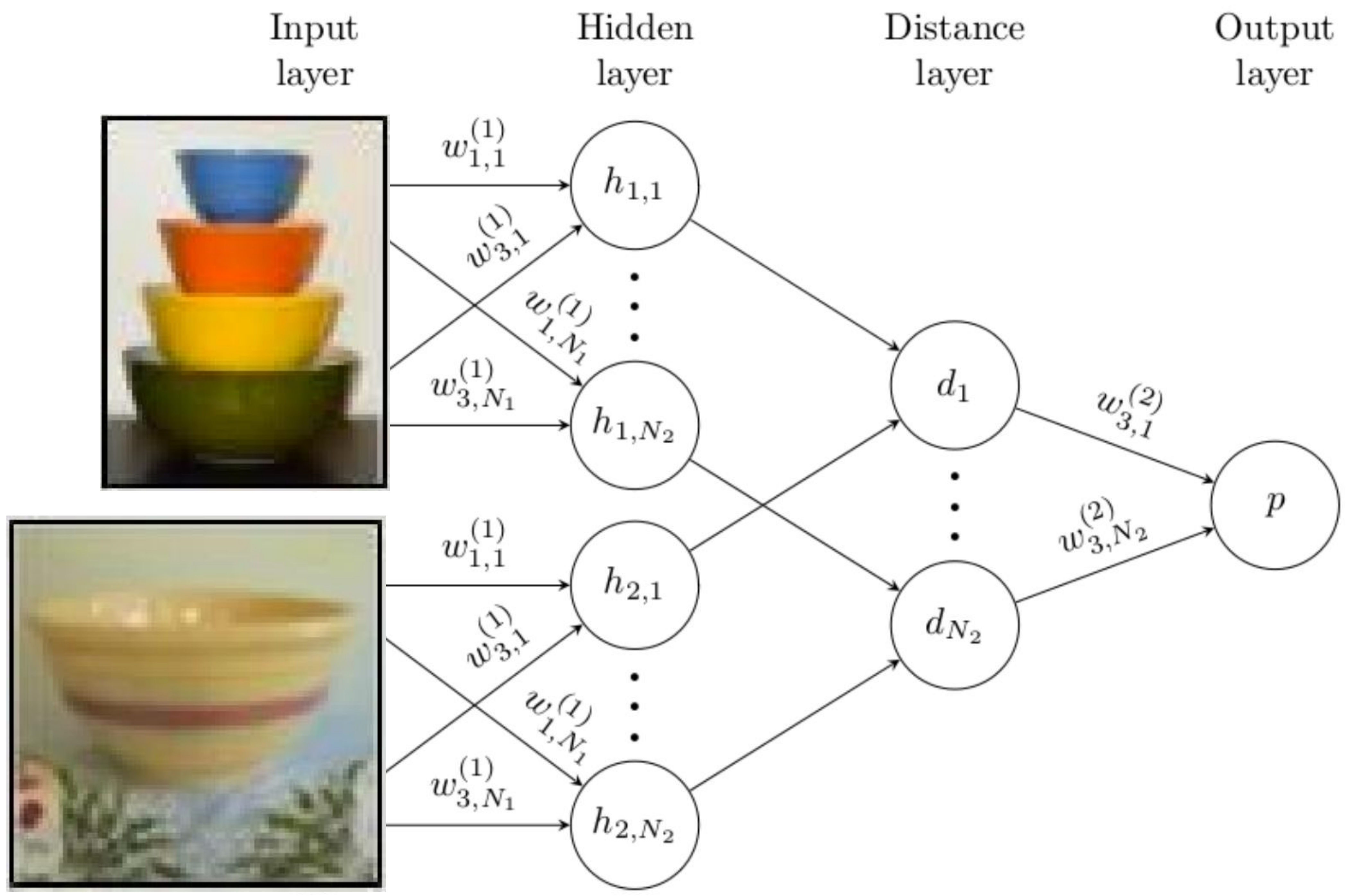
label

0

Siamese Networks for One-Shot Image Recognition

Koch et al., ICML ‘15

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



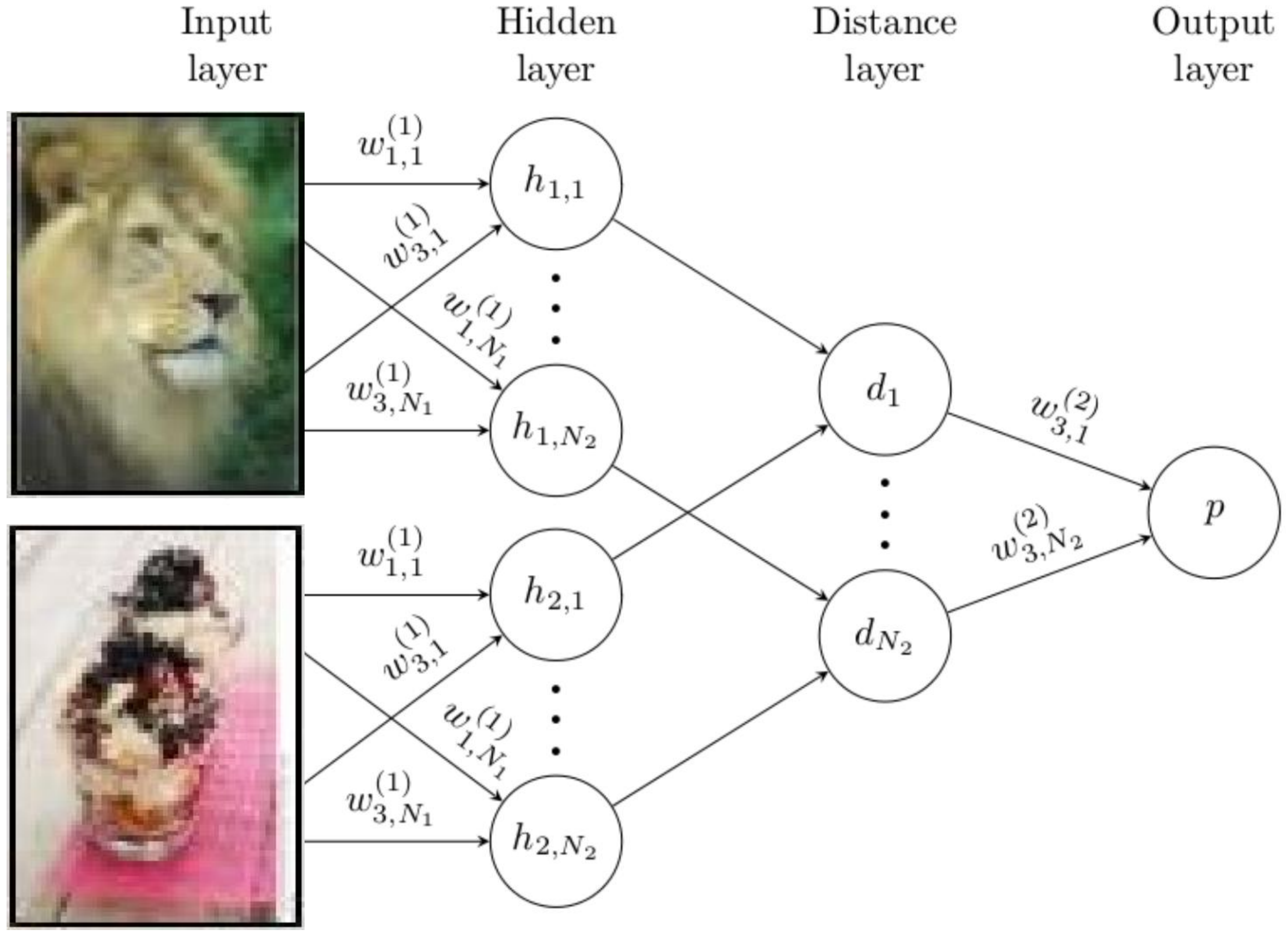
label

1

Siamese Networks for One-Shot Image Recognition

Koch et al., ICML '15

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



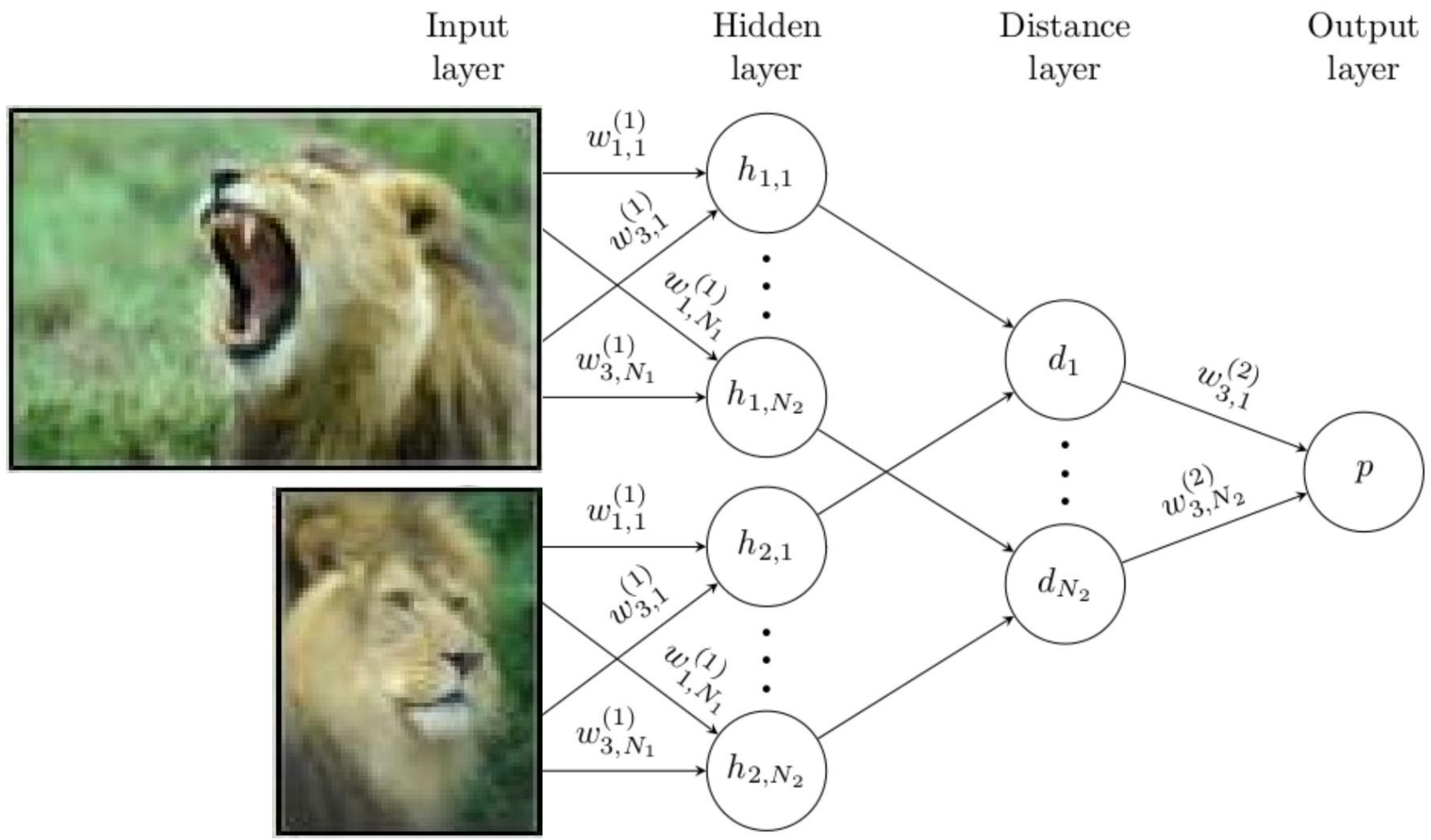
label

0

Siamese Networks for One-Shot Image Recognition

Koch et al., ICML '15

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



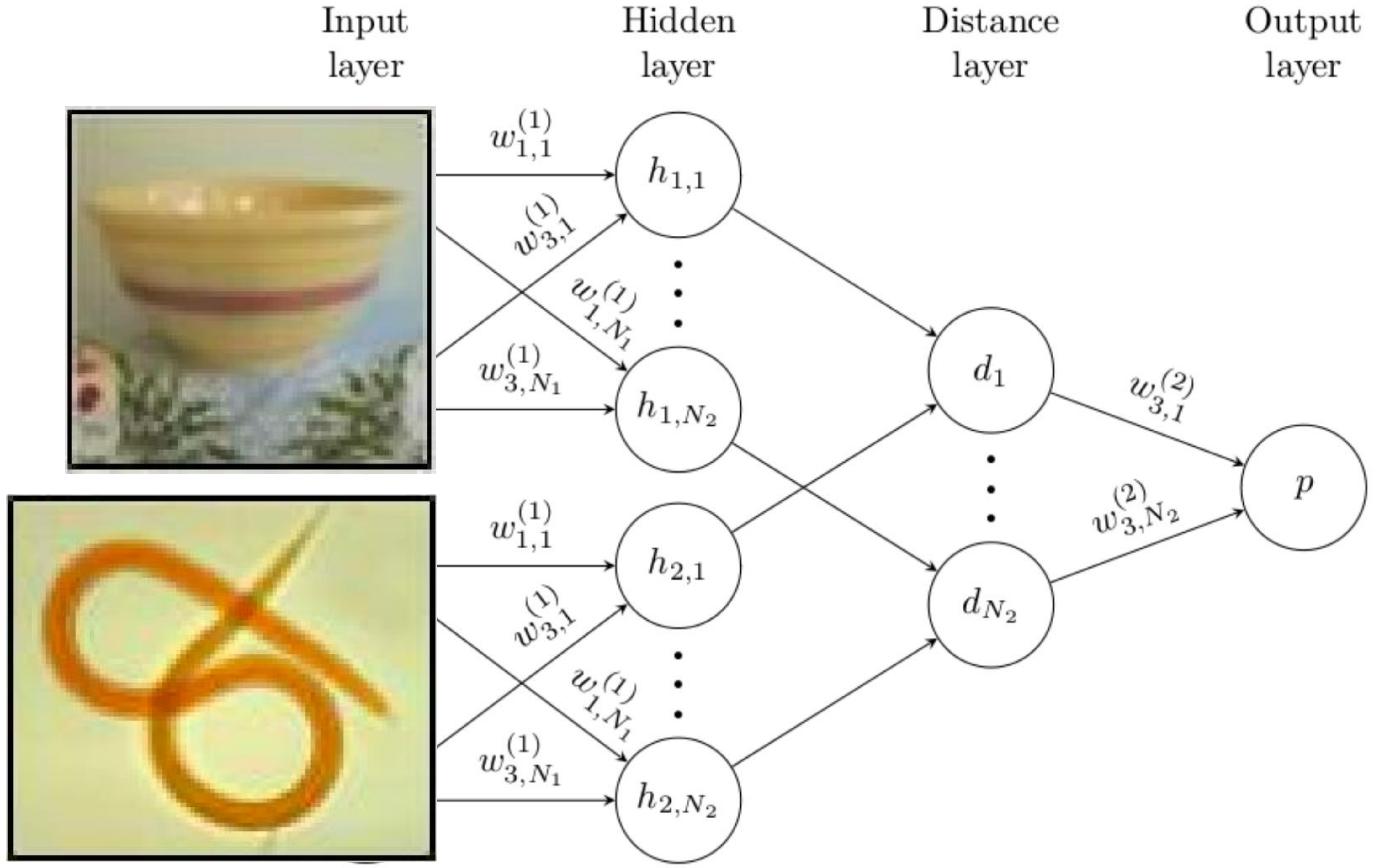
label

1

Siamese Networks for One-Shot Image Recognition

Koch et al., ICML '15

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



label

0

Test time: compare image $\mathbf{x}_{\text{test}}$ to each image in $\mathcal{D}_{\text{train}}$

Metric Learning Example: Matching Networks

Vinyals et al., NIPS '16

Matching Networks for One Shot Learning

Oriol Vinyals
Google DeepMind
vinyals@google.com

Charles Blundell
Google DeepMind
cblundell@google.com

Timothy Lillicrap
Google DeepMind
countzero@google.com

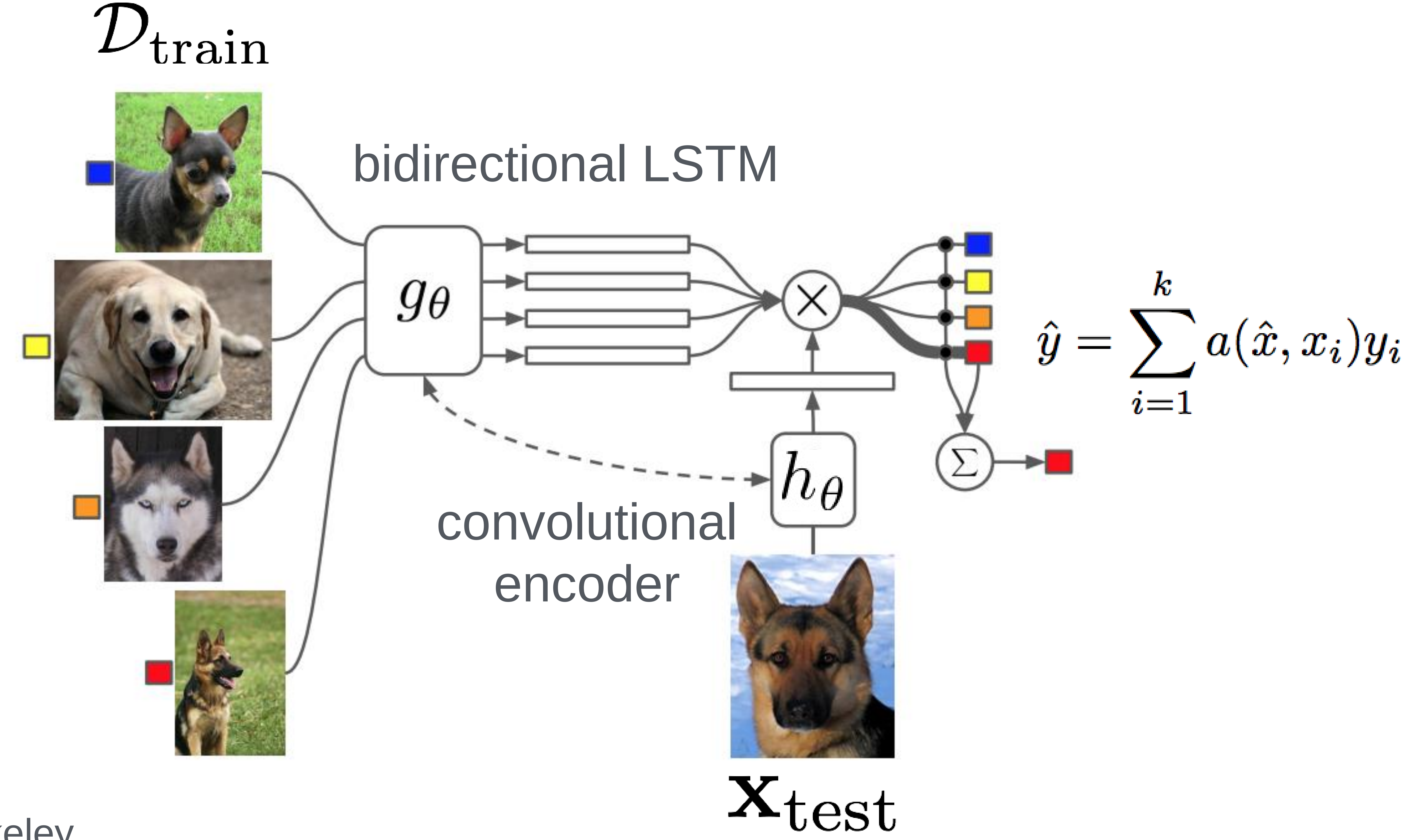
Koray Kavukcuoglu
Google DeepMind
korayk@google.com

Daan Wierstra
Google DeepMind
wierstra@google.com

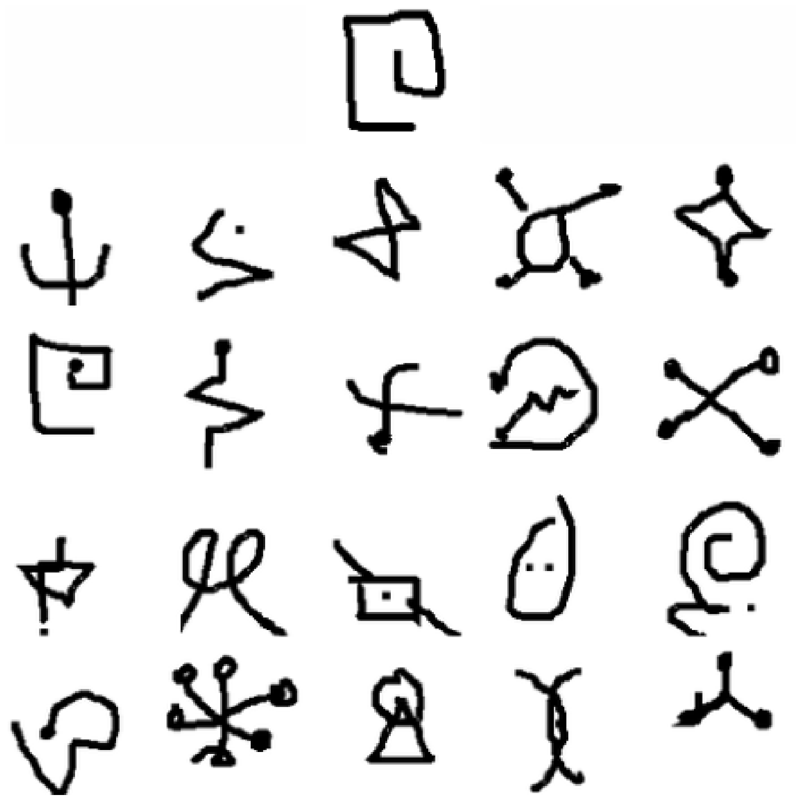
Matching Networks for One-Shot Learning

Vinyals et al., NIPS '16

motivating principle: train and test conditions must match



Few-Shot Classification Results



Omniglot dataset
Lake et al. '15

"MNIST transpose"

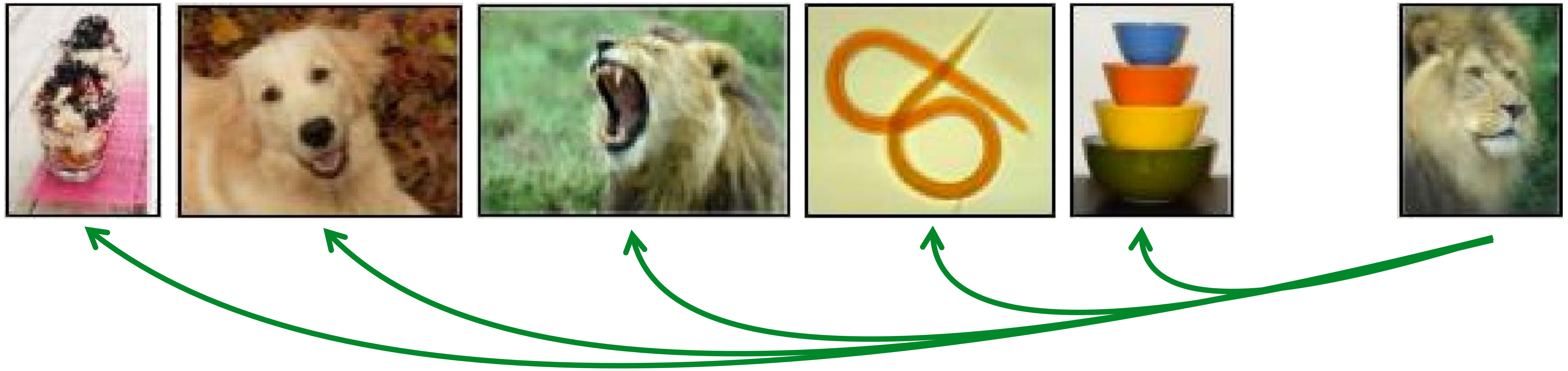
1623 character classes
each written by **20** different people

Model	Matching Fn	Fine Tune	5-way Acc		20-way Acc	
			1-shot	5-shot	1-shot	5-shot
PIXELS	Cosine	N	41.7%	63.2%	26.7%	42.6%
BASELINE CLASSIFIER	Cosine	N	80.0%	95.0%	69.5%	89.1%
BASELINE CLASSIFIER	Cosine	Y	82.3%	98.4%	70.6%	92.0%
BASELINE CLASSIFIER	Softmax	Y	86.0%	97.6%	72.9%	92.3%
MANN (NO CONV) [21]	Cosine	N	82.8%	94.9%	–	–
CONVOLUTIONAL SIAMESE NET [11]	Cosine	N	96.7%	98.4%	88.0%	96.5%
CONVOLUTIONAL SIAMESE NET [11]	Cosine	Y	97.3%	98.4%	88.1%	97.0%
MATCHING NETS (OURS)	Cosine	N	**98.1%**	**98.9%**	**93.8%**	98.5%
MATCHING NETS (OURS)	Cosine	Y	97.9%	98.7%	93.5%	**98.7%**

Metric Learning Approach

$\mathcal{D}_{\text{train}} \quad \mathbf{x}_{\text{test}} \longrightarrow \mathbf{y}_{\text{test}}$

training data $\mathcal{D}_{\text{train}}$ test set $\mathbf{x}_{\text{test}}$



Key idea: compare test image with training images

f := nearest neighbors

Takeaways:

+ Successful approach for few-shot classification
- Hasn't been demonstrated on non few-shot classification problems

Outline

1. Applications of learning to learn
2. Problem formulation
3. **Solution Classes:**
    a) metric-learning approach
    b) **direct black-box approach**
    c) gradient-based approach
4. Open Questions / Problems

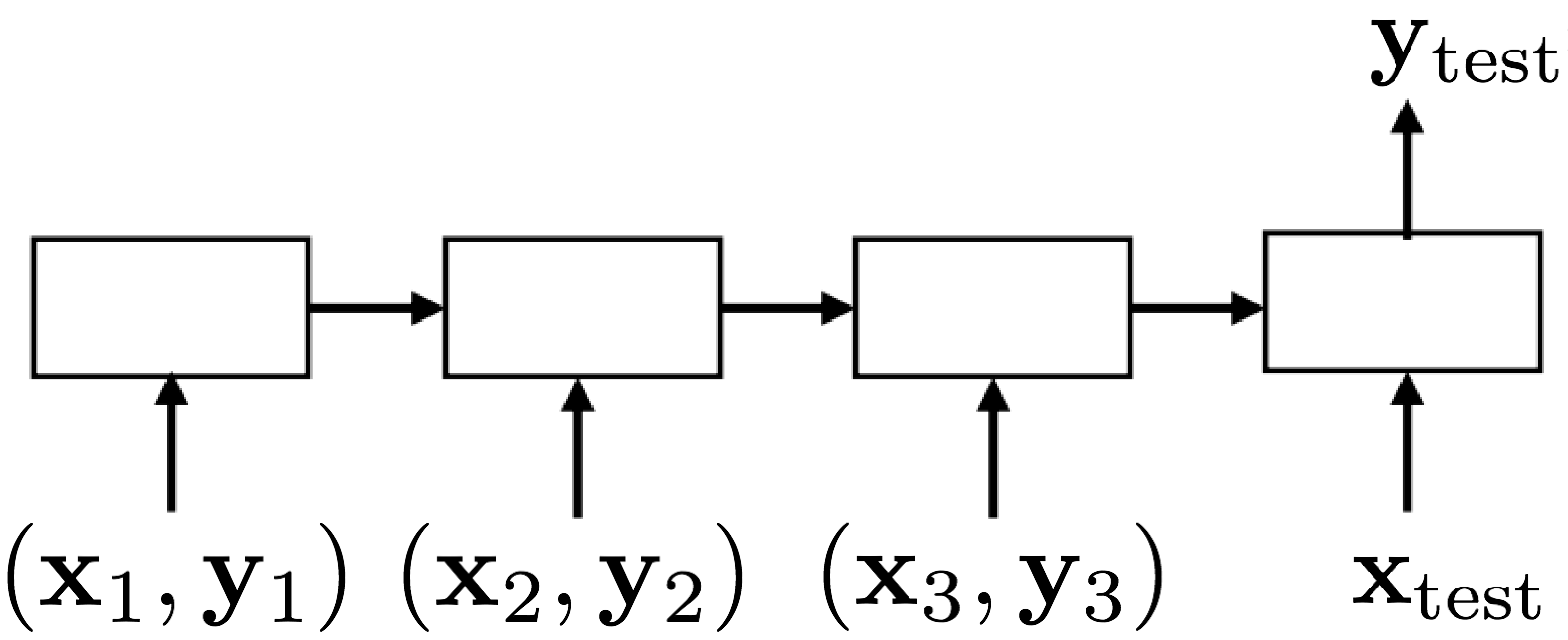
Design of *f* ?

$\mathcal{D}_{\text{train}}\ \mathbf{x}_{\text{test}} \longrightarrow \mathbf{y}_{\text{test}}$

Recurrent network
(LSTM, NTM, Conv)

$$\mathbf{y}_{\text{test}} = f(\mathcal{D}_{\text{train}}, \mathbf{x}_{\text{test}}; \theta)$$

Santoro et al. ’16, Duan et al. ’17, Wang et al. ’17,
Munkhdalai & Yu ’17, Mishra et al. ’17, …



Design of *f* ?

$\mathcal{D}_{\text{train}}\ \mathbf{x}_{\text{test}} \longrightarrow \mathbf{y}_{\text{test}}$

Recurrent network
(LSTM, NTM, Conv)

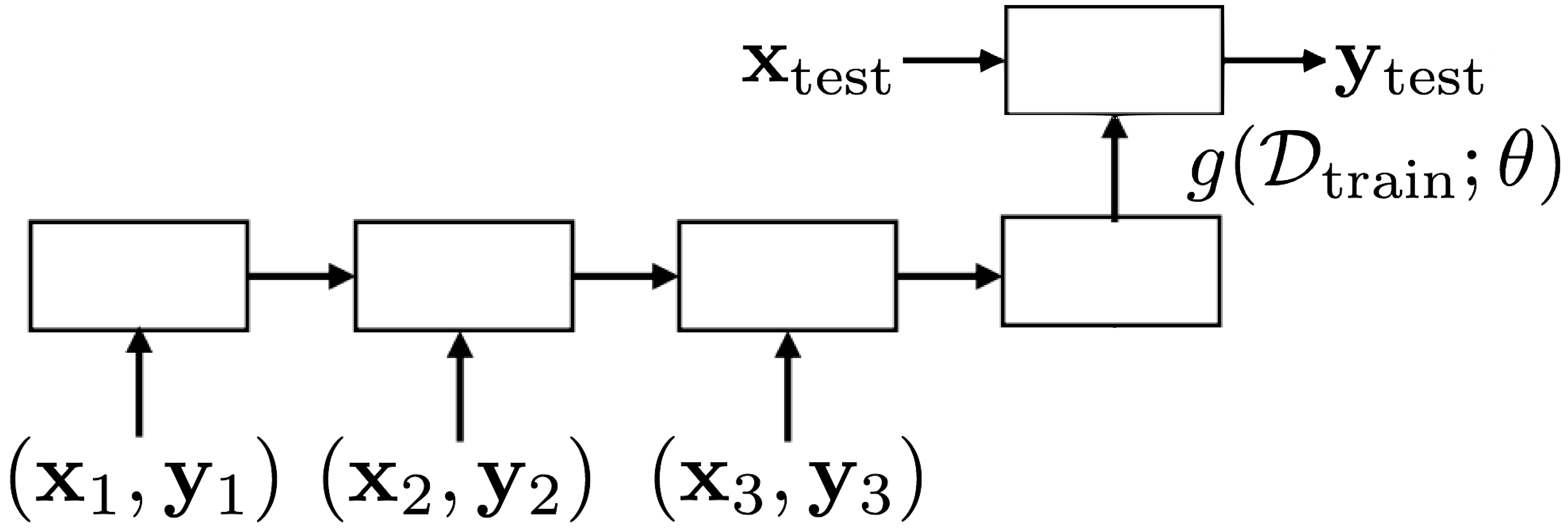
$$\mathbf{y}_{\text{test}} = f(\mathcal{D}_{\text{train}}, \mathbf{x}_{\text{test}}; \theta)$$

Santoro et al. '16, Duan et al. '17, Wang et al. '17,
Munkhdalai & Yu '17, Mishra et al. '17, …

Learned optimizer
(often uses recurrence)

$$\mathbf{y}_{\text{test}} = f(\mathbf{x}_{\text{test}}; g(\mathcal{D}_{\text{train}}; \theta))$$

Schmidhuber et al. '87, Bengio et al. '90, Hochreiter et al. '01, Li & Malik '16, Andrychowicz et al. '16, Ha et al. '17, Ravi & Larochelle '17, …



Case Study: Simple Neural Attentive Meta-Learner

Mishra et al, ICLR '18

A SIMPLE NEURAL ATTENTIVE META-LEARNER

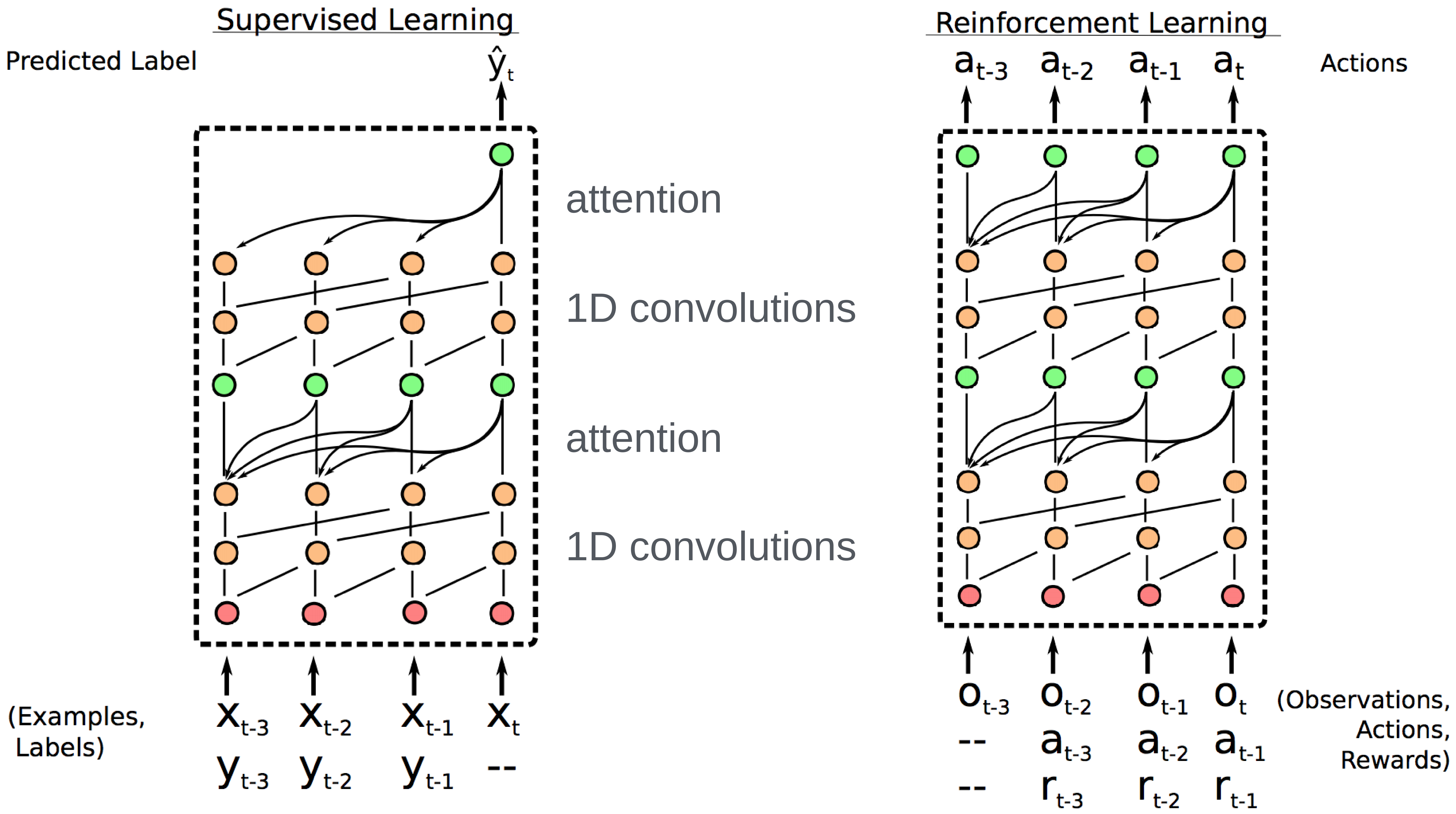
Nikhil Mishra [*,†] **Mostafa Rohaninejad**[*] **Xi Chen**[†] **Pieter Abbeel**[†]

UC Berkeley, Department of Electrical Engineering and Computer Science
Embodied Intelligence
{nmishra, rohaninejadm, c.xi, pabbeel}@berkeley.edu

Simple Neural Attentive Meta-Learner

Mishra et al, ICLR '18

interleave 1D convolutions and attention



Simple Neural Attentive Meta-Learner

Mishra et al, ICLR ‘18

Experiment: Learning to visually navigate a maze

- train on 1000 small mazes
- test on held-out small mazes and large mazes

Simple Neural Attentive Meta-Learner

Mishra et al, ICLR '18

Experiment: Learning to visually navigate a maze

- train on 1000 small mazes
- test on held-out small mazes and large mazes

Method	Small Maze		Large Maze	
	Episode 1	Episode 2	Episode 1	Episode 2
Random	188.6 ± 3.5	187.7 ± 3.5	420.2 ± 1.2	420.8 ± 1.2
LSTM	52.4 ± 1.3	39.1 ± 0.9	180.1 ± 6.0	150.6 ± 5.9
SNAIL (ours)	$\mathbf{50.3 \pm 0.3}$	$\mathbf{34.8 \pm 0.2}$	$\mathbf{140.5 \pm 4.2}$	$\mathbf{105.9 \pm 2.4}$

Table 5: Average time to find the goal on each episode

Design of *f* ?

$$\mathcal{D}_{\text{train}} \; \mathbf{x}_{\text{test}} \longrightarrow \mathbf{y}_{\text{test}}$$

Recurrent network
(LSTM, NTM, Conv)

$$\mathbf{y}_{\text{test}} = f(\mathcal{D}_{\text{train}}, \mathbf{x}_{\text{test}}; \theta)$$

Santoro et al. '16, Duan et al. '17, Wang et al. '17,
Munkhdalai & Yu '17, Mishra et al. '17, …

Learned optimizer
(often uses recurrence)

$$\mathbf{y}_{\text{test}} = f(\mathbf{x}_{\text{test}}; g(\mathcal{D}_{\text{train}}; \theta))$$

Schmidhuber et al. '87, Bengio et al. '90, Hochreiter et al. '01, Li & Malik '16, Andrychowicz et al. '16, Ha et al. '17, Ravi & Larochelle '17, …

Takeaways:

\+ General and powerful approach to meta-learning, has been demonstrated for a range of meta-learning problems

\- complex model and complex task (very little structure)

\- typically data inefficient

Outline

1. Applications of learning to learn
2. Problem formulation
3. **Solution Classes:**
   a) metric-learning approach
   b) direct black-box approach
   c) **gradient-based approach**
4. Open Questions / Problems

Learning Few-Shot Adaptation

pretrained parameters

Fine-tuning $\theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}_{\text{train}}(\theta)$

[test-time]

training data for new task

Our method $\min_\theta \sum_{\text{task } i} \mathcal{L}^i_{\text{test}}(\theta - \alpha \nabla_\theta \mathcal{L}^i_{\text{train}}(\theta))$

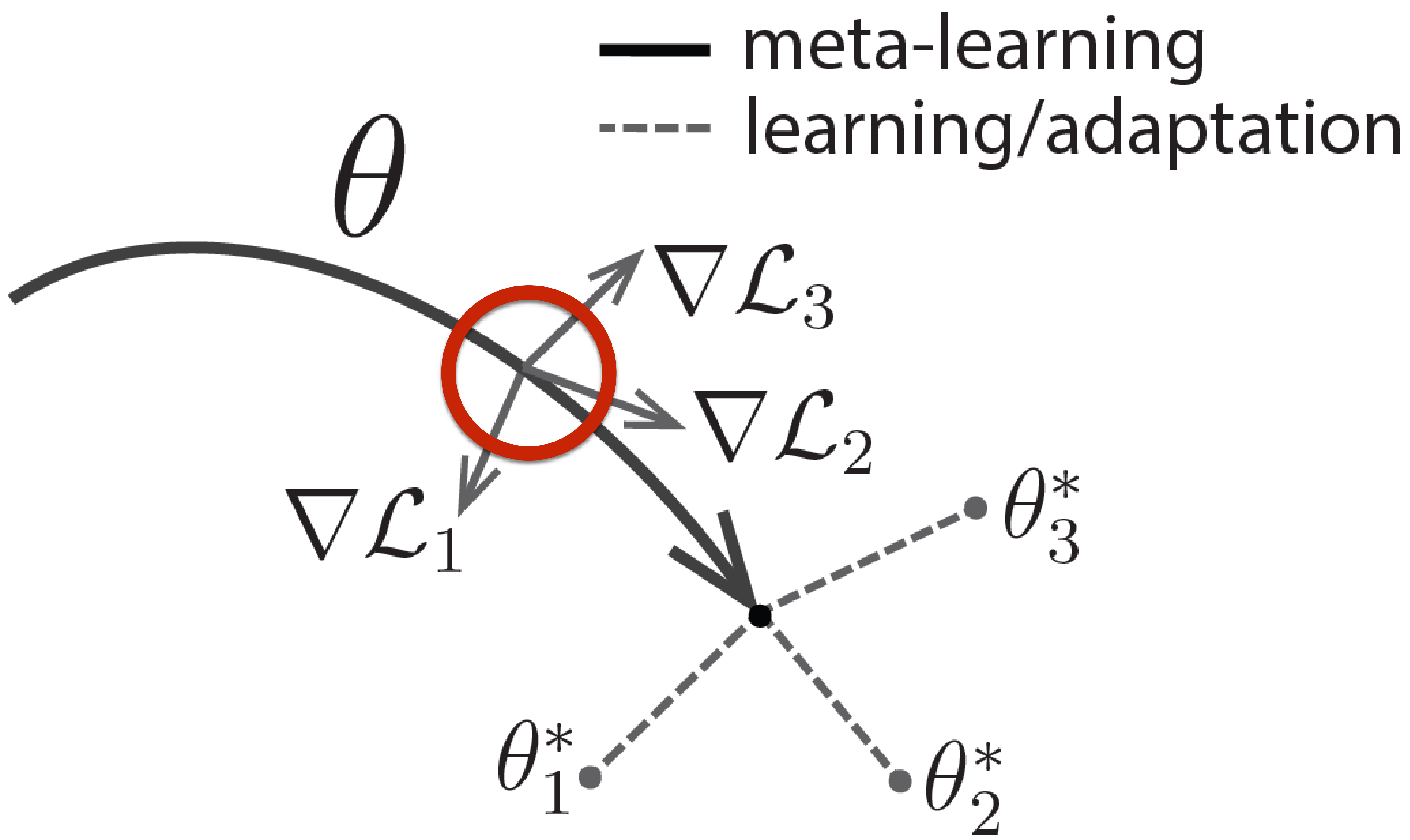
Key idea: Train over many tasks, to learn parameter vector θ that transfers

Learning Few-Shot Adaptation

$$\min_{\theta} \sum_{\text{task } i} \mathcal{L}^i_{\text{test}}(\theta - \alpha \nabla_\theta \mathcal{L}^i_{\text{train}}(\theta))$$

θ parameter vector being meta-learned

θ^*_i optimal parameter vector for task i



Model-**A**gnostic **M**eta-**L**earning

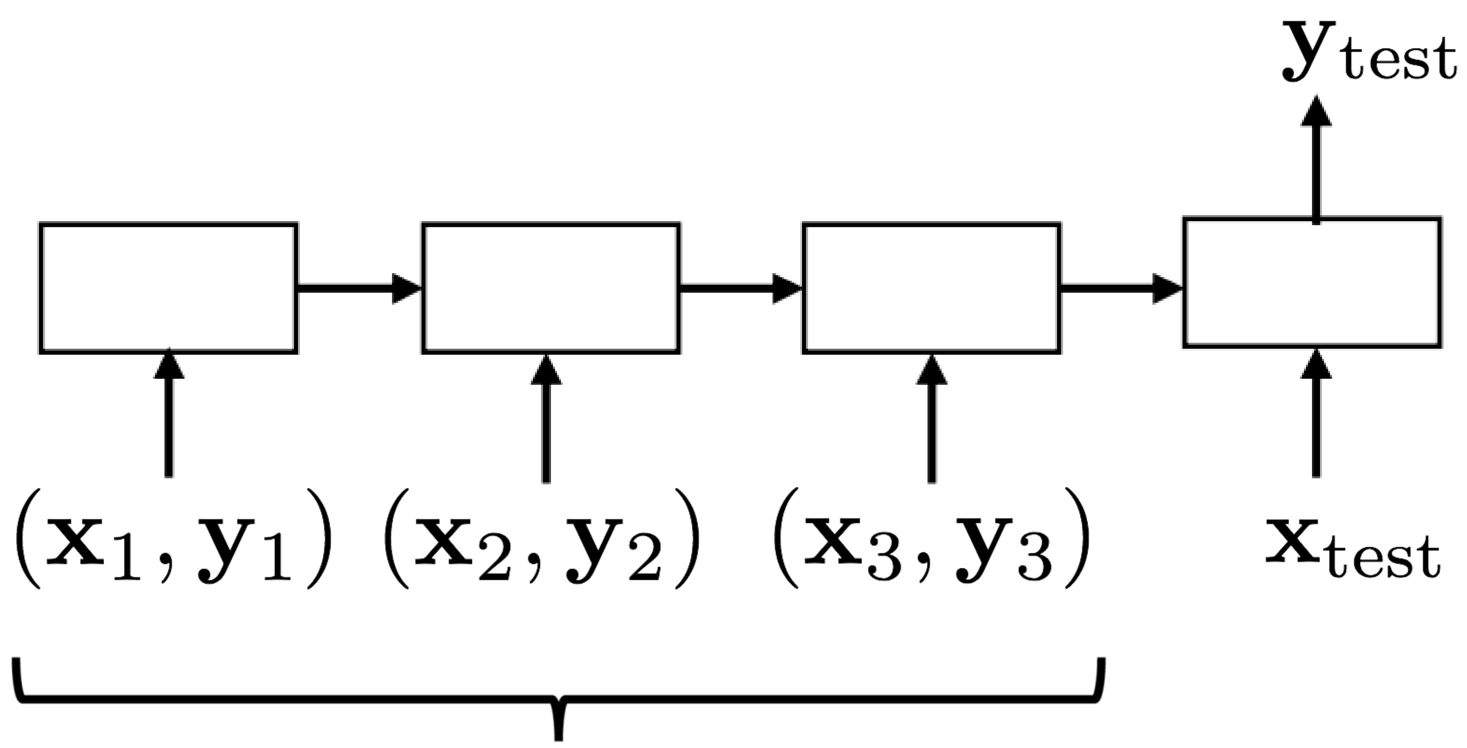
Finn, Abbeel, Levine ICML '17

Design of f ?

$$\mathcal{D}_{\text{train}} \quad \mathbf{x}_{\text{test}} \longrightarrow \mathbf{y}_{\text{test}}$$

Recurrent network

$$\mathbf{y}_{\text{test}} = f(\mathcal{D}_{\text{train}}, \mathbf{x}_{\text{test}}; \theta)$$



network implements the
"learned learning procedure"

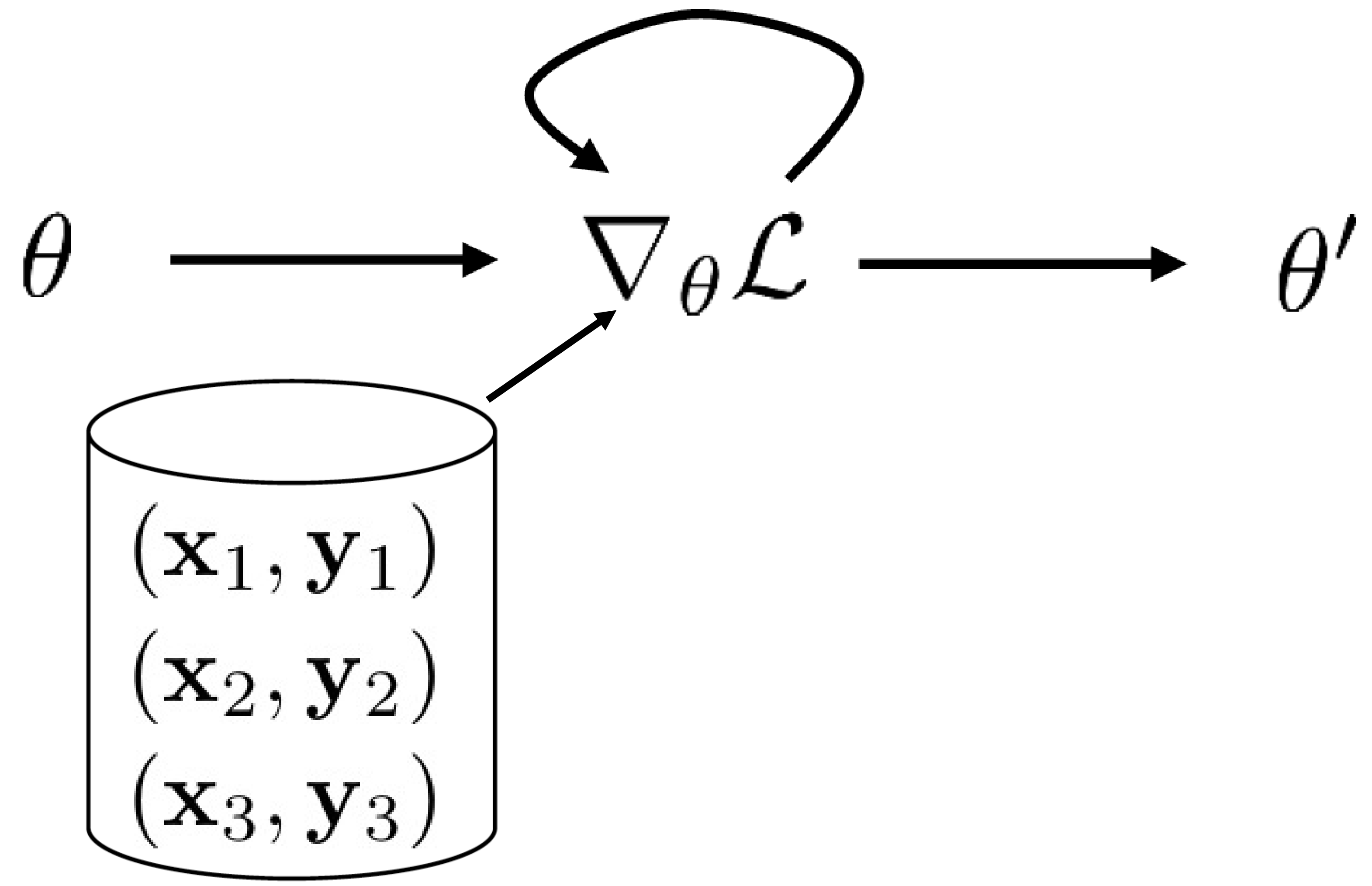
Does it converge?
- Sort of?

What does it converge to?
- Who knows…

What to do if not good enough?
- Nothing

MAML

$$\mathbf{y}_{\text{test}} = f(\mathbf{x}_{\text{test}}; \theta - \alpha \nabla_\theta \mathcal{L}(\mathcal{D}_{\text{train}}))$$



Does it converge?
- Yes (it's gradient descent…)

What does it converge to?
- A local optimum (it's gradient descent…)

What to do if not good enough?
- Keep taking gradient steps (it's gradient descent..)

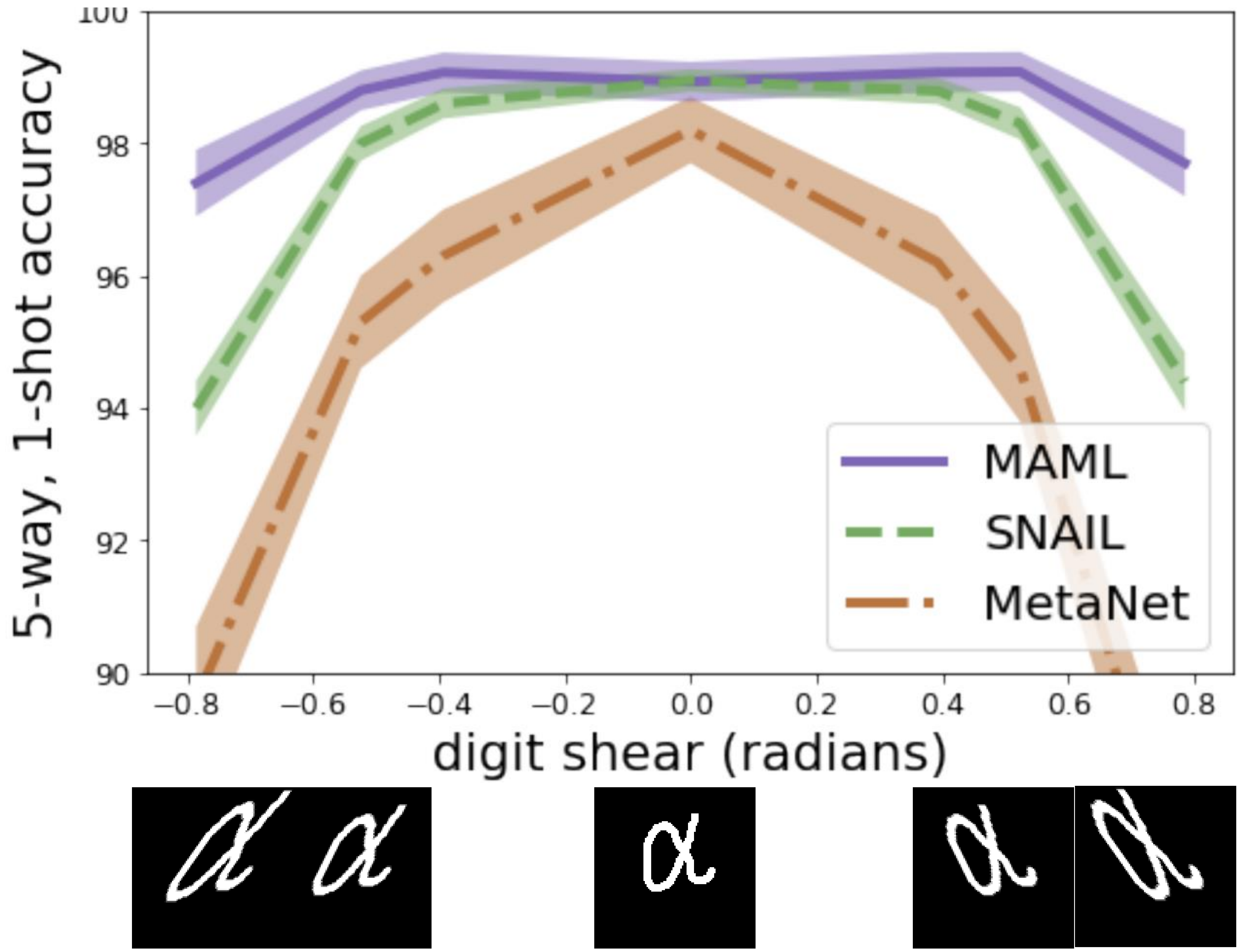
How well can methods generalize to similar, but extrapolated tasks?

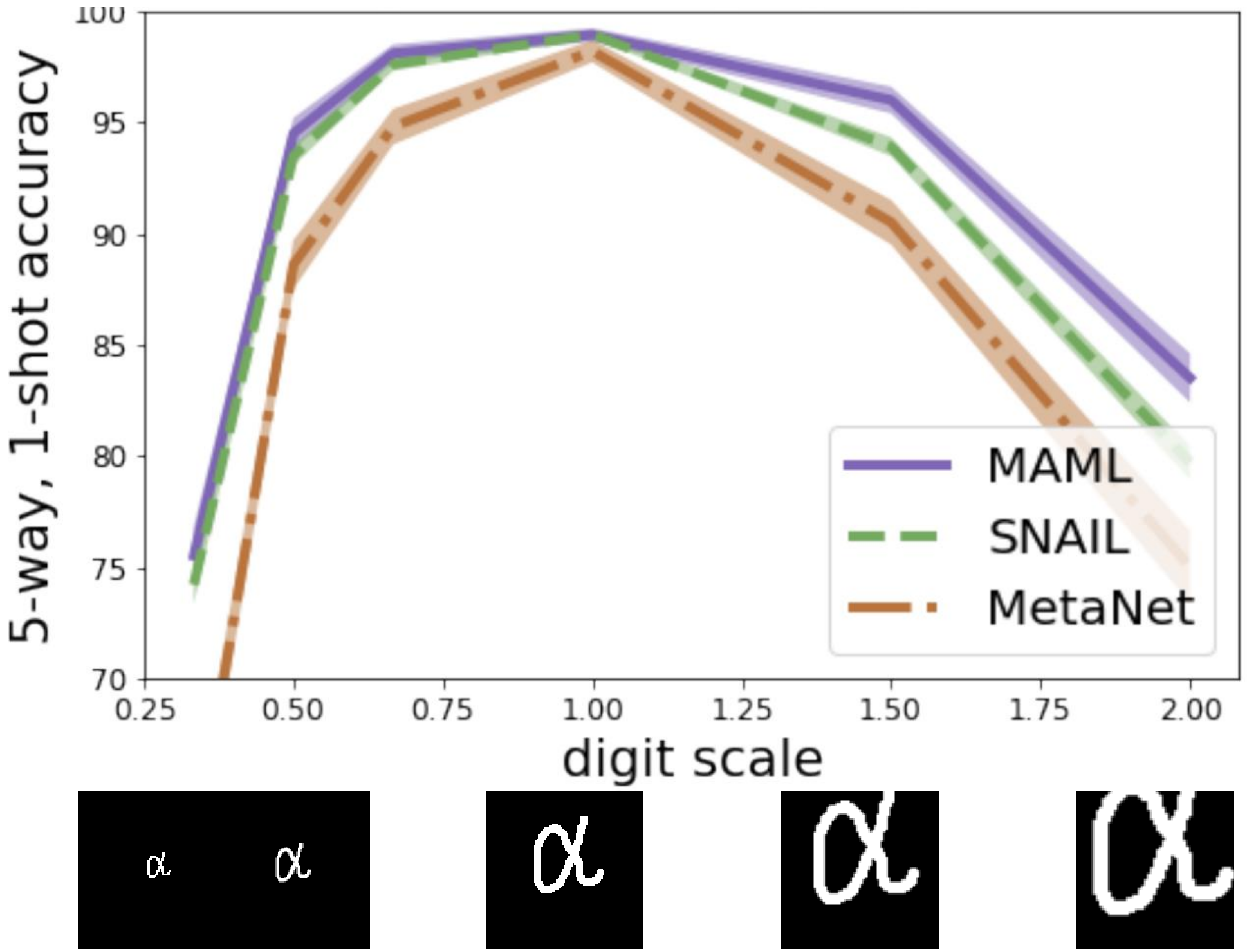
The world is non-stationary.

MAML SNAIL, MetaNetworks

Omniglot image classification

performance





task variability

Finn & Levine ICLR '18

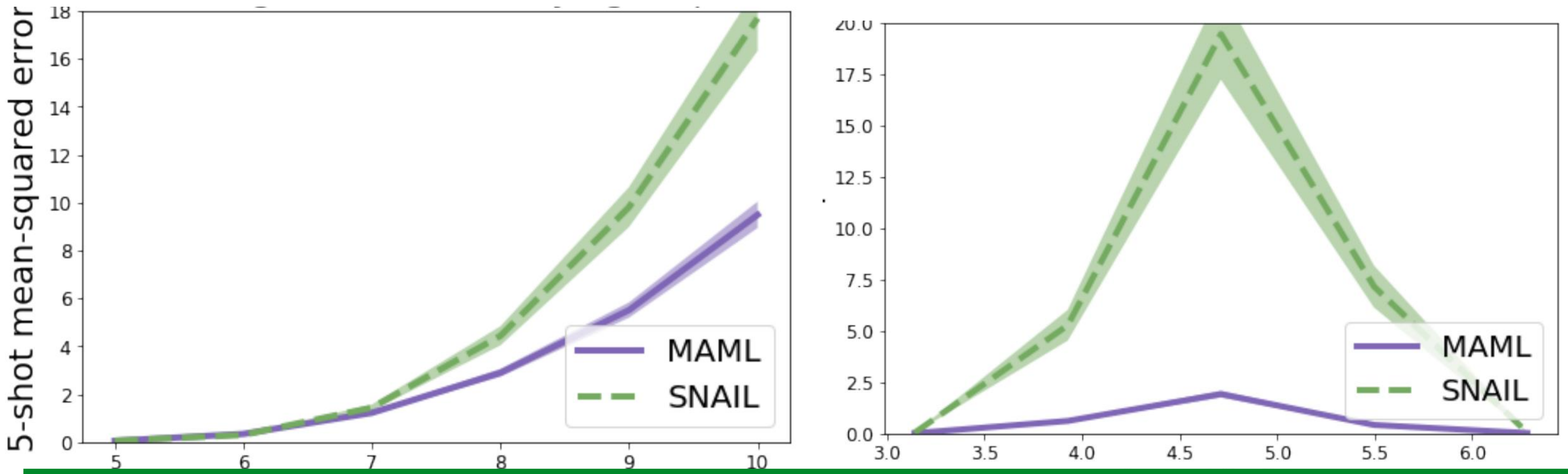
How well can methods generalize to similar, but extrapolated tasks?

The world is non-stationary.

MAML SNAIL

Sinusoid curve regression

error



Takeaway: Strategies learned with MAML consistently generalize better to out-of-distribution tasks

Does this structure come at a cost?

Finn & Levine ICLR ’18

Recurrent network

$$\mathbf{y}_{\text{test}} = f(\mathcal{D}_{\text{train}}, \mathbf{x}_{\text{test}}; \theta)$$

MAML

$$\mathbf{y}_{\text{test}} = f(\mathbf{x}_{\text{test}}; \theta - \alpha \nabla_\theta \mathcal{L}(\mathcal{D}_{\text{train}}))$$

Does this structure come at a cost?

For a sufficiently deep *f*,
MAML function can approximate any function of $\mathcal{D}_{\text{train}}, \mathbf{x}_{\text{test}}$

Finn & Levine, ICLR 2018

Assumptions:

- nonzero α
- loss function gradient does not lose information about the label
- datapoints in $\mathcal{D}_{\text{train}}$ are unique

Why is this interesting?
MAML has benefit of inductive bias without losing expressive power.

Application: One-Shot Visual Imitation Learning

One-Shot Visual Imitation Learning via Meta-Learning

Chelsea Finn*[1], Tianhe Yu*[1], Tianhao Zhang[1], Pieter Abbeel[1,2], Sergey Levine[1]
[1]University of California, Berkeley, [2]OpenAI
cbfinn,tianhe.yu,tianhao.z,pabbeel,svlevine@berkeley.edu

One-Shot Imitation from Observing Humans via Domain-Adaptive Meta-Learning

Tianhe Yu*, Chelsea Finn*, Annie Xie, Sudeep Dasari, Tianhao Zhang, Pieter Abbeel, Sergey Levine
University of California, Berkeley
Email: {tianhe.yu,cbfinn,anniexie,sdasari,tianhao.z,pabbeel,svlevine}@berkeley.edu
* denotes equal contribution

One-Shot Visual Imitation Learning

Goal: Given one visual demonstration of a new task, learn a policy

Visual imitation is expensive.

Rahmanizadeh et al. ‘17 Zhang et al. ‘17

learns from raw pixels,

but requires many demonstrations

Through meta-learning: reuse data from other tasks/objects/environments

Finn*, Yu*, Zhang, Abbeel, Levine CoRL ‘17

One-Shot Visual Imitation Learning

imitation loss

$$\mathcal{L} = \sum_t \|\pi_\theta(o_t) - a_t^*\|^2$$

meta-training time $\min_\theta \sum_{\text{task } i} \mathcal{L}^i_{\text{test}}(\theta - \alpha \nabla_\theta \mathcal{L}^i_{\text{train}}(\theta))$

meta-training tasks

test demo

training demo

meta-test time $\theta' \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}(\theta)$

demo of meta-test task

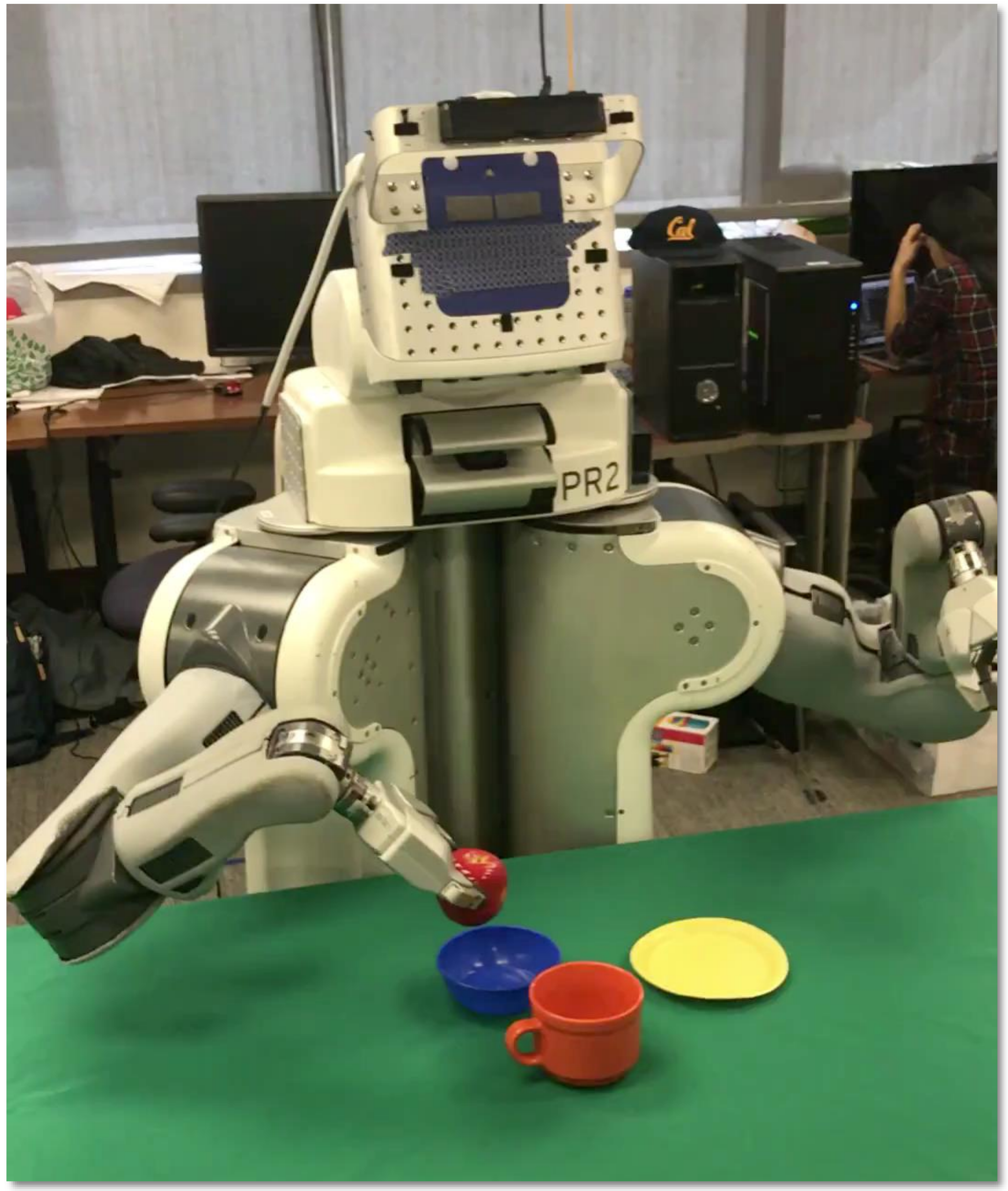
Finn*, Yu*, Zhang, Abbeel, Levine CoRL '17

Object placing from pixels

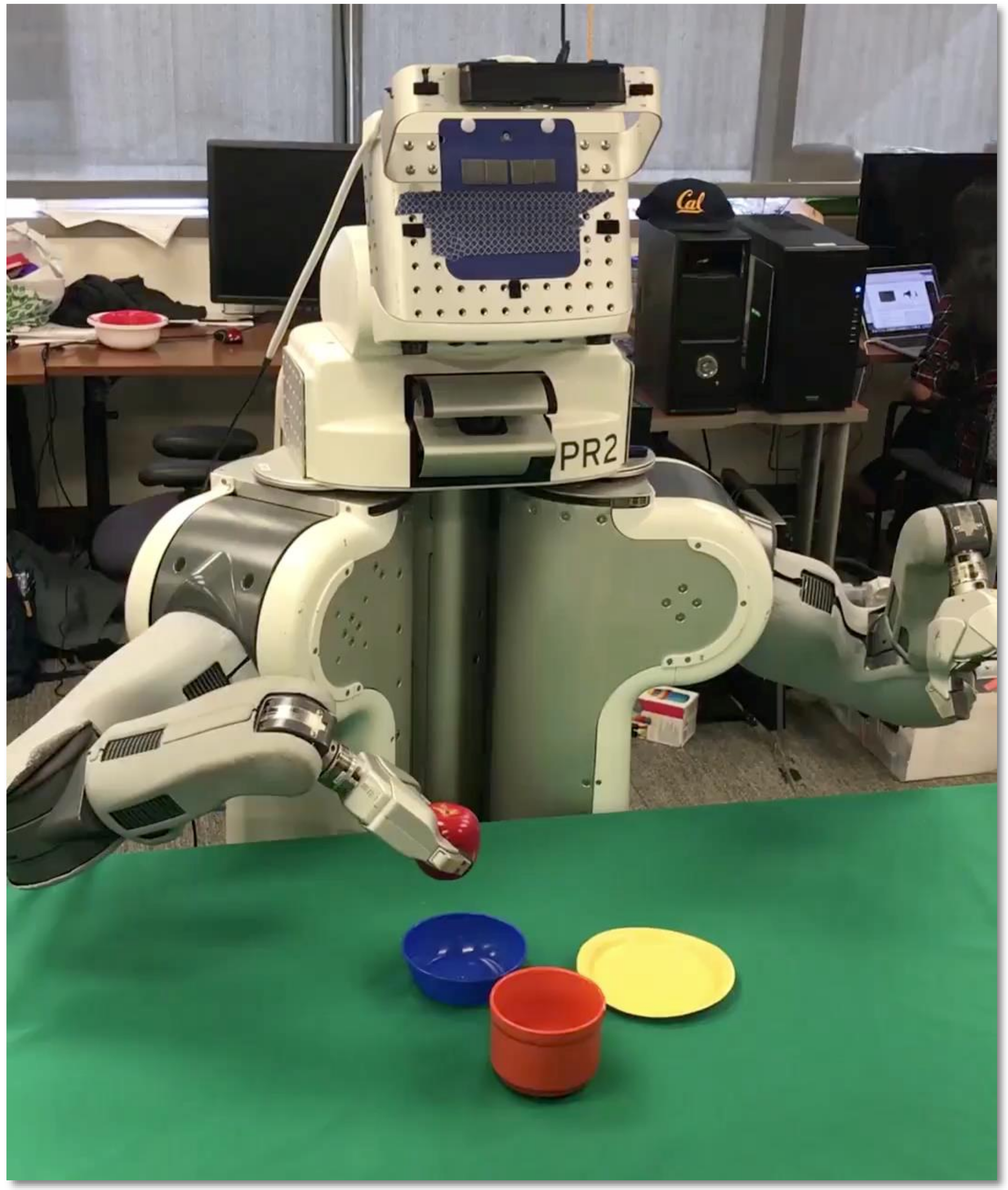
subset of
training objects

held-out test objects

input demo
(via teleoperation)



resulting policy



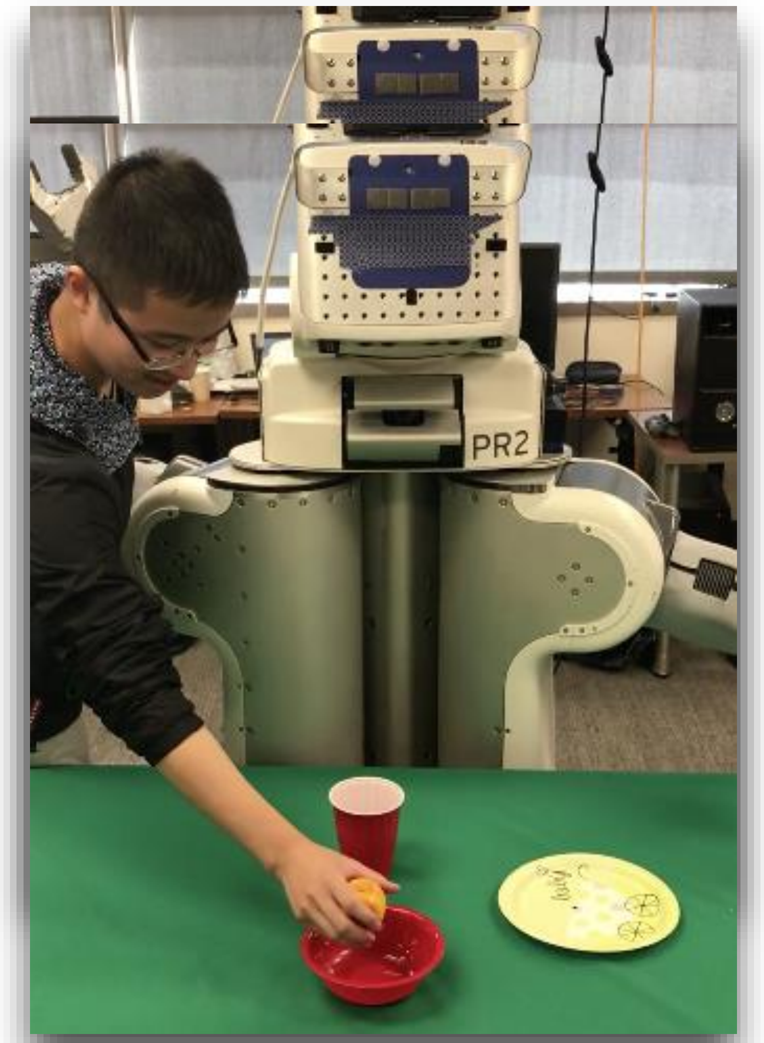
[real-time execution]

Finn*, Yu*, Zhang, Abbeel, Levine CoRL '17

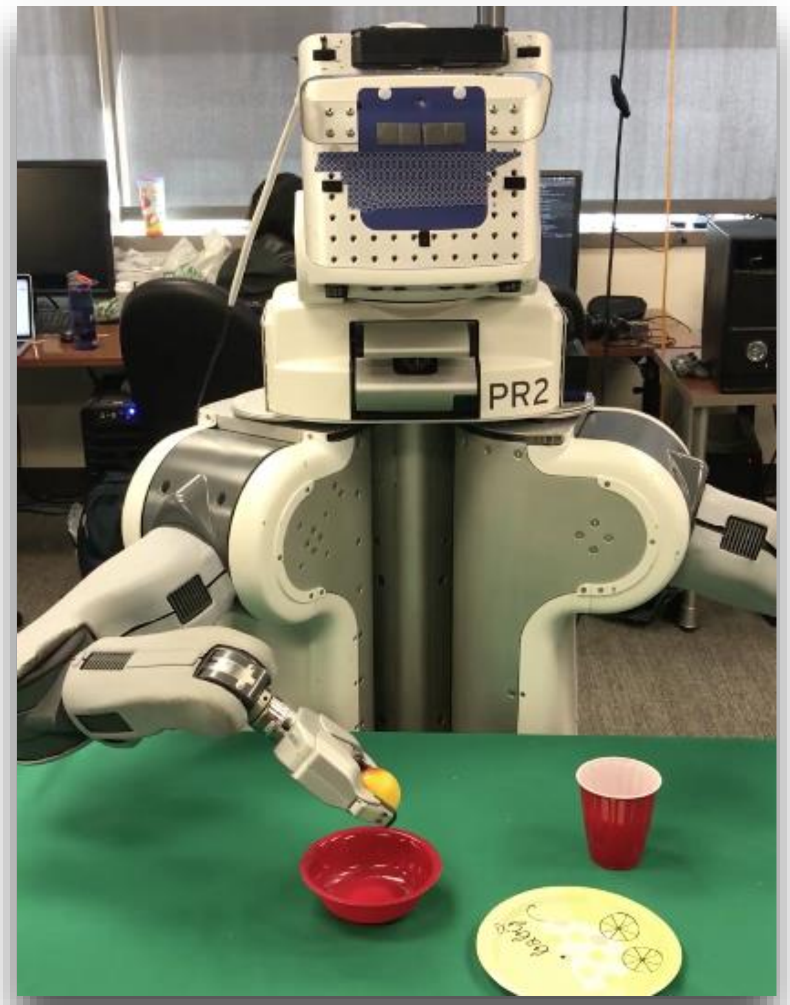
Few-Shot Imitation Learning *from Weak Supervision*

~~Given one teleoperated demonstration:~~

Given a video of a human:



Learn a policy.



Yu*, Finn*, Xie, Dasari, Zhang, Abbeel, Levine RSS ’18

Learning to Learn from Weak Supervision

meta-training

$$\min_{\theta} \sum_{\text{task } i} \mathcal{L}^i_{\text{test}}(\theta - \alpha \nabla_\theta \mathcal{L}^i_{\text{train}}(\theta))$$

fully supervised **weakly supervised**

meta-test

$$\theta' \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}(\theta)$$

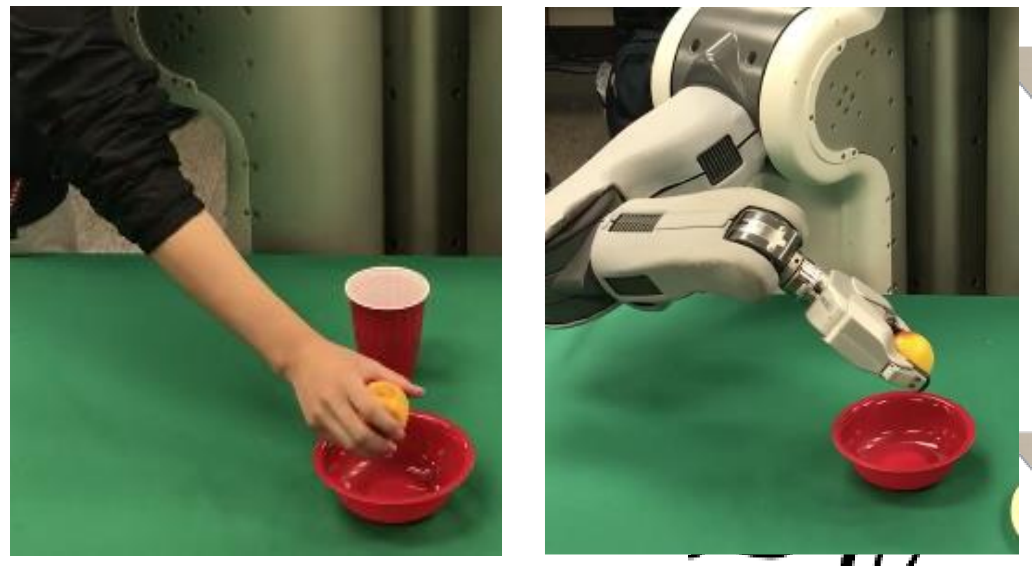
weakly supervised

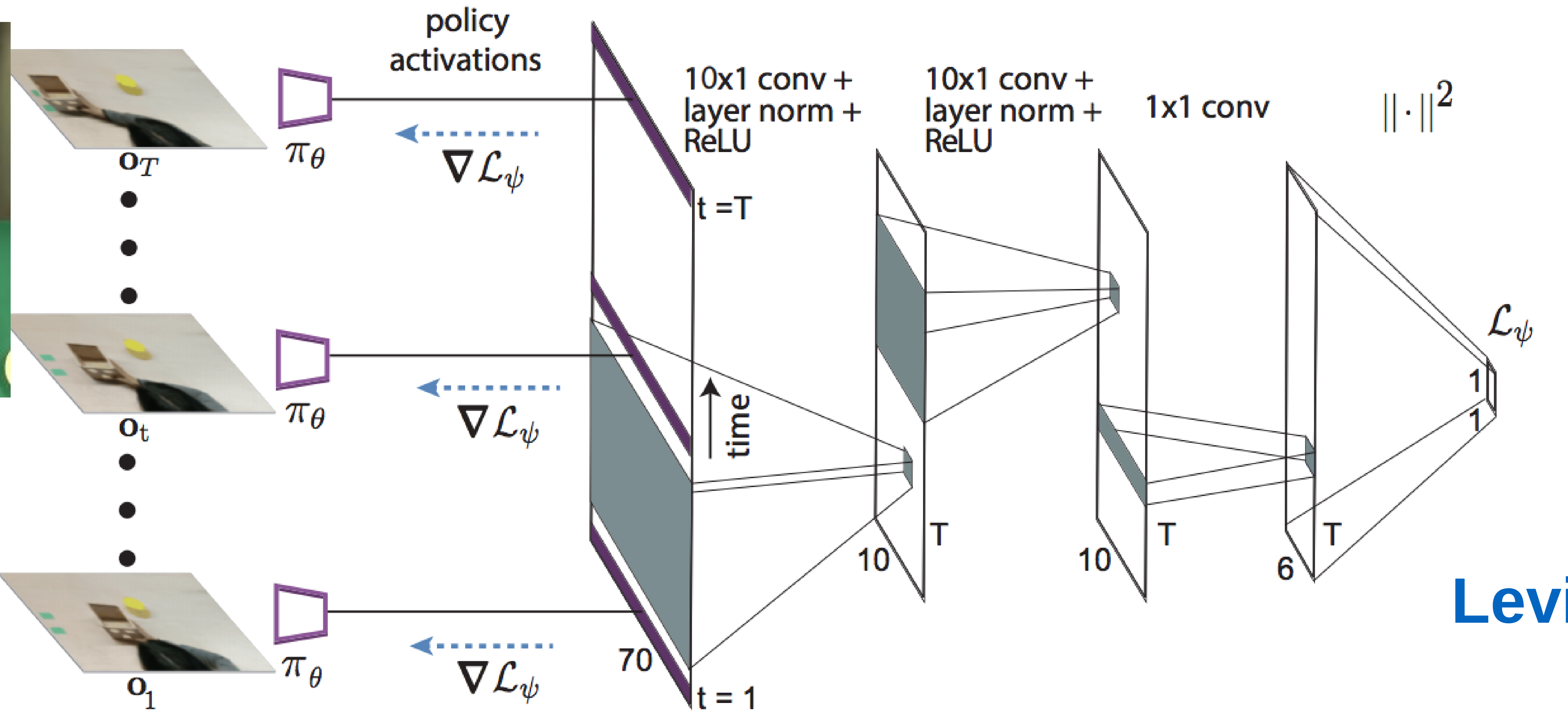
What if the weakly supervised loss is unavailable?

imitation loss

$$\mathcal{L} = \sum_t \|\pi_\theta(o_t) - a^*_t\|^2$$

$$\min_{\theta,\psi} \sum_{\text{task } i} \mathcal{L}^i_{\text{test}}(\theta - \alpha \nabla_\theta \mathcal{L}^i_\psi(\theta))$$

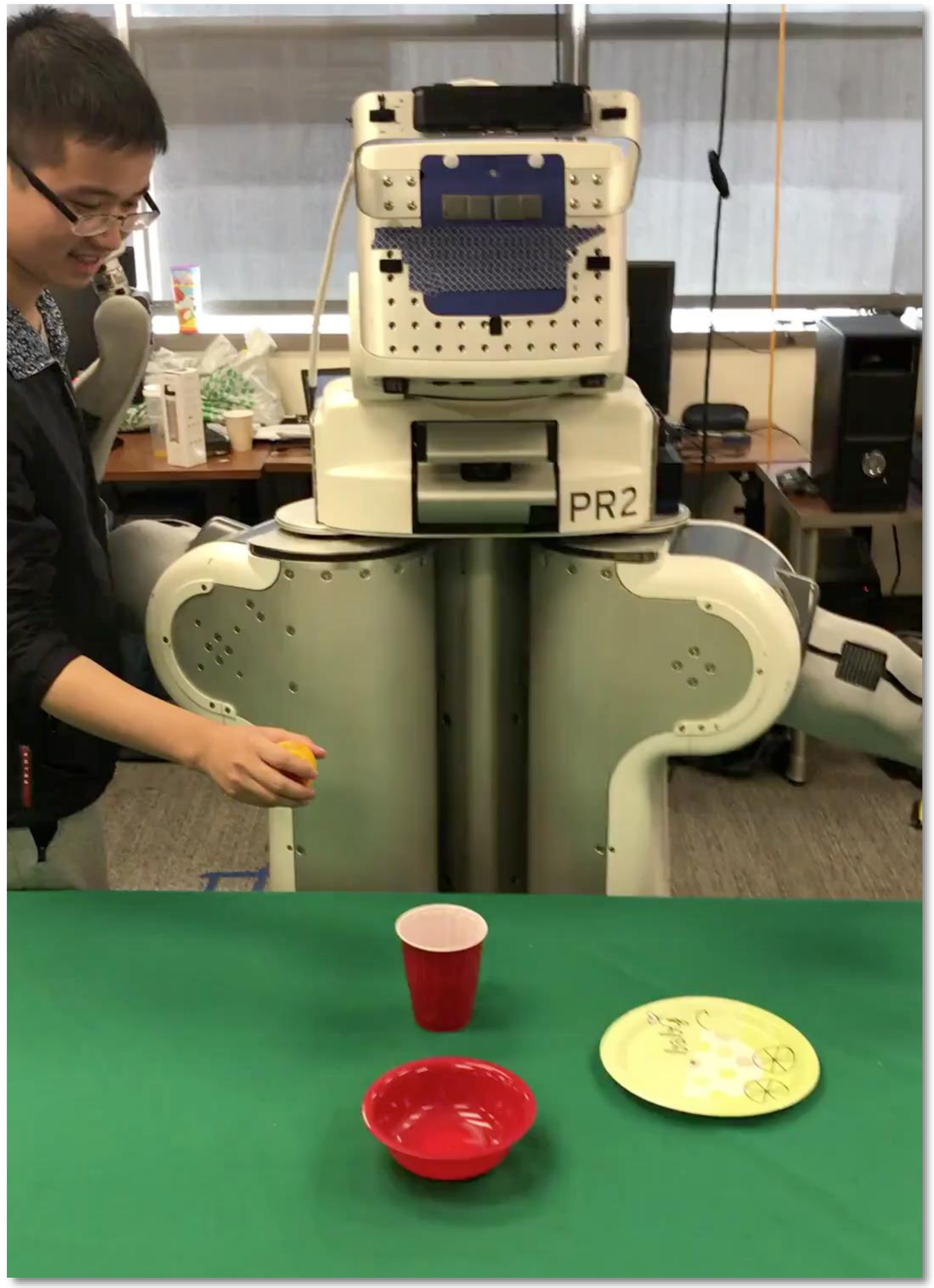




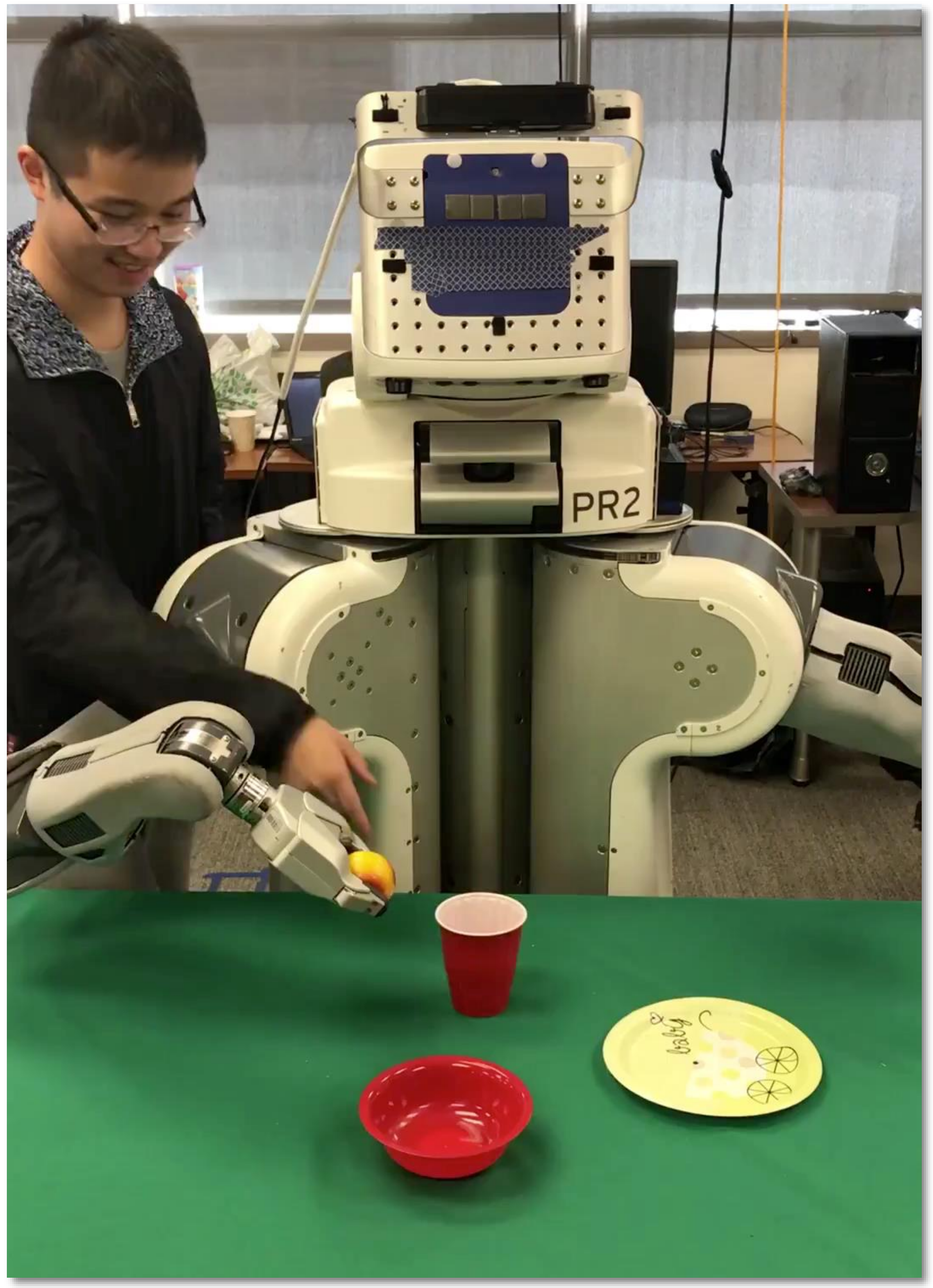
Yu*, Finn*, Xie, Dasari, Zhang, Abbeel, Levine RSS '18
Grant, Finn, Peterson, Abbott, Levine, Darrell, Griffiths NIPS CIAI Workshop '17

One-shot imitation from human video

input human demo



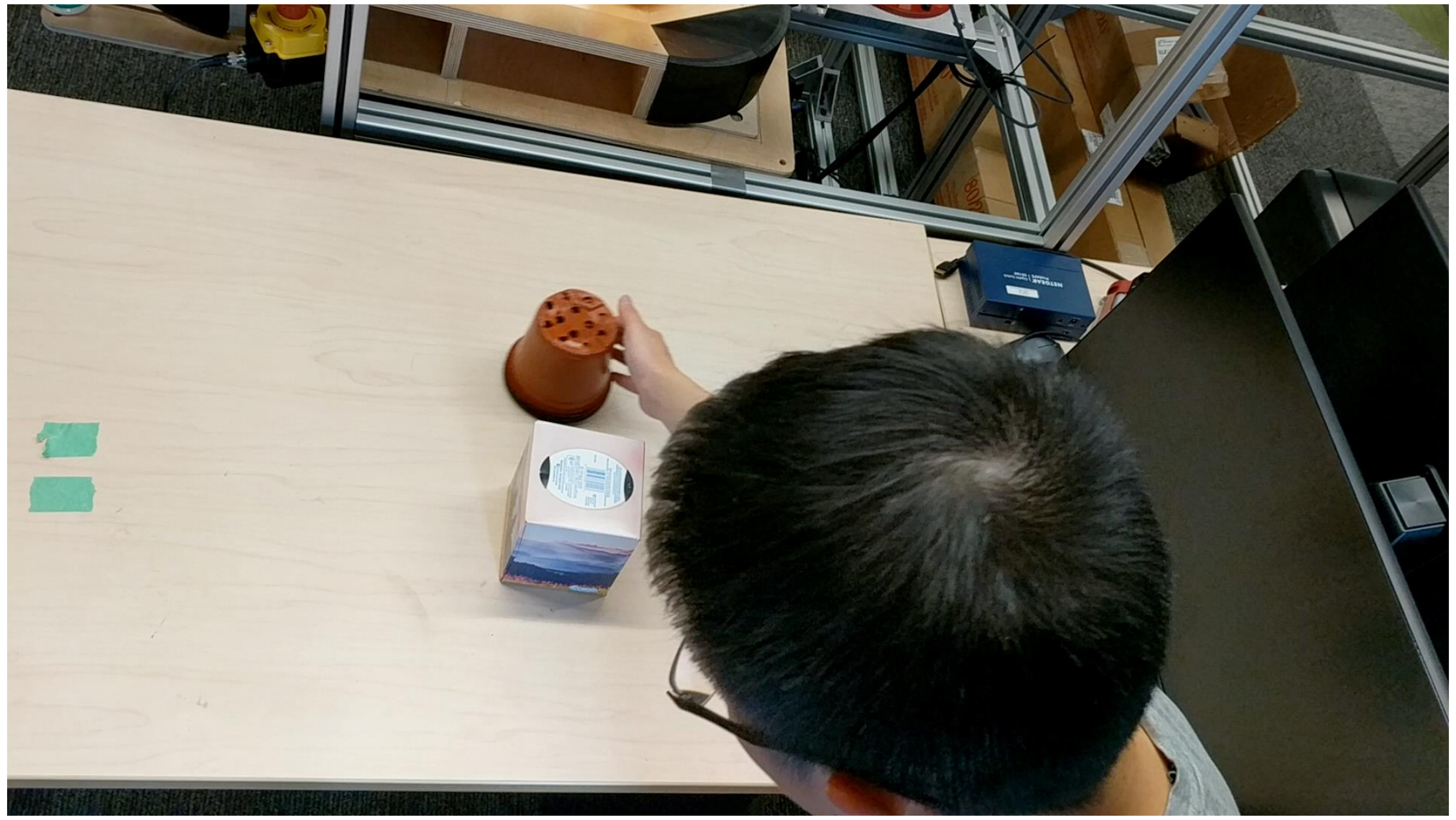
resulting policy



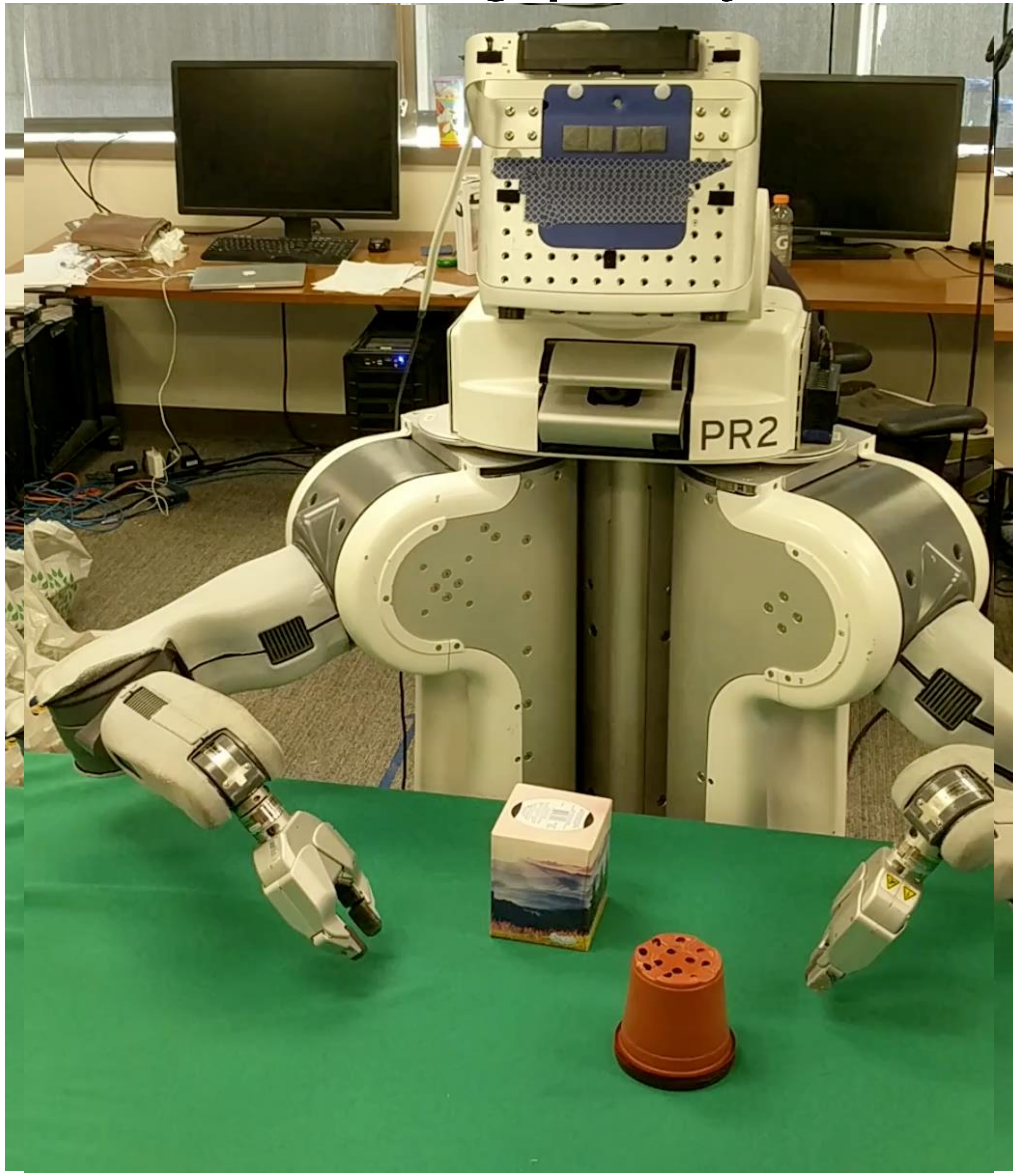
Yu*, Finn*, Xie, Dasari, Zhang, Abbeel, Levine RSS ’18

One-shot imitation from human video

input human demo



resulting policy



Yu*, Finn*, Xie, Dasari, Zhang, Abbeel, Levine RSS '18

Takeaways

- Meta-learning can be seen as learning a function $\mathcal{D}_{\text{train}} \; \mathbf{x}_{\text{test}} \longrightarrow \mathbf{y}_{\text{test}}$
- Discussed three major classes of approaches
  - metric learning approach
    - \+ works well for few-shot classification
  - direct black box approach
    - \+ general, powerful
    - \- data inefficient, poor extrapolation
  - gradient-based approach
    - \+ general, good extrapolation
    - \- somewhat difficult optimization

This class:
1. Get large dataset
2. Get large compute (i.e. GPUs)
3. ?????
4. Profit!

Humans can learn with a very small amount of data. How?

Do neural networks need a large dataset?

Sort of.

Humans still require a lot of data, but not for each and every task.
Data is ***amortized*** across tasks.

Outline

1. Applications of learning to learn
2. Problem formulation
3. Solution Classes:
    a) metric-learning approach
    b) direct black-box approach
    c) gradient-based approach
4. **Open Questions / Problems**

Open Problems in Meta-Learning

- Where do the **tasks** come from?
- Theoretically study of learning performance on **similar, but extrapolated tasks**
- **Meta-RL** algorithms that are **consistent** & **universal**
- **Benchmarks**, **datasets**, **environments** for meta-learning
- Meta-learning for **continual learning** (NIPS 2018 competition)

Further Reading on Meta-Learning

- **Metric learning**: Vinyals et al., Snell et al.
- **Black-box approach**: Hochreiter et al. ’01, Santoro et al. ICML ’16, Li & Malik arXiv ’18, Ha et al. ICLR ‘17, Mishra et al. ICLR ‘18
- **Gradient-based approach**: Finn et al. ICML ’17
- **Bayesian concept learning**: Tenenbaum thesis ’99, Lake et al. Science ‘15
- **Meta-reinforcement learning**: J. Wang et al. ’16, Y. Duan et al. ’16,
  A. Gupta et al. ’18, Clavera et al. ’18, Houthooft et al. ’18
- **Few-shot generative modeling**: Reed et al. ICLR ’18
- **One-shot visual imitation**: Yu et al. RSS ’18
- **Semi-supervised few-shot learning:** Ren et al. ICLR ’18
- **Learning unsupervised learning rules**: L. Metz et al. ‘18

excludes architecture search, learning update rules, hyperparameter optimization

Questions?

Further Resources:
Learning to Learn blog post:
http://bair.berkeley.edu/blog/2017/07/18/learning-to-learn/
Learning to Optimize blog post:
http://bair.berkeley.edu/blog/2017/09/12/learning-to-optimize-with-rl/
Meta-RL lecture (Deep RL course): https://youtu.be/Xe9bktyYB34

Universal Function Approximation Theorem

Hornik et al. '89, Cybenko '89, Funahashi '89

A neural network with one hidden layer of finite width can approximate any continuous functio

$$\mathbf{y} = f(\mathbf{x}; \theta)$$

"universal function approximator"

How can we define a notion of universality / expressive power for meta-learning

$$\mathbf{y}_{\text{test}} = f(\mathcal{D}_{\text{train}}, \mathbf{x}_{\text{test}}; \theta)$$

"universal learning procedure approximator"

Recurrent network

$$\mathbf{y}_{\text{test}} = f(\mathcal{D}_{\text{train}}, \mathbf{x}_{\text{test}}; \theta)$$

Learned optimizer

$$\mathbf{y}_{\text{test}} = f(\mathbf{x}_{\text{test}}; g(\mathcal{D}_{\text{train}}; \theta))$$

With sufficient depth, both are universal learning procedure approximators.

Are we losing expressive power when using MAML?

Finn & Levine '17

Can we interpret MAML in a probabilistic framework?

meta-learning ≈ learning a prior

Bayesian concept learning

[Tenenbaum ’99, Fei-Fei et al. ’03, Lawrence & Platt ’04, …]

formulate few-shot learning as probabilistic inference problem

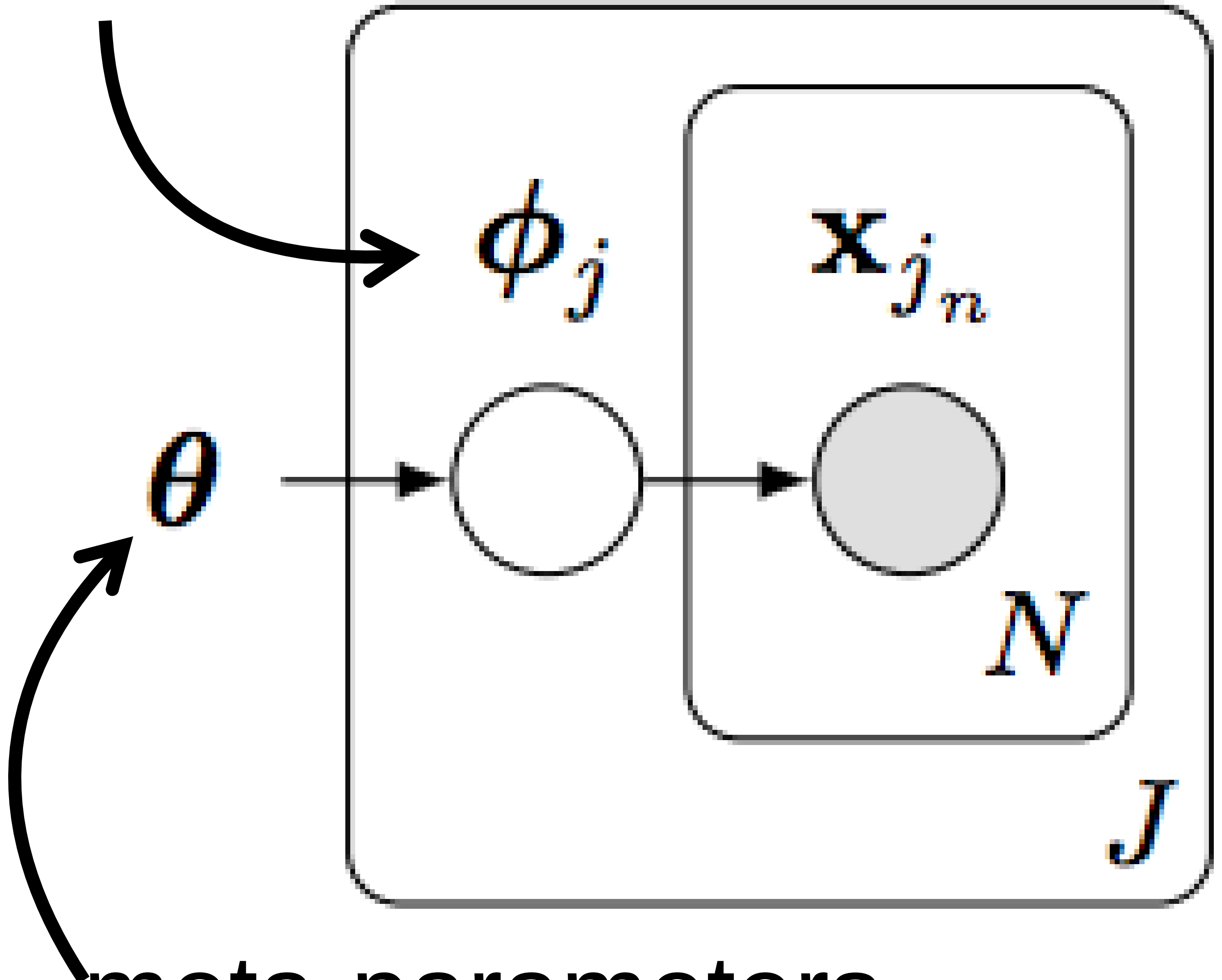
+ can effectively generalize from limited evidence

- hard to scale to complex models

Can we interpret MAML in a probabilistic framework?

Bayesian meta-learning approach

task-specific parameters



meta-parameters

$$\max_\theta \prod_j p(\mathcal{D}_{\text{train}}^{(j)} | \theta)$$

$$= \prod_j \int p(\mathcal{D}_{\text{train}}^{(j)} | \phi_j) p(\phi_j | \theta) d\phi_j$$

(empirical Bayes)

$$\approx \prod_j p(\mathcal{D}_{\text{train}}^{(j)} | \hat{\phi}_j) p(\hat{\phi}_j | \theta)$$

MAP estimate

How to compute MAP estimate?

Gradient descent with early stopping = MAP inference under Gaussian prior with mean at initial parameters [Santos '96]

(exact in linear case, approximate in nonlinear case)

MAML approximates hierarchical Bayesian inference. [Grant et al. '17]

Erin Grant

Chelsea Finn, UC Berkeley

Learning to Learn from Weak Supervision

Meta-Supervised Learning:

Inputs: $\boxed{\mathcal{D}_{\text{train}}}$ $\boxed{\mathbf{x}_{\text{test}}}$ Outputs: $\boxed{\mathbf{y}_{\text{test}}}$ Data: $\{\mathcal{D}_i\}$

Weakly supervised

Fully supervised

During meta-training: access full supervision for each task

During meta-testing: only use weakly-supervised datapoints

With MAML: $$\min_\theta \sum \mathcal{L}_{\text{v}}(\theta - \alpha \nabla_\theta \mathcal{L}_{\text{tr}}(\theta))$$

Key insight: **inner loss** can be different than **outer loss**

Examples: *only some data is labeled imitation without actions*

Weak Supervision Results

- Learning from positive examples
  Grant, Finn, Peterson, Abbott, Levine, Darrell, Griffiths, NIPS ‘17 CIAI workshop
- One-shot Imitation from human video
  (in preparation, with Yu, Abbeel, Levine)

Typical Objective of Few-Shot Learning

Image recognition

Given 1 example of 5 classes:

Classify new examples

Human Concept Learning

Given *1 positive example*:

Classify new examples:

Beyond how humans learn, this setting is also more interesting.

Grant et al. ’17 (NIPS CIAI workshop)

Human Concept Learning

Given *1 positive example*:

Classify new examples:

$$\min_{\theta} \sum \mathcal{L}_{\mathrm{v}}(\theta - \alpha \nabla_{\theta} \mathcal{L}_{\mathrm{tr}}(\theta))$$

only positive examples

both positive & negatives

Why does this make sense?

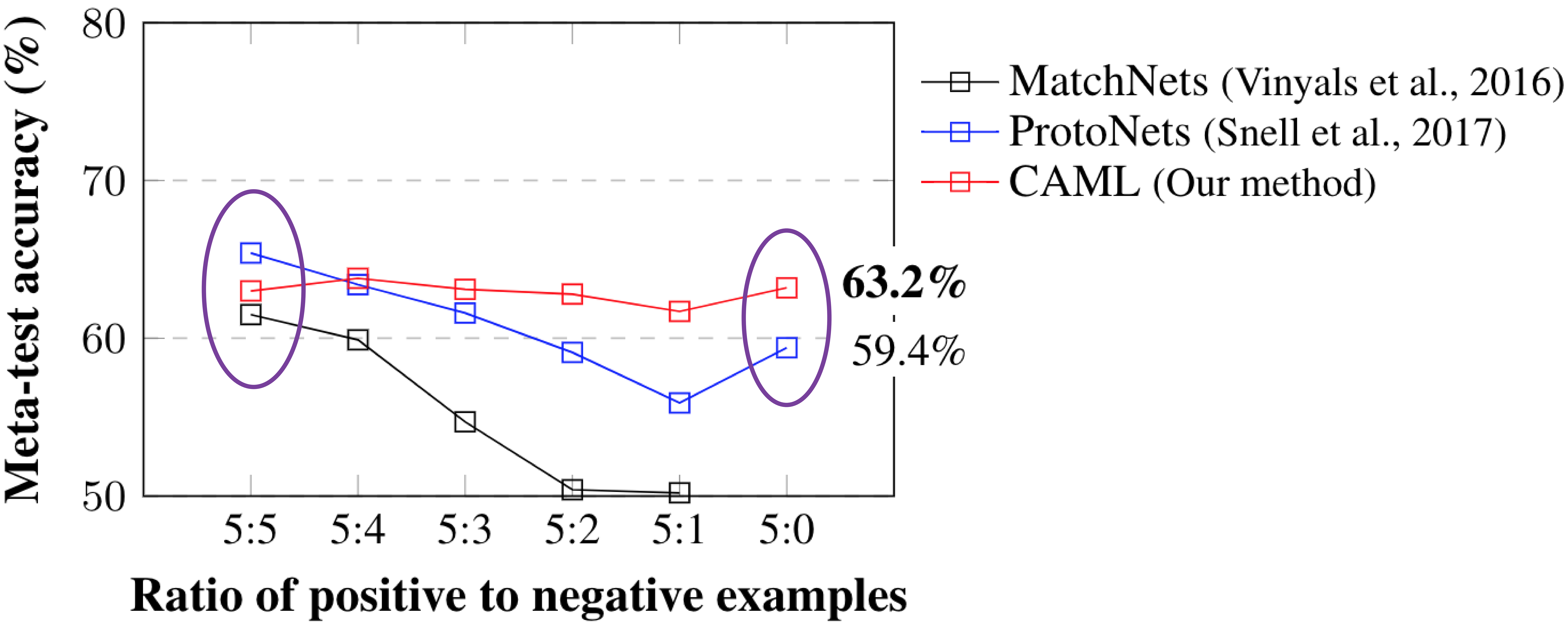
MAML approximates hierarchical Bayesian inference

Concept **A**cquisition through **M**eta-**L**earning (CAML)

Grant et al. ’17 (NIPS CIAI workshop)

Few-Shot Image Classification from Positive Examples

MiniImagenet dataset



Grant et al. '17 (NIPS CIAI workshop)

One-Shot Visual Imitation Learning

Goal: Given one visual demonstration of a new task, learn a policy

Visual imitation is expensive.

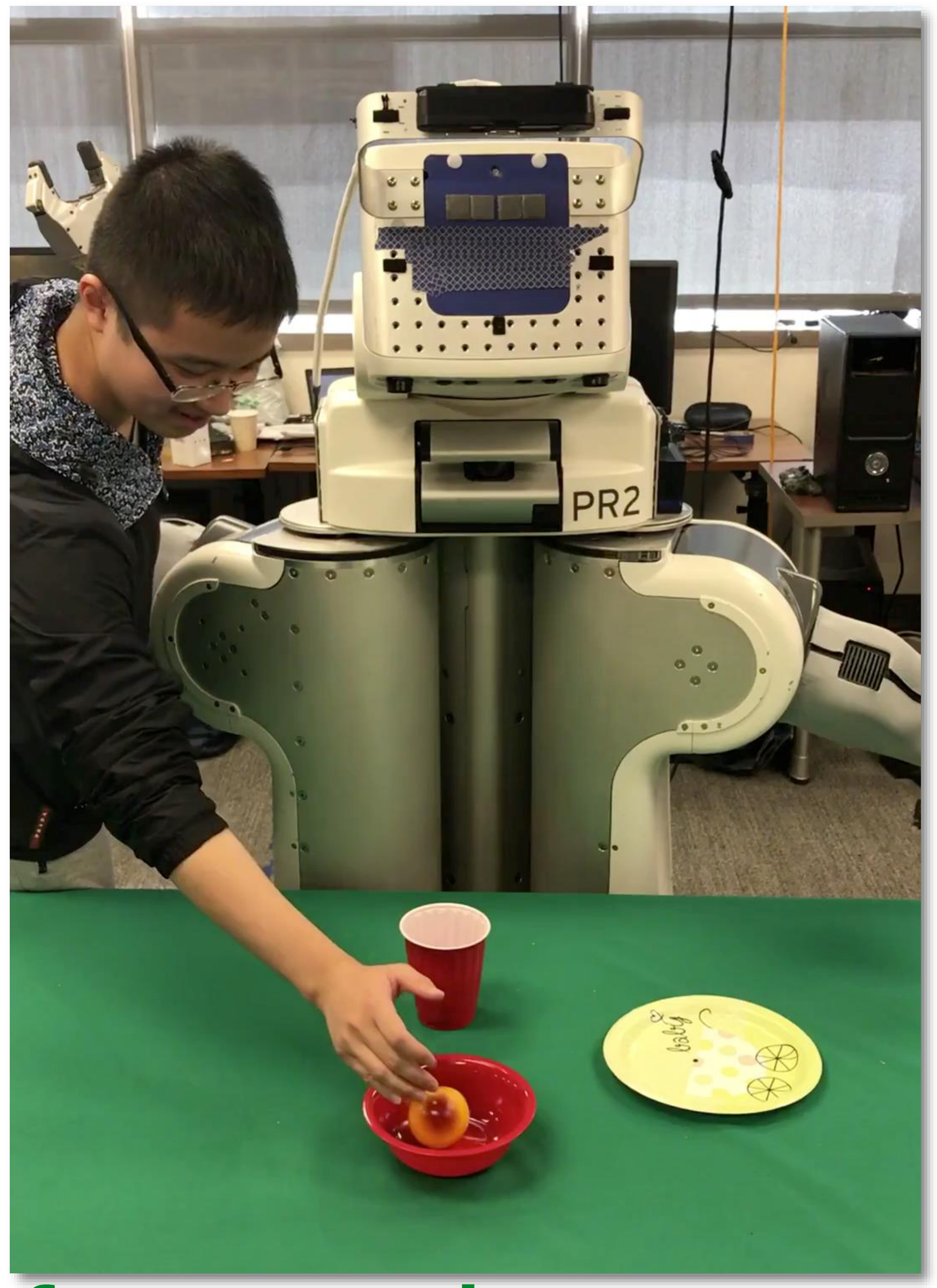
behavior cloning / supervised learning

Rahmanizadeh et al. ‘17 Zhang et al. ‘17

learns from raw pixels,

but requires many demonstrations

No direct supervision signal in video of human.



Through meta-learning: reuse data from other tasks/objects/envionrments

Yu*, Finn*, et al.

One-Shot Visual Imitation from Humans

imitation loss

$$\mathcal{L} = \sum_t \|\pi_\theta(o_t) - a_t^*\|^2$$

meta-training time $$\min_\theta \sum_{\text{tasks}} \mathcal{L}_{\text{v}}(\theta - \alpha \nabla_\theta \mathcal{L}_{\text{tr}}(\theta))$$

meta-training tasks

val demo (robot demo)

training demo (video of human)

meta-test time $$\theta' \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}(\theta)$$

demo of meta-test task (video of human)

Yu*, Finn*, et al.