Online Meta-Learning

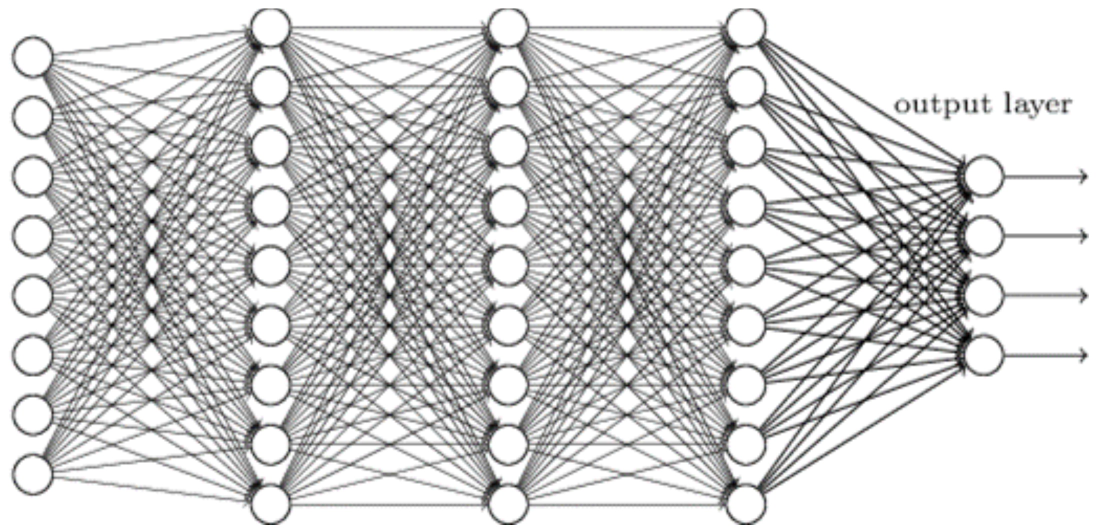
Chelsea Finn*, Aravind Rajeswaran*, Sham Kakade, Sergey Levine

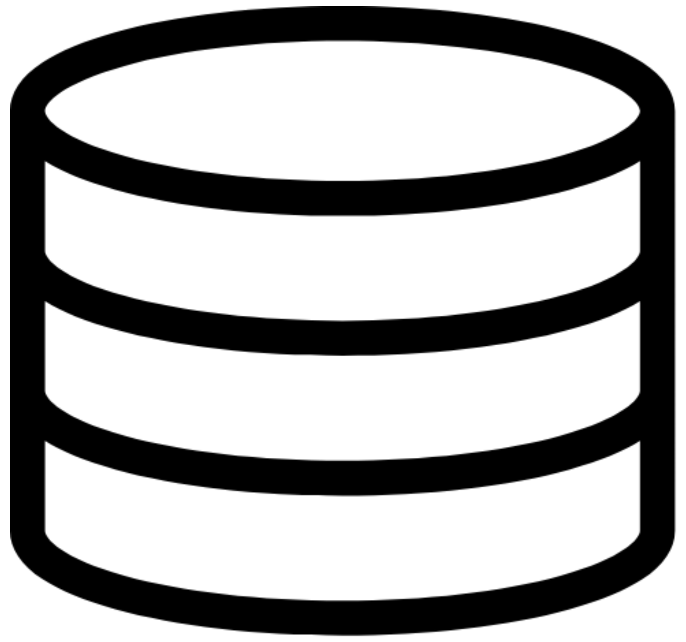



BAIR
BERKELEY ARTIFICIAL INTELLIGENCE RESEARCH

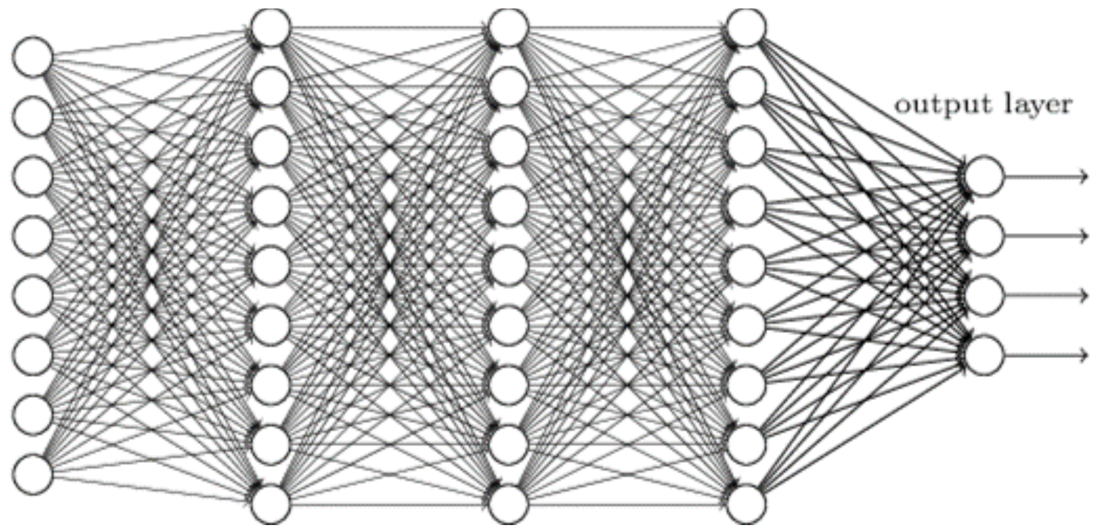
W PAUL G. ALLEN SCHOOL
OF COMPUTER SCIENCE & ENGINEERING

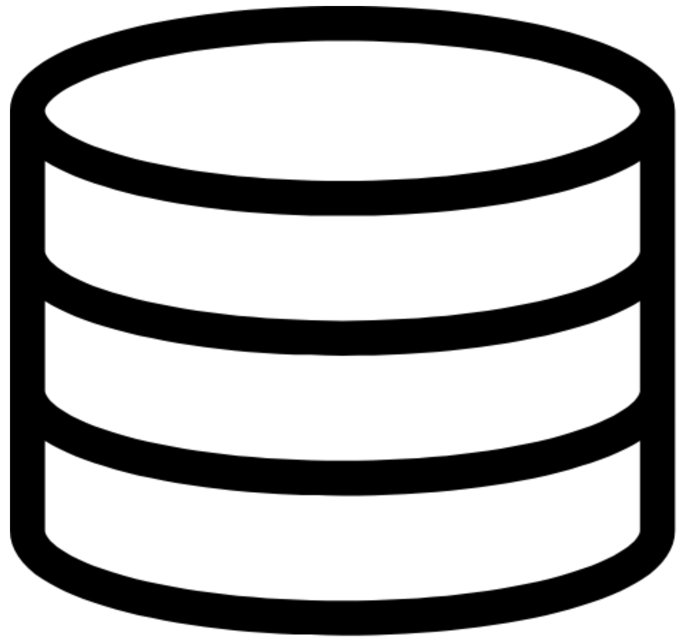
Deep networks + large datasets = 



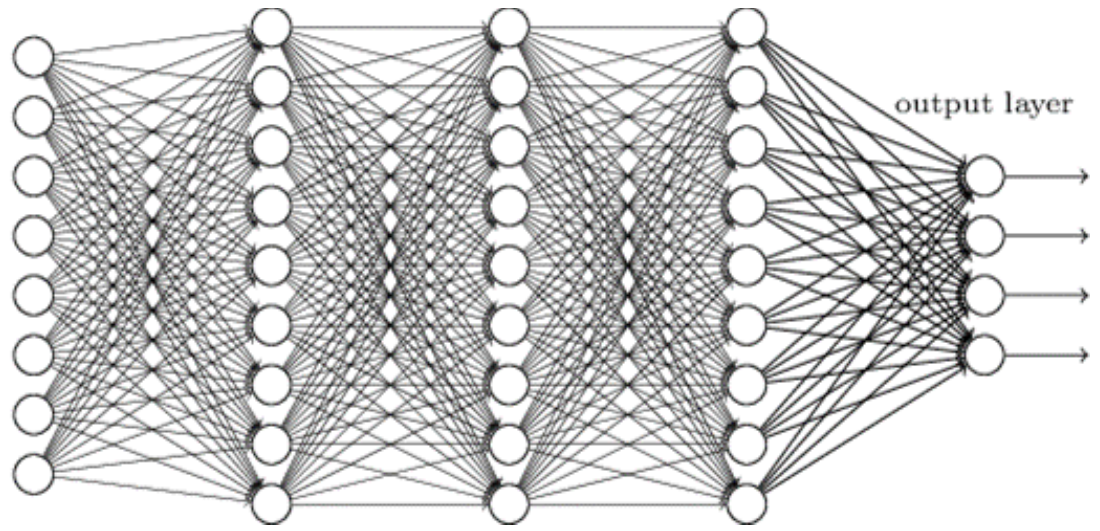


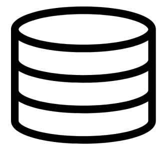
Deep networks + large datasets = 



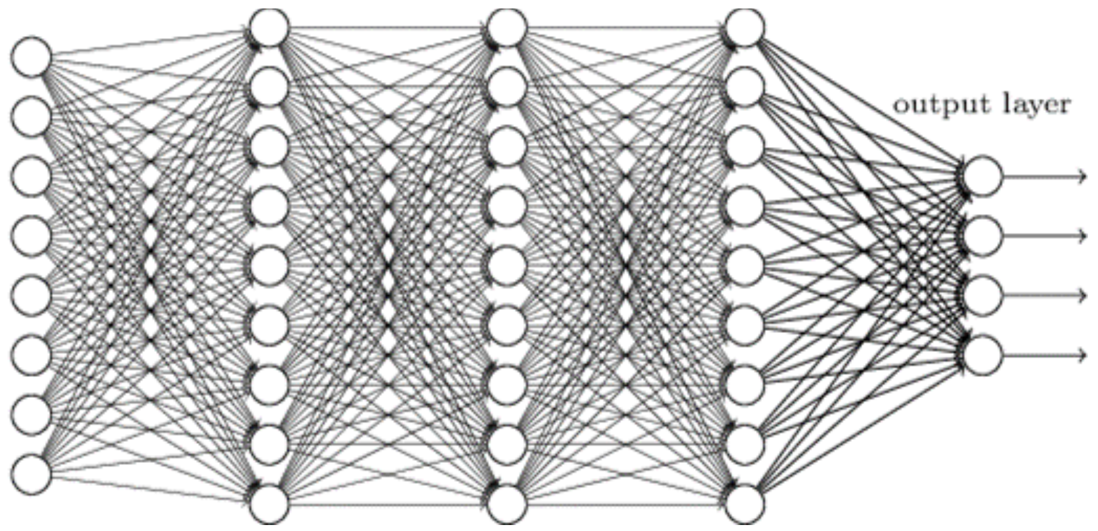


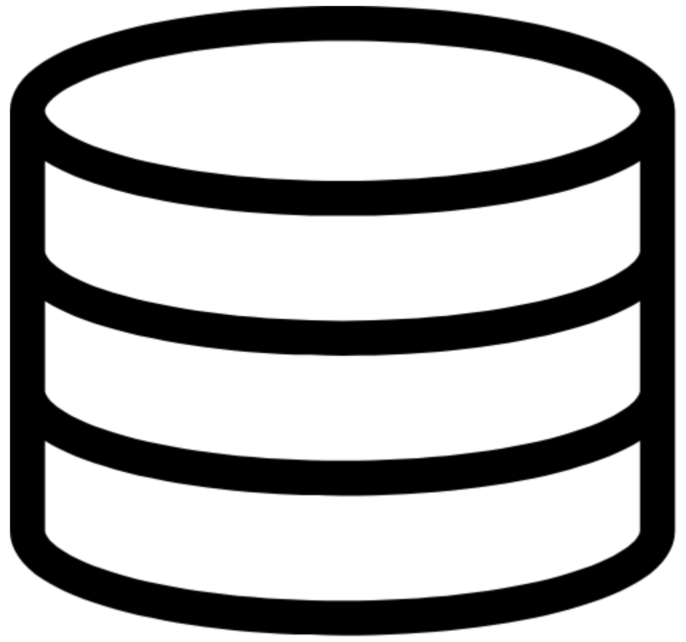
In many practical situations:
Learn new task with only a **few** datapoints



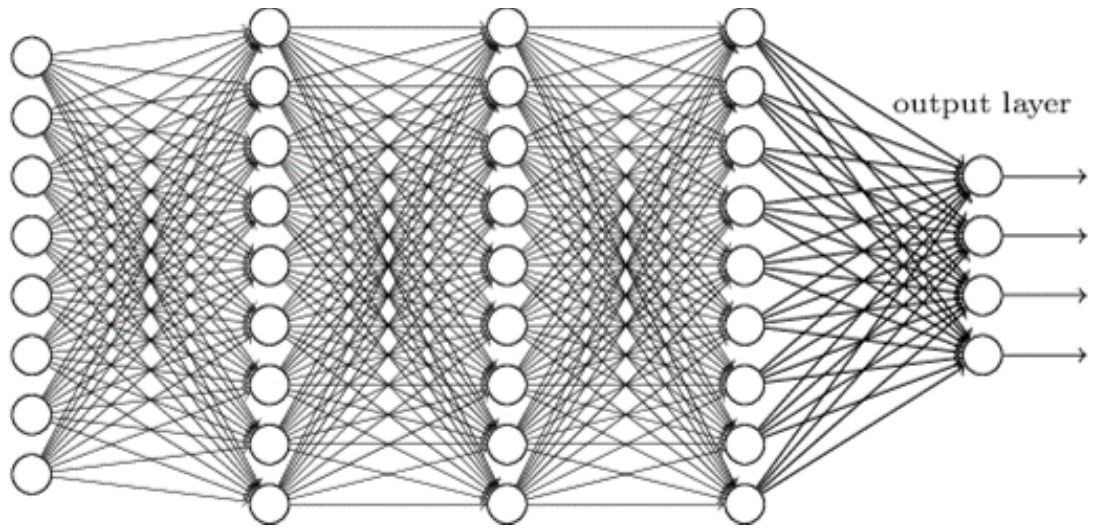


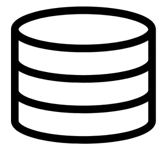
Deep networks + large datasets = 





In many practical situations:
Learn new task with only a **few** datapoints



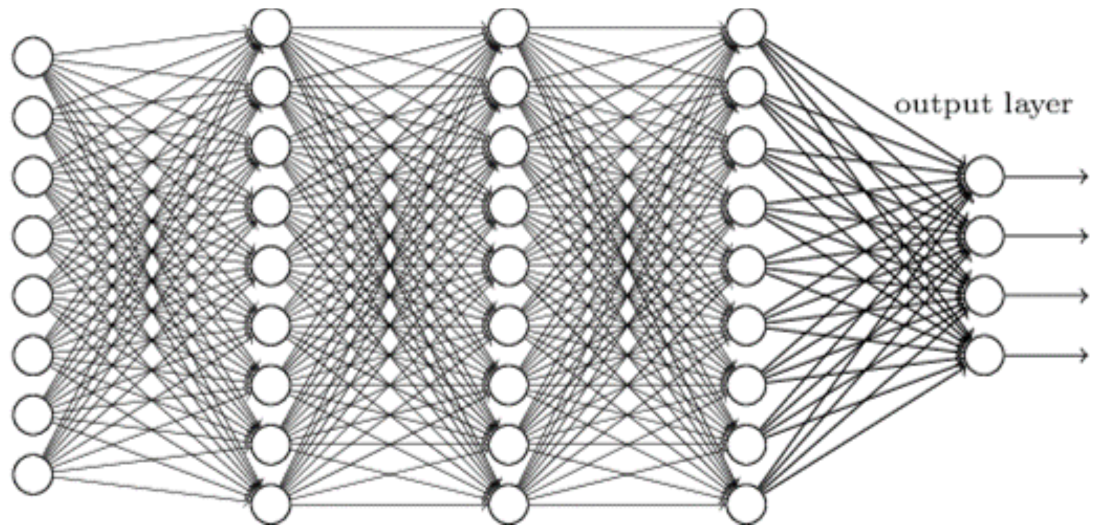


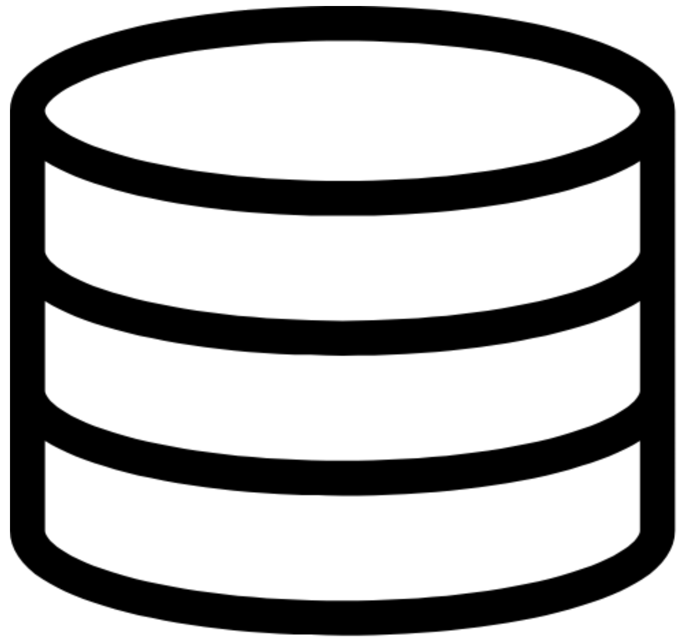
Meta-Learning

(Schmidhuber et al. ’87, Bengio et al. ’92)

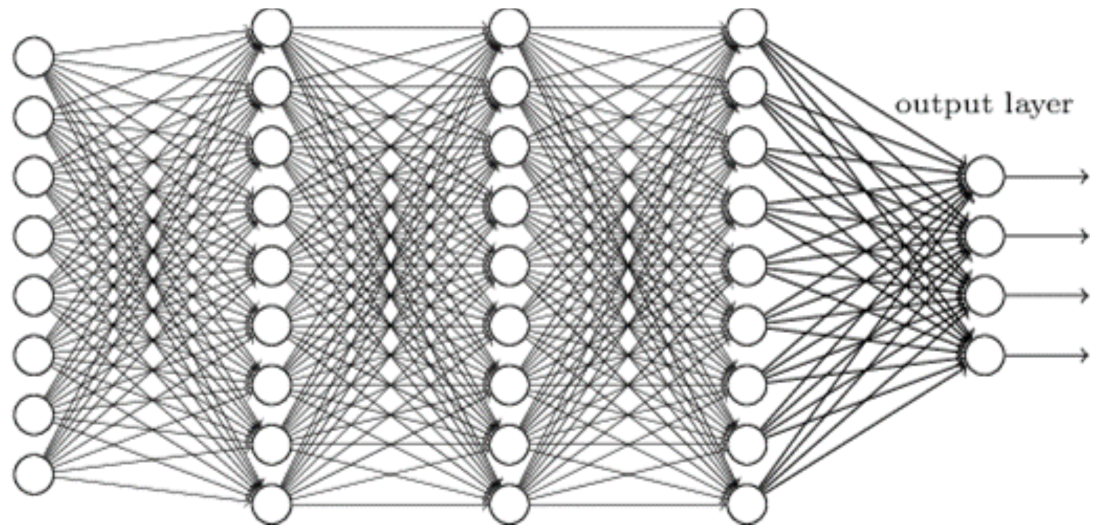
Given i.i.d. task distribution,
learn a new task efficiently

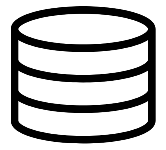
Deep networks + large datasets = 😍





In many practical situations:
Learn new task with only a **few** datapoints



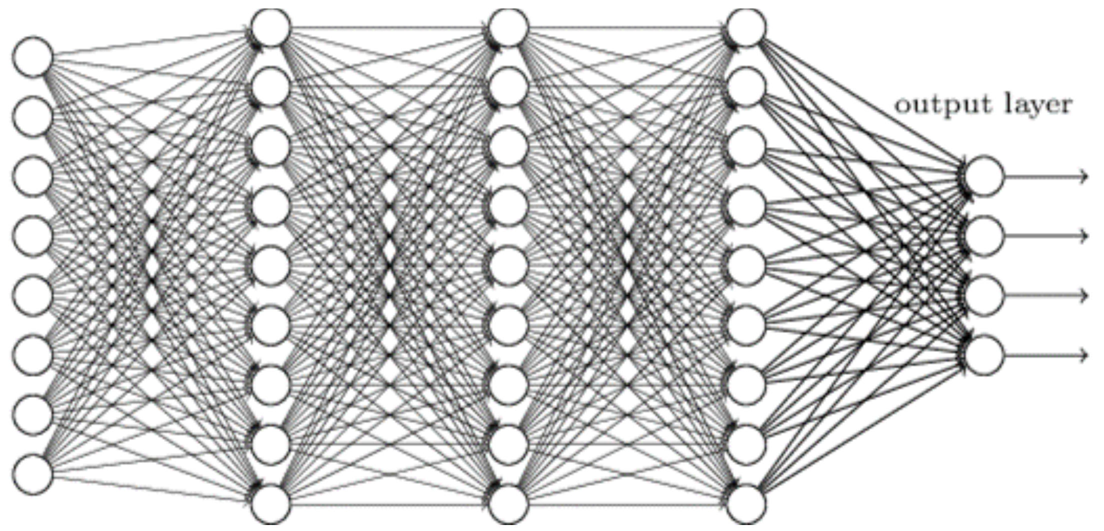


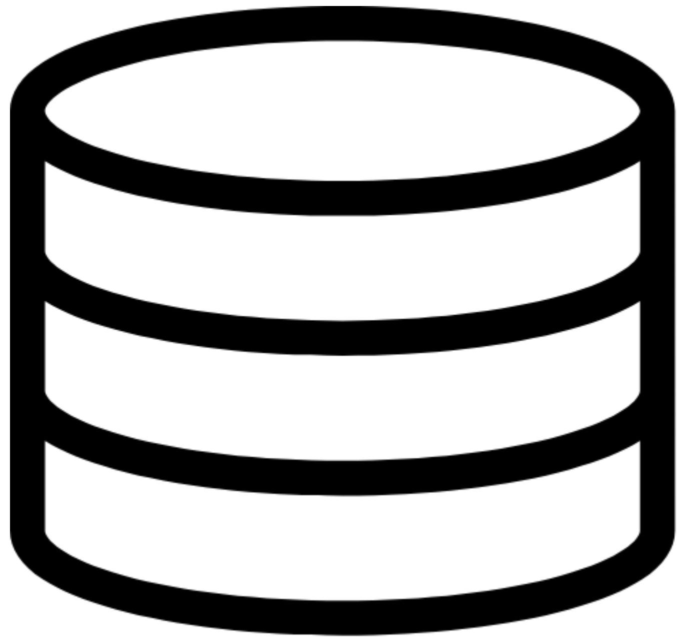
Meta-Learning

(Schmidhuber et al. '87, Bengio et al. '92)

Given i.i.d. task distribution,
learn a new task efficiently

Deep networks + large datasets = 



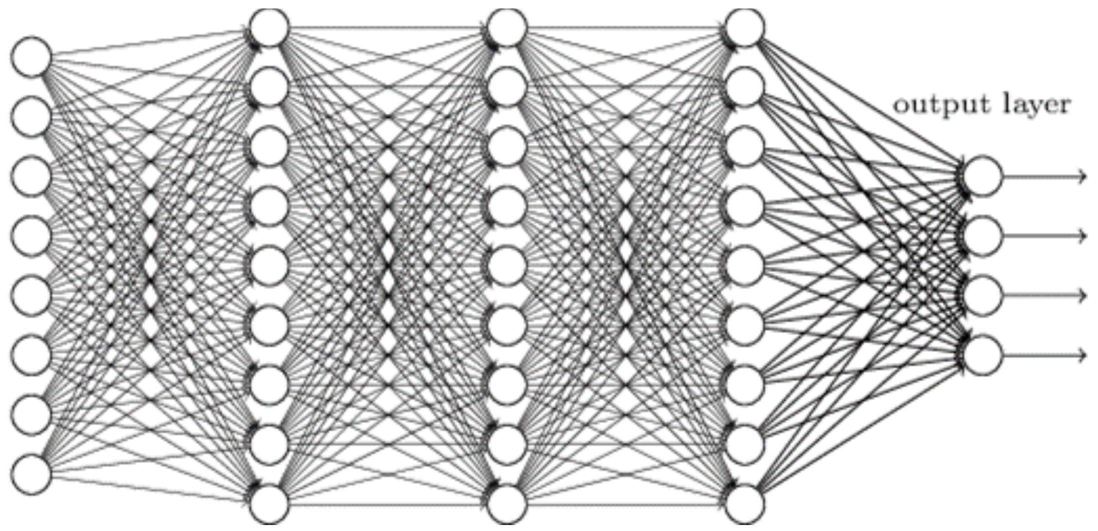


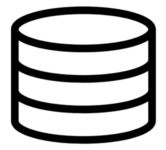
Meta-Learning

(Schmidhuber et al. ’87, Bengio et al. ’92)

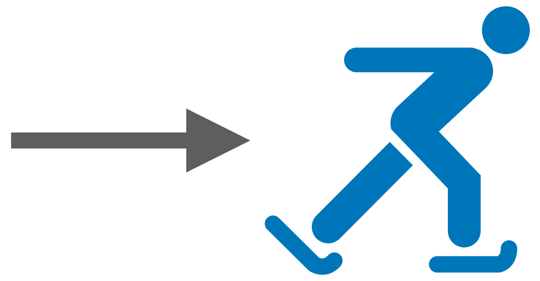
Given i.i.d. task distribution,
learn a new task efficiently

In many practical situations:
Learn new task with only a **few** datapoints

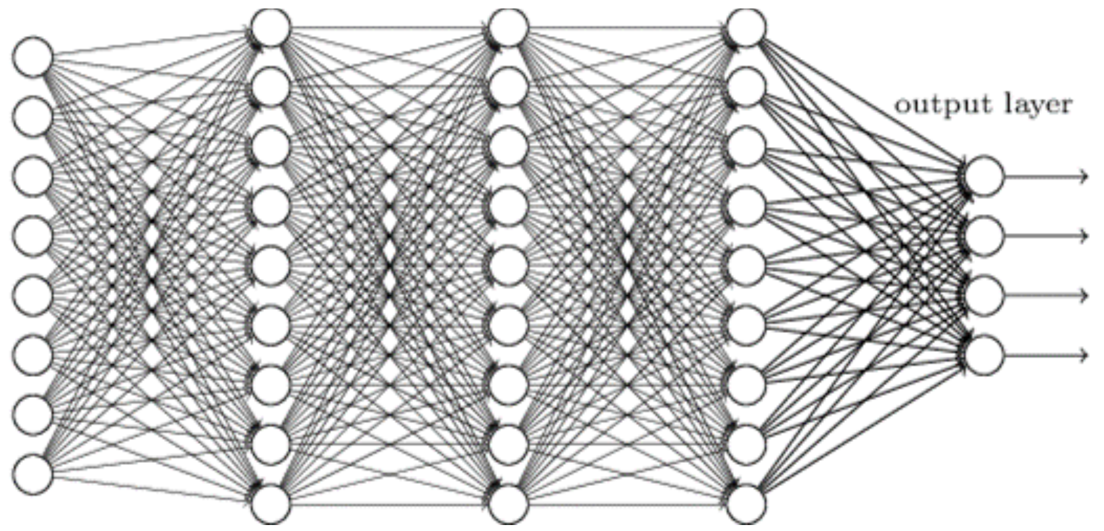


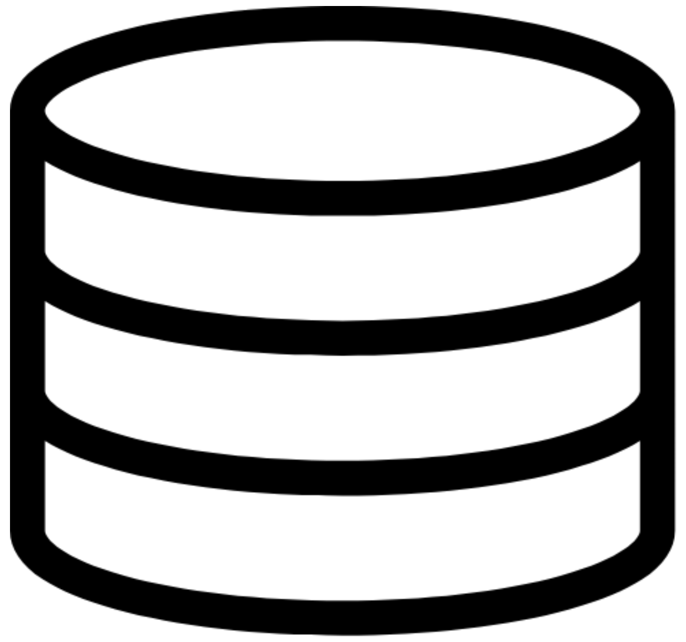


learn

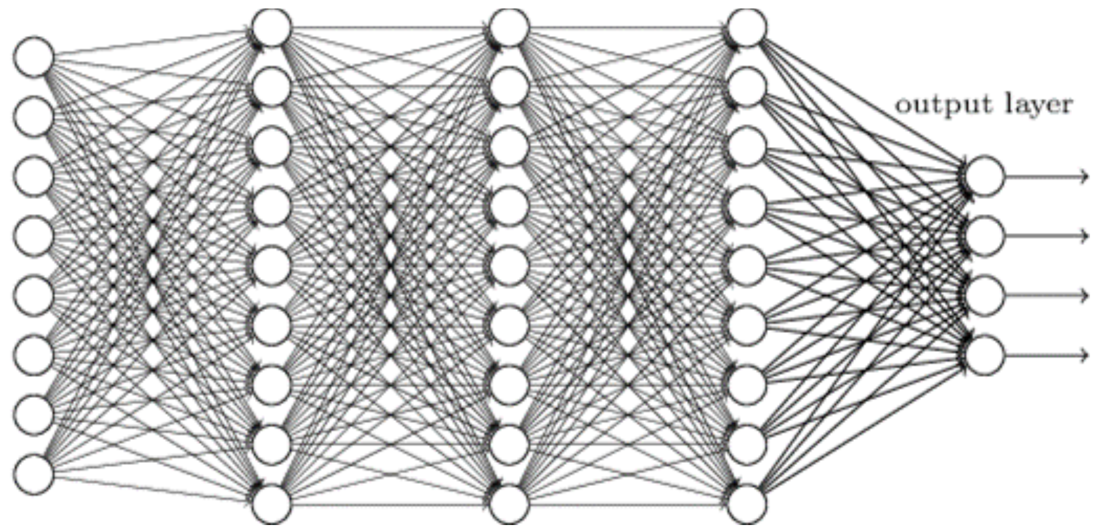


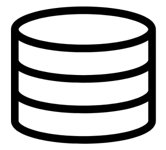
Deep networks + large datasets = 





In many practical situations:
Learn new task with only a **few** datapoints



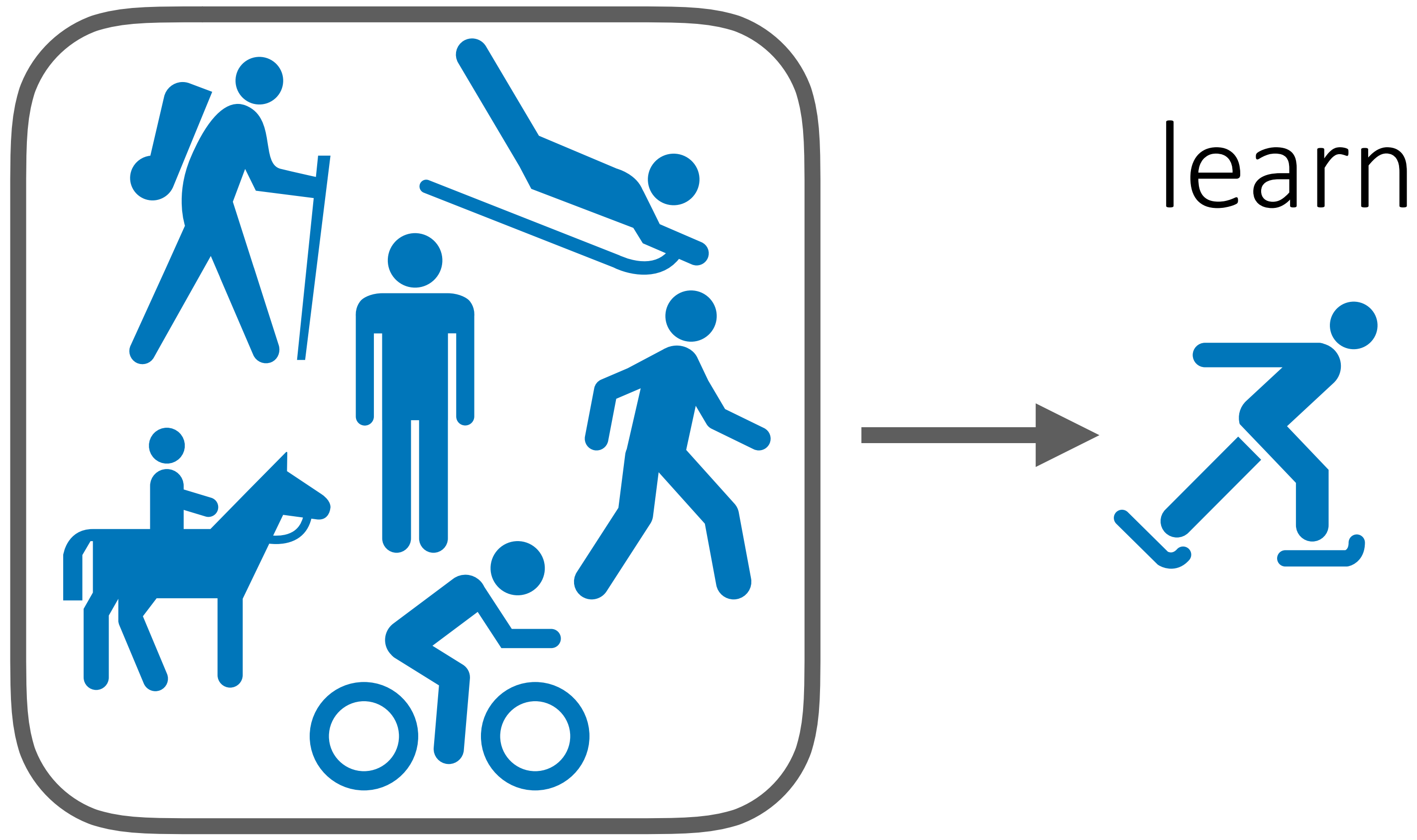


Meta-Learning

(Schmidhuber et al. ’87, Bengio et al. ’92)

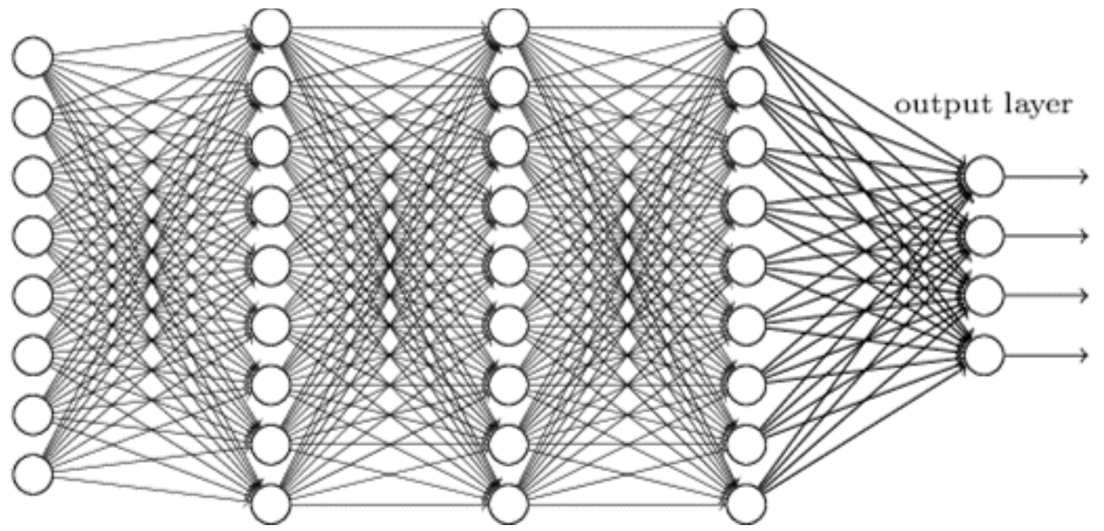
Given i.i.d. task distribution,
learn a new task efficiently

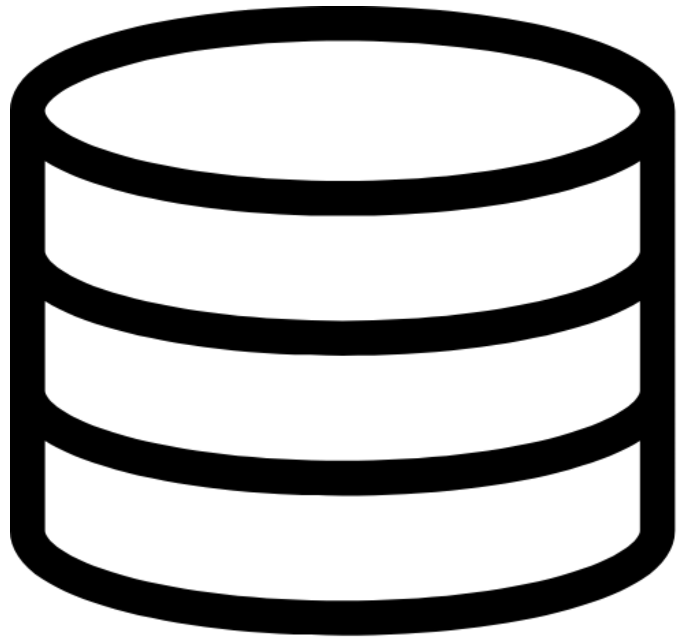
learn



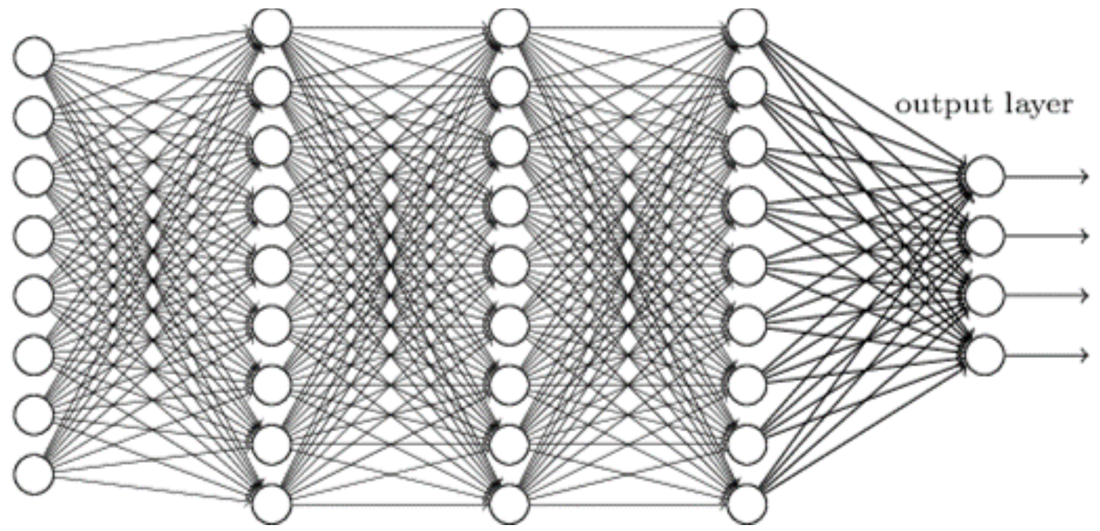
More realistically:

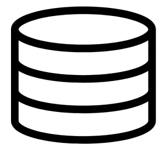
Deep networks + large datasets = 





In many practical situations:
Learn new task with only a **few** datapoints



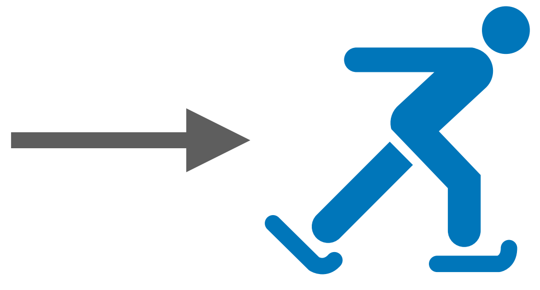


Meta-Learning

(Schmidhuber et al. '87, Bengio et al. '92)

Given i.i.d. task distribution,
learn a new task efficiently

learn

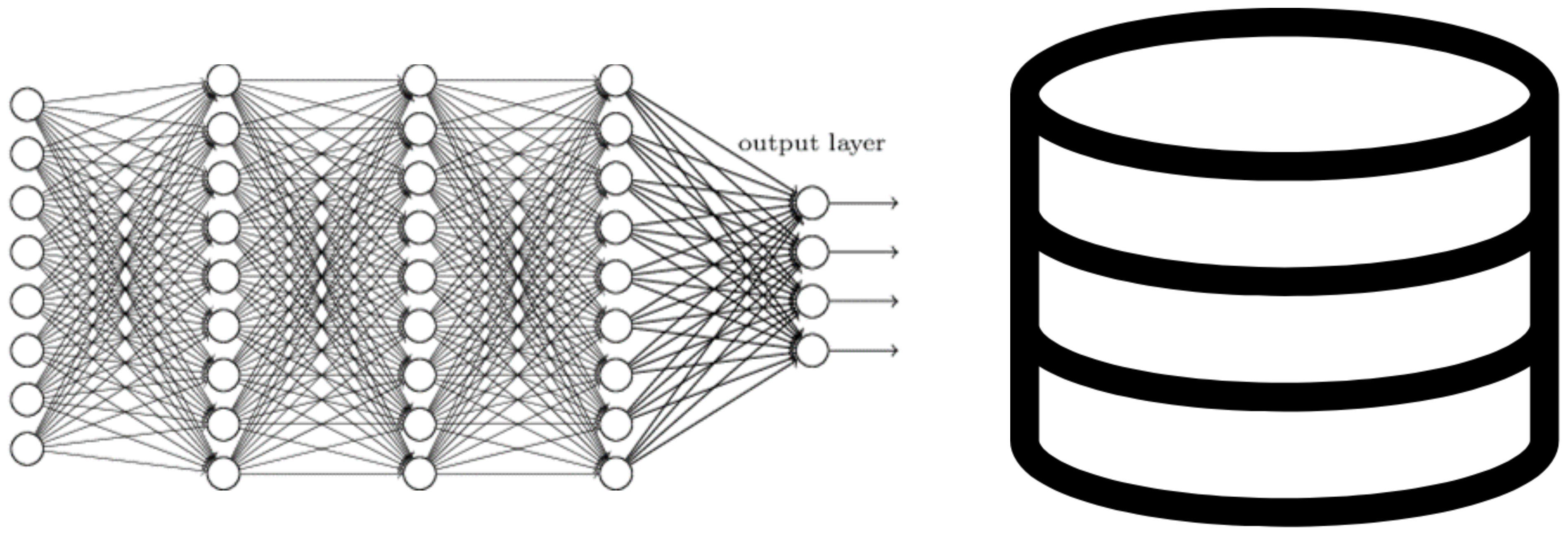


More realistically:

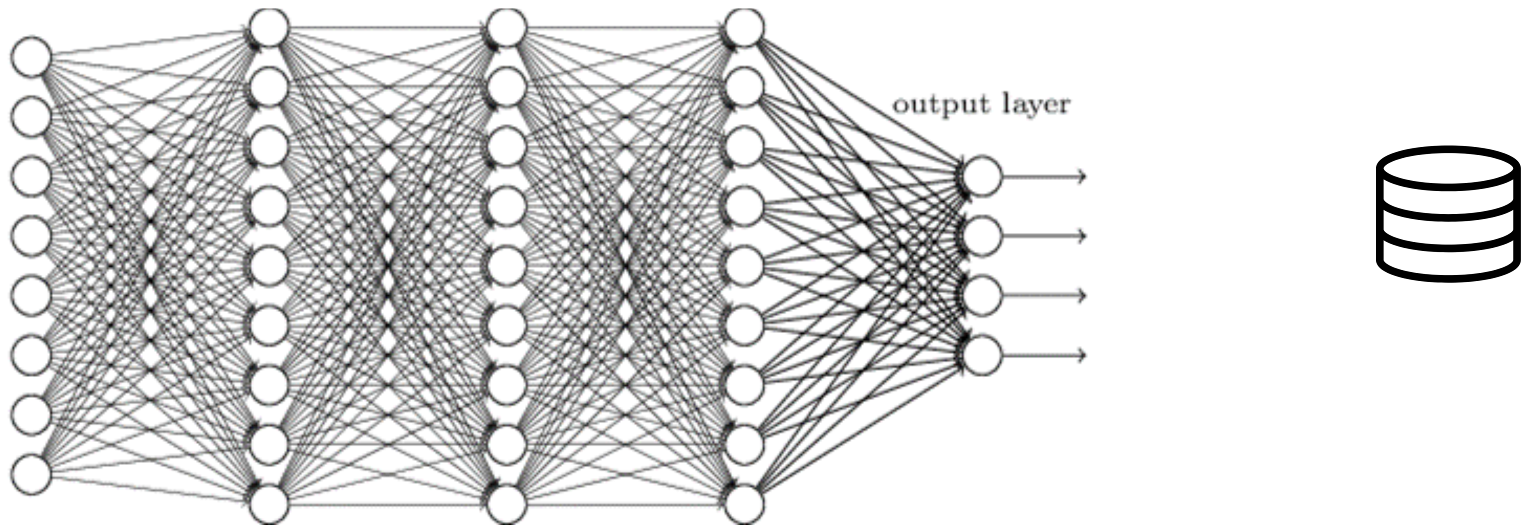
learn

time

Deep networks + large datasets = 😍



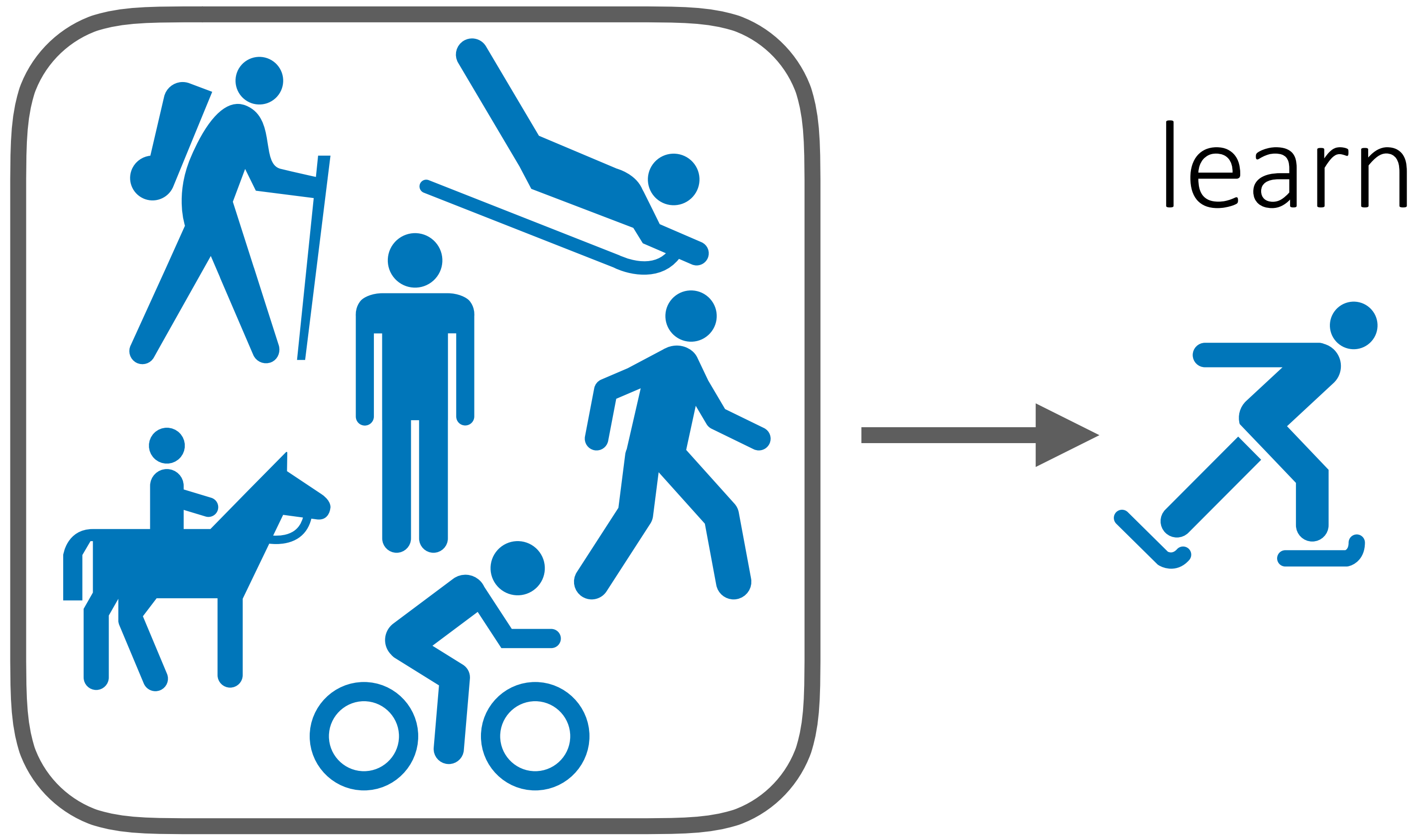
In many practical situations:
Learn new task with only a **few** datapoints



Meta-Learning

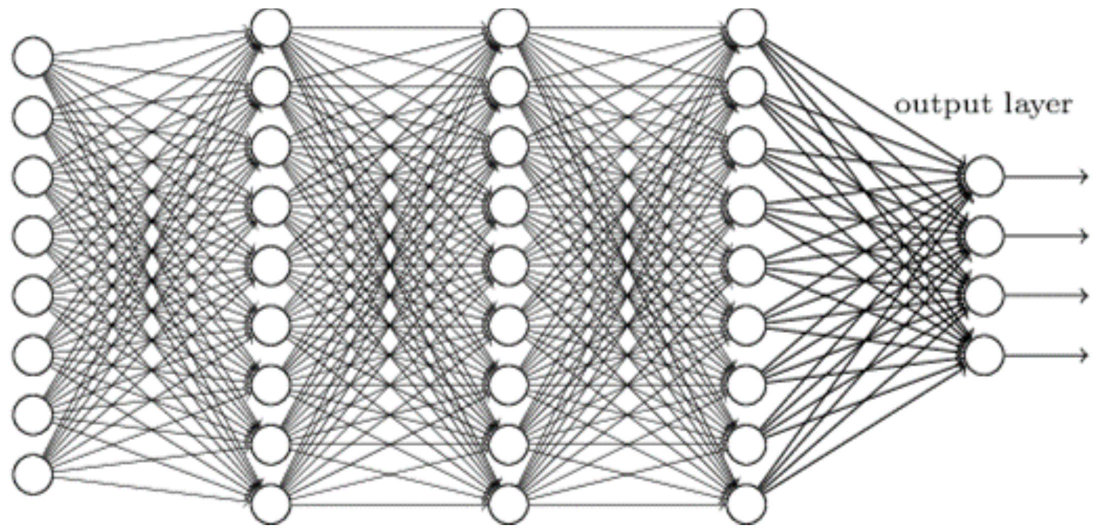
(Schmidhuber et al. ’87, Bengio et al. ’92)

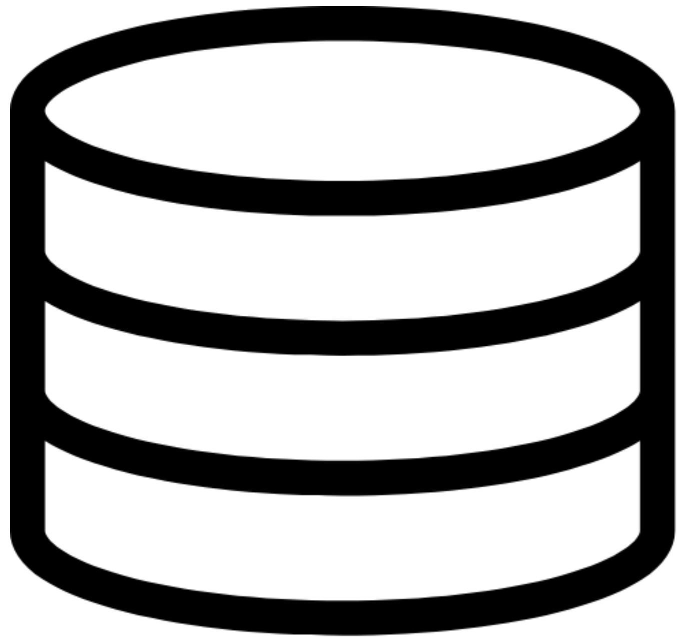
Given i.i.d. task distribution,
learn a new task efficiently



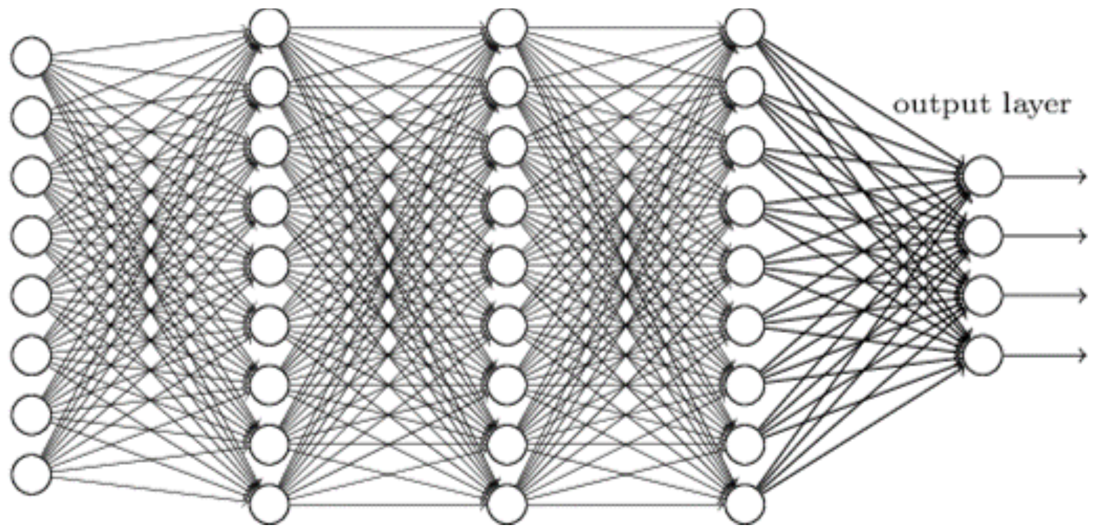
More realistically:

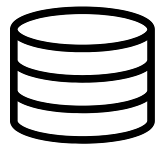
Deep networks + large datasets = 





In many practical situations:
Learn new task with only a **few** datapoints



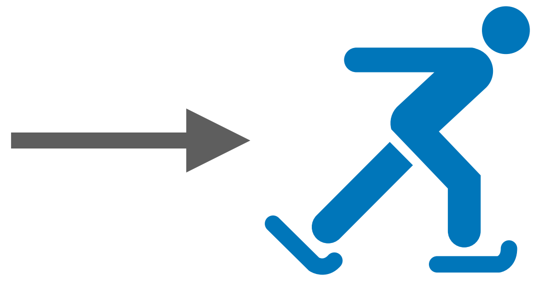


Meta-Learning

(Schmidhuber et al. ’87, Bengio et al. ’92)

Given i.i.d. task distribution,
learn a new task efficiently

learn

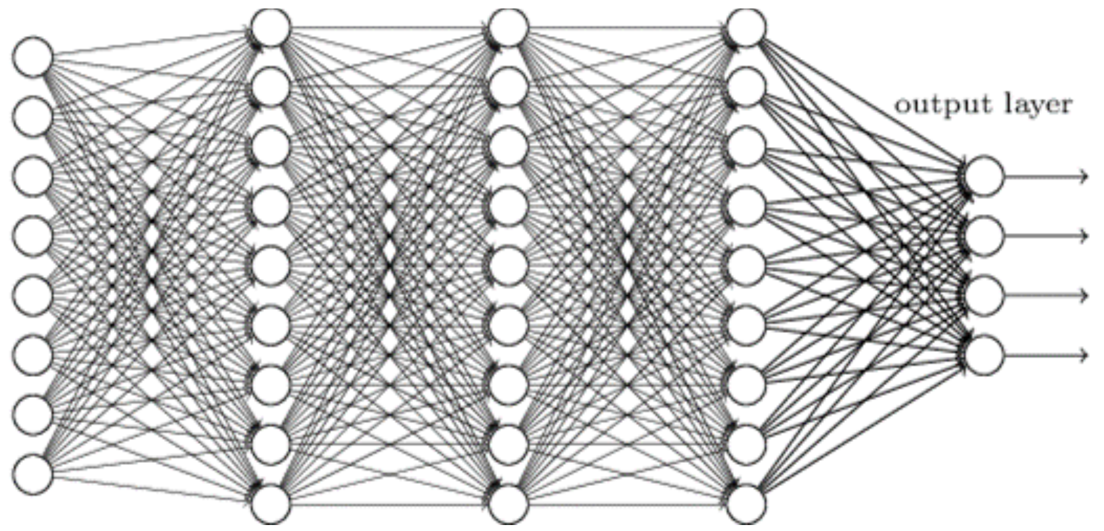


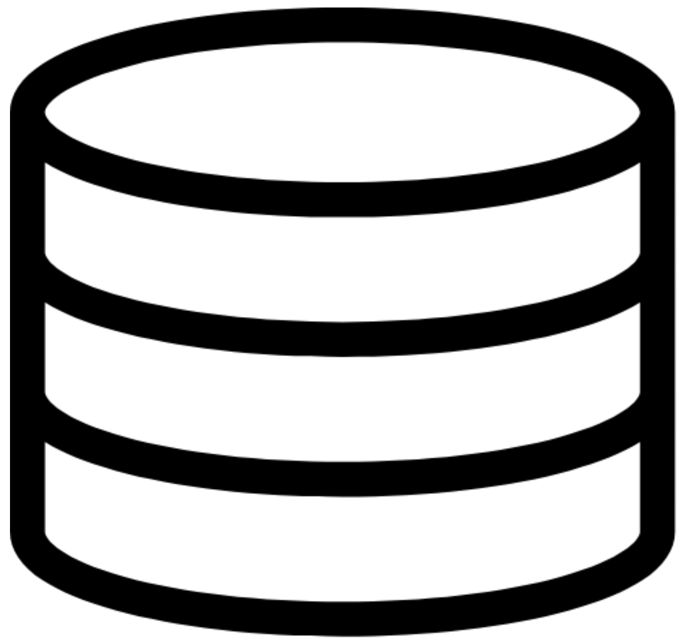
More realistically:

learn learn learn

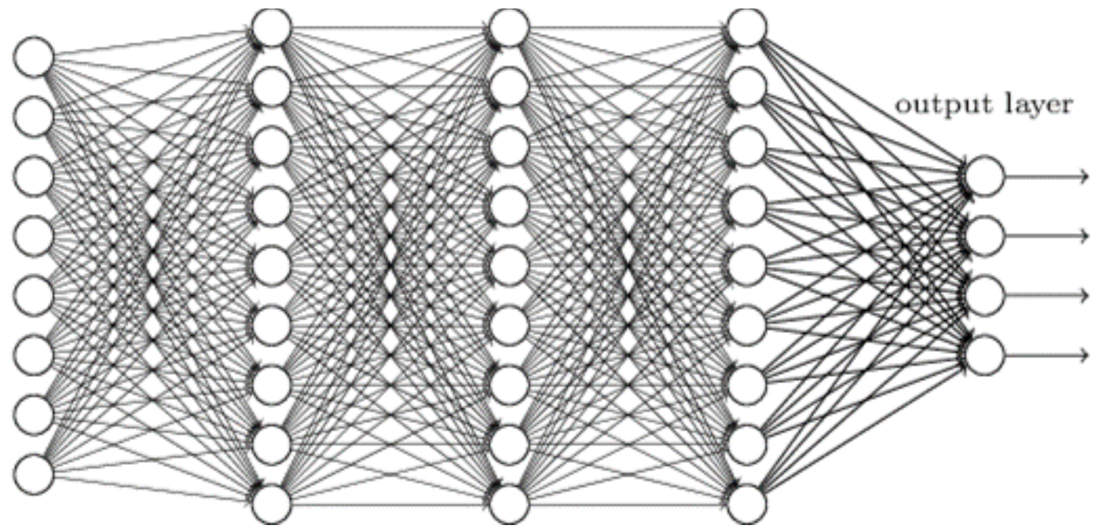
time

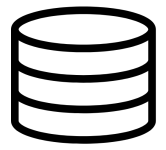
Deep networks + large datasets = 





In many practical situations:
Learn new task with only a **few** datapoints

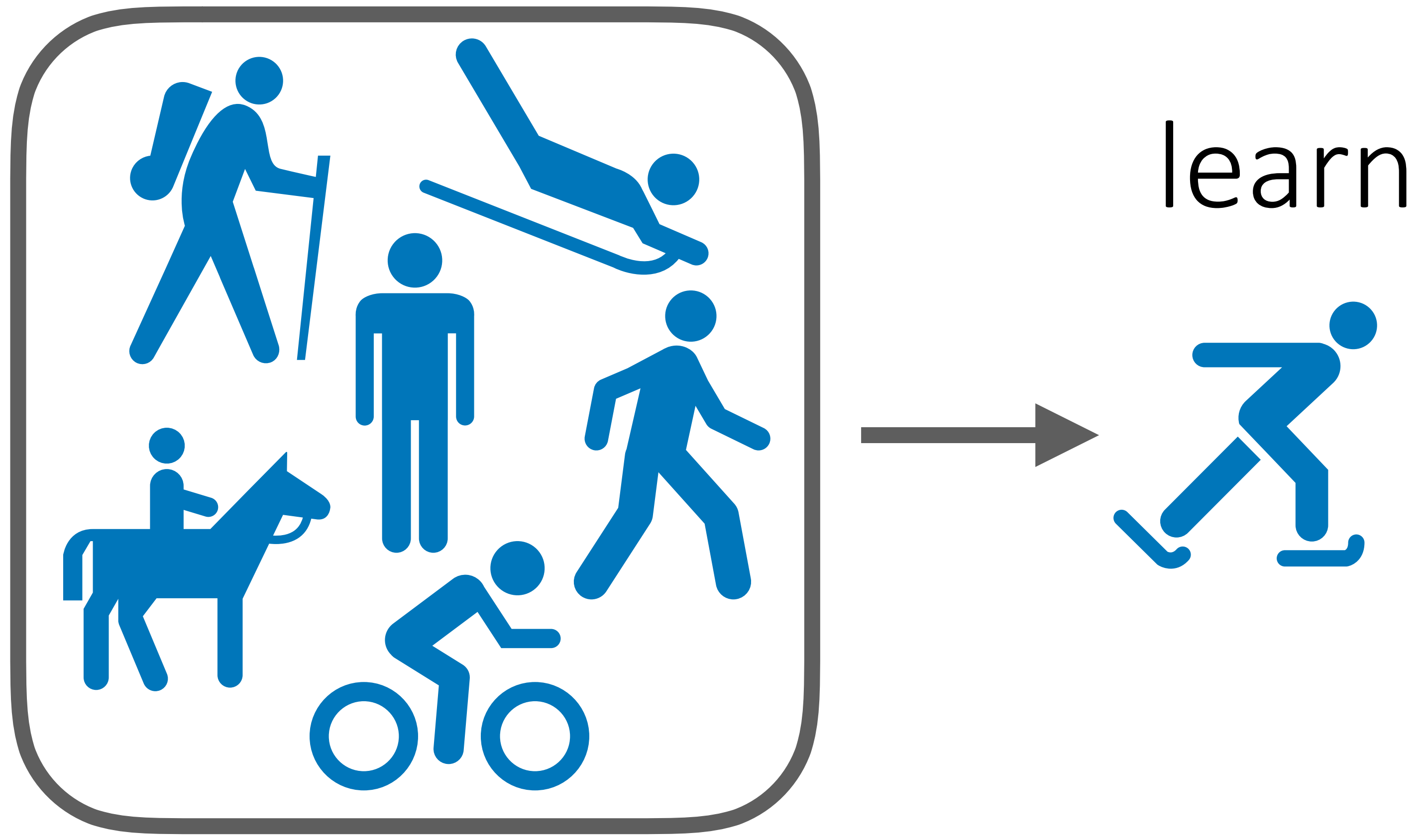




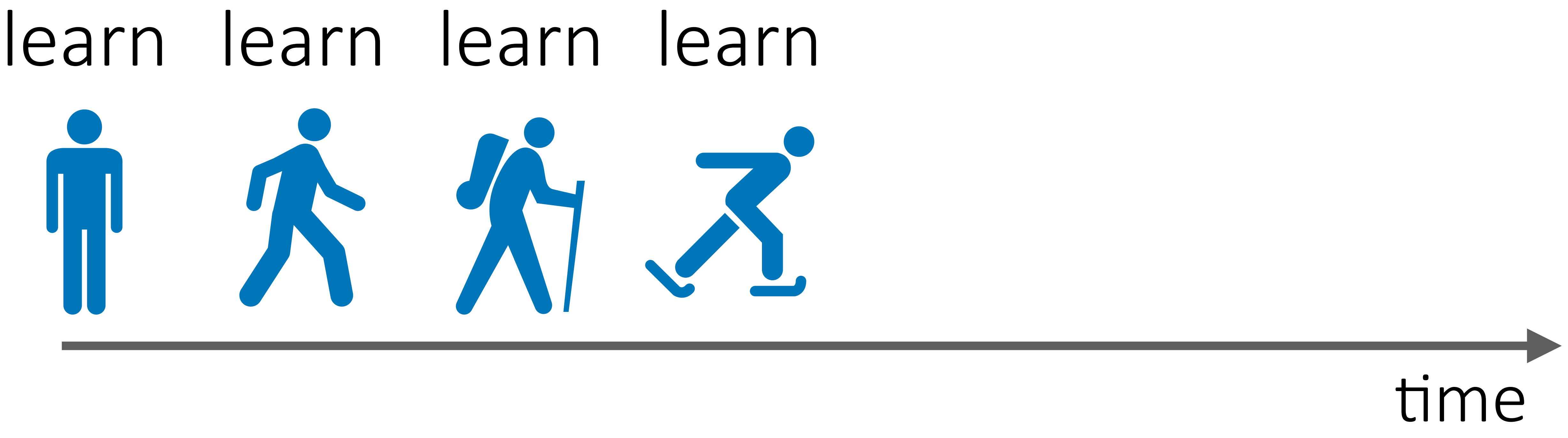
Meta-Learning

(Schmidhuber et al. '87, Bengio et al. '92)

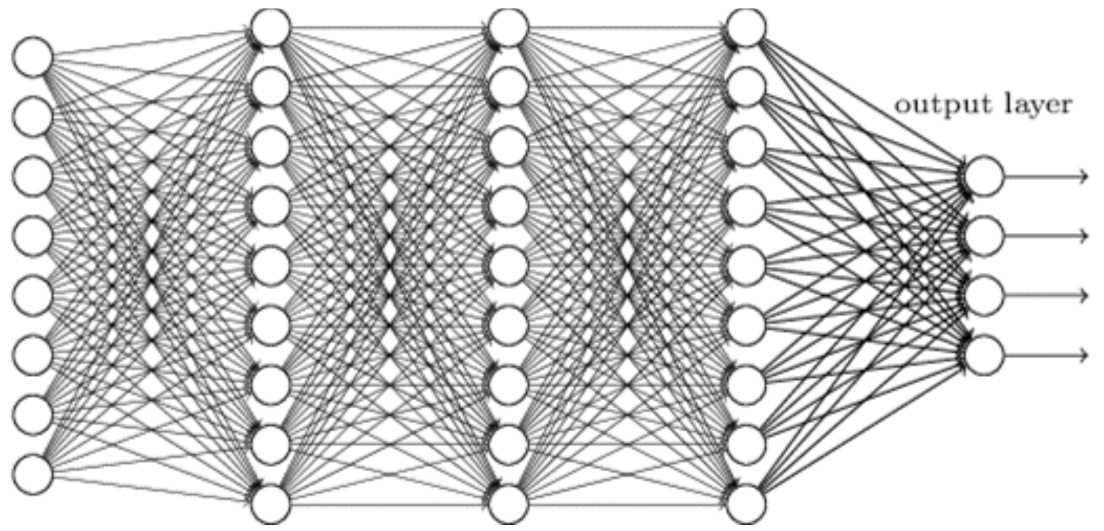
Given i.i.d. task distribution,
learn a new task efficiently

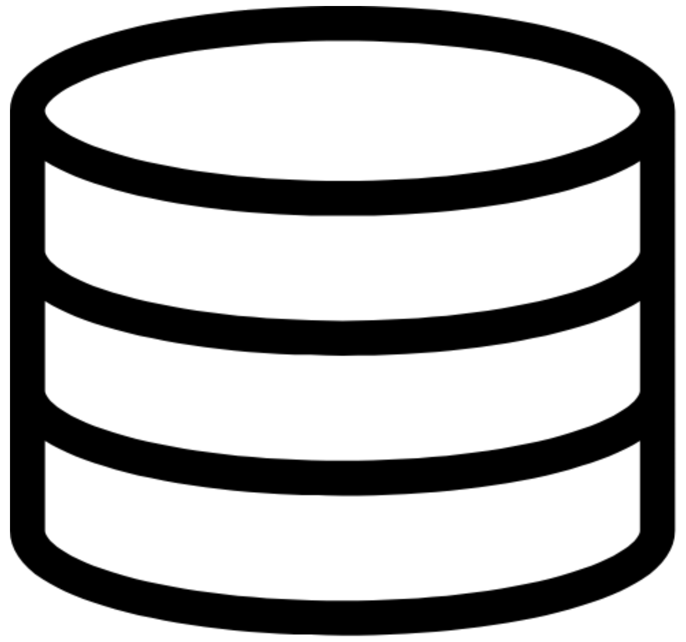


More realistically:

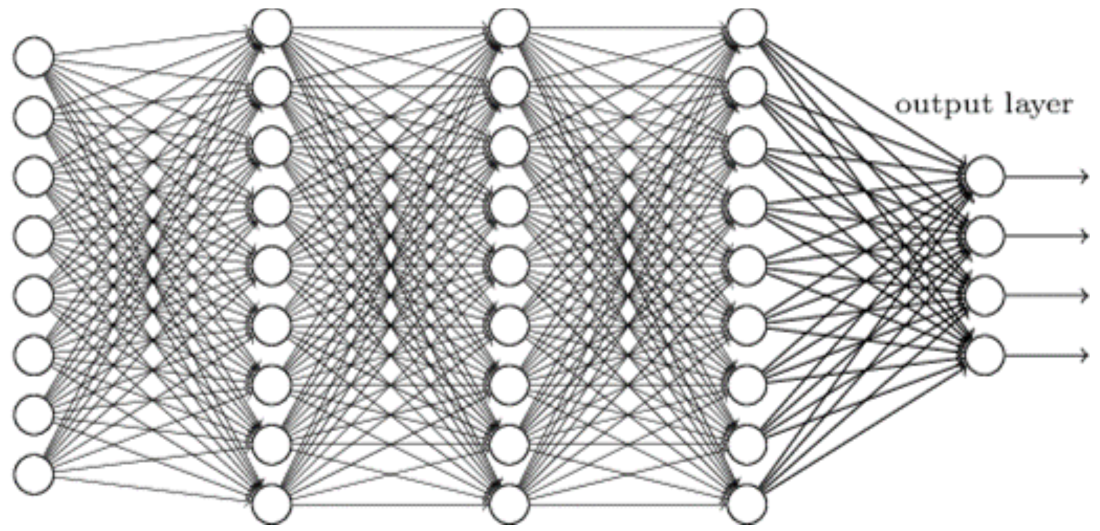


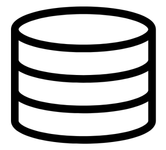
Deep networks + large datasets = 😍





In many practical situations:
Learn new task with only a **few** datapoints



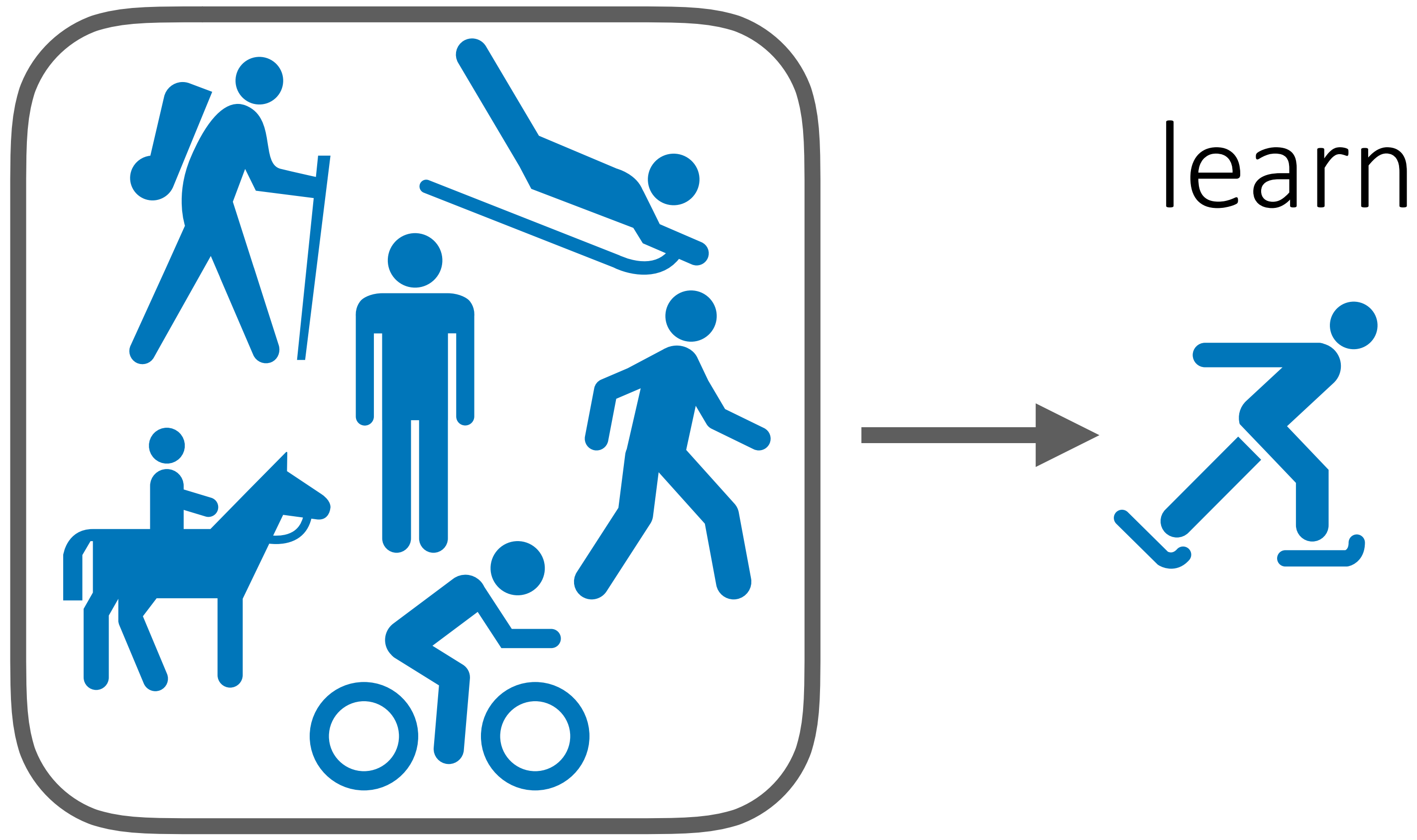


Meta-Learning

(Schmidhuber et al. ’87, Bengio et al. ’92)

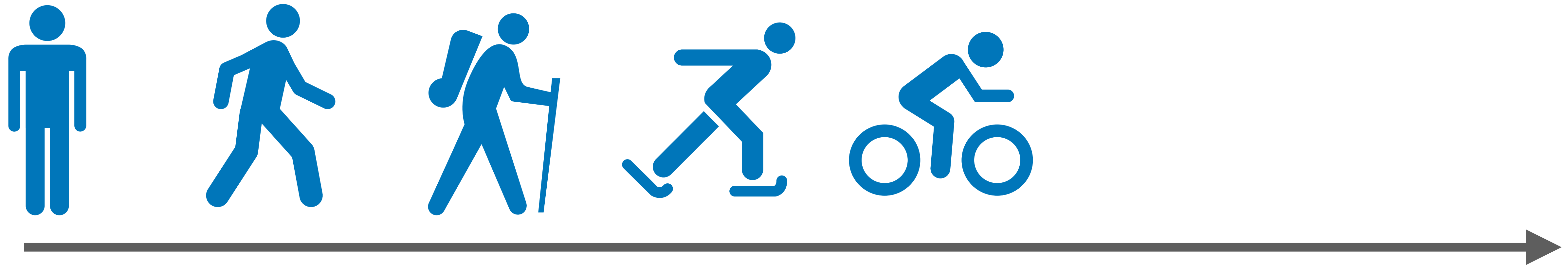
Given i.i.d. task distribution,
learn a new task efficiently

learn



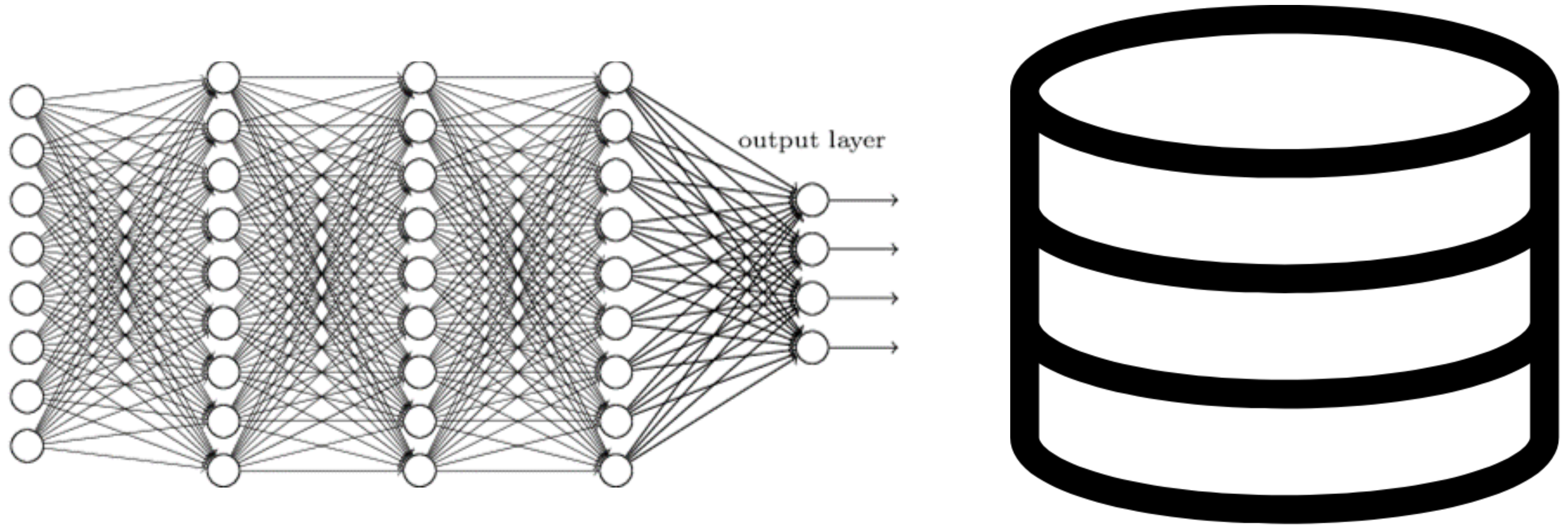
More realistically:

learn learn learn learn learn

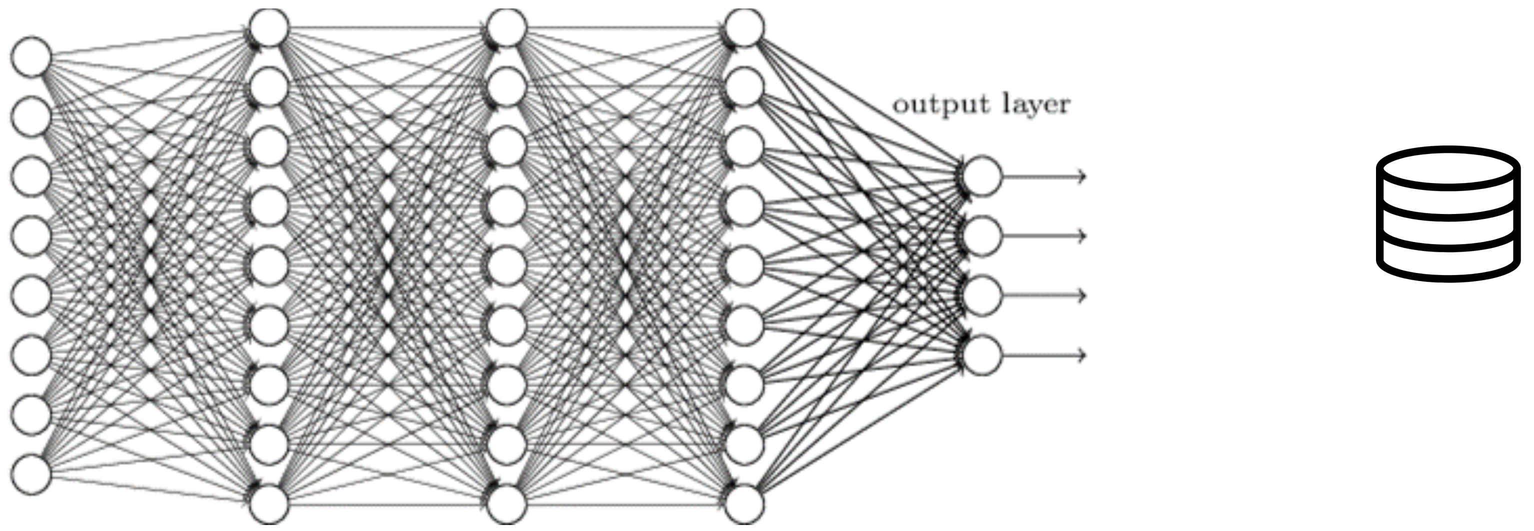


time

Deep networks + large datasets =



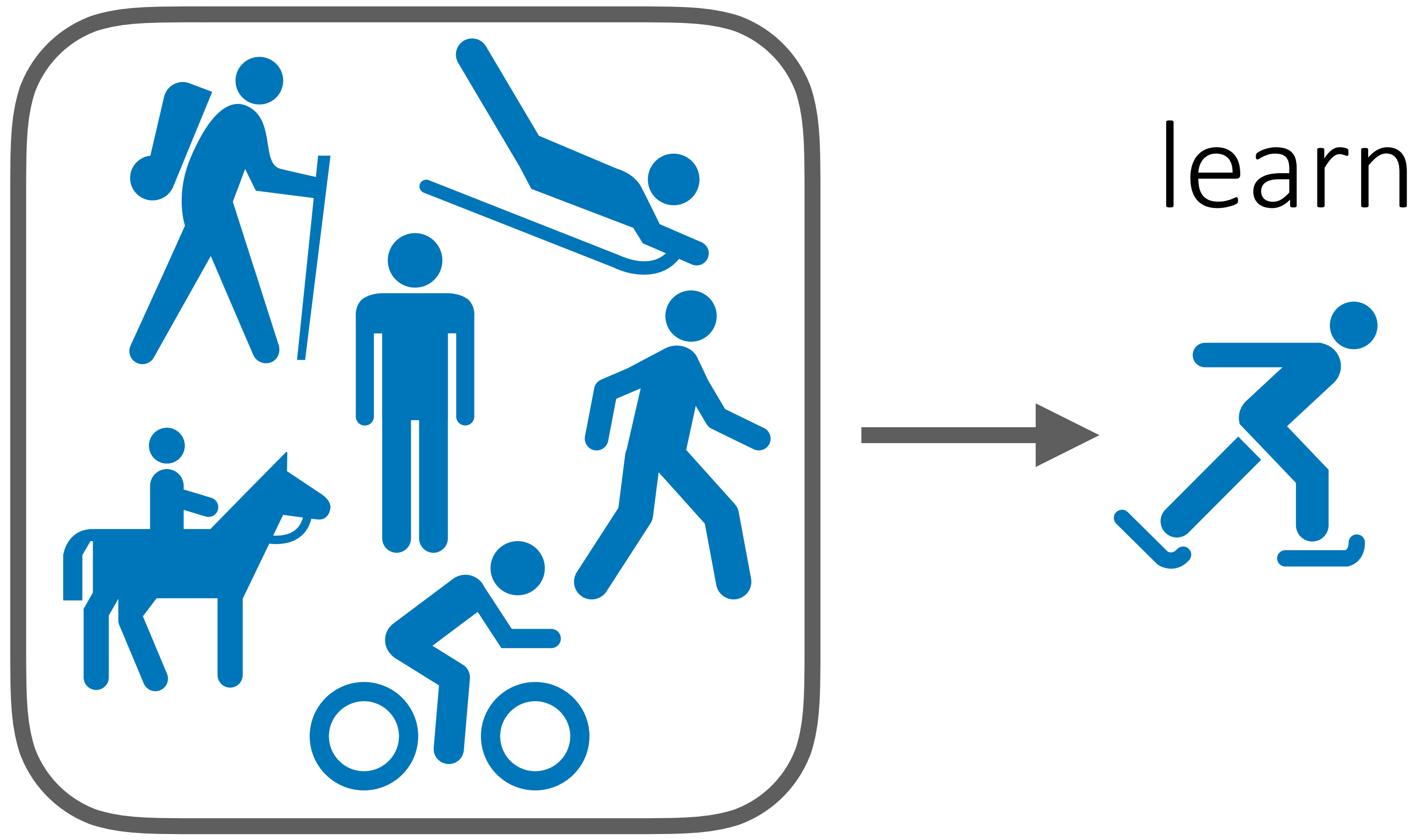
In many practical situations:
Learn new task with only a **few** datapoints



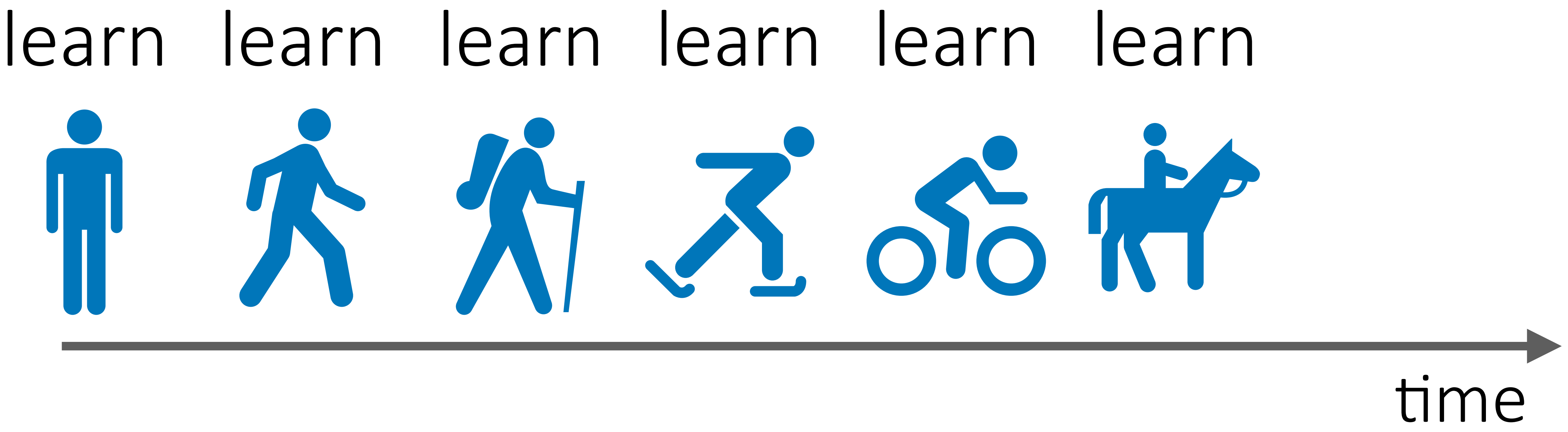
Meta-Learning

(Schmidhuber et al. '87, Bengio et al. '92)

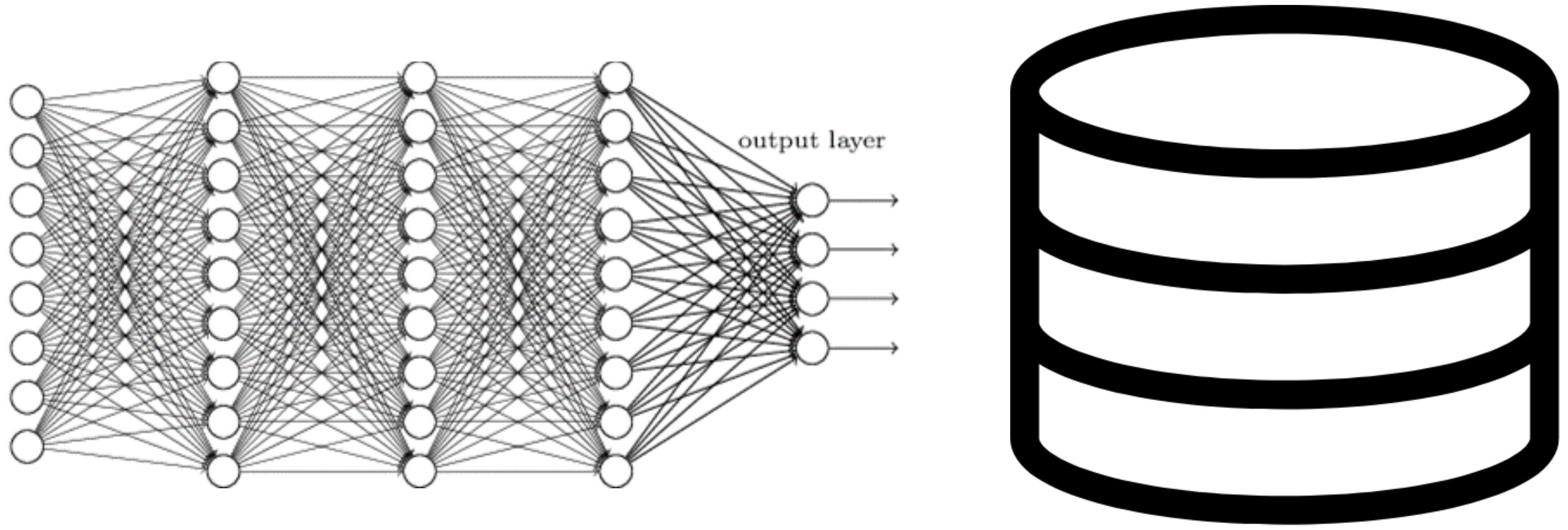
Given i.i.d. task distribution,
learn a new task efficiently



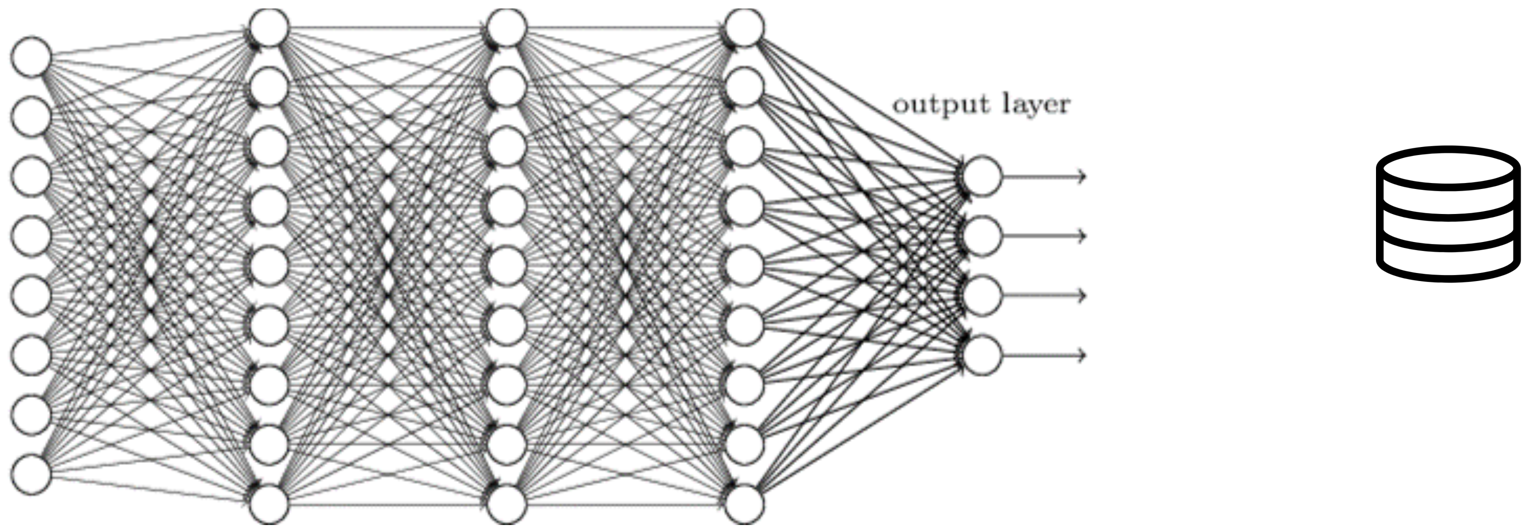
More realistically:



Deep networks + large datasets = 



In many practical situations:
Learn new task with only a **few** datapoints

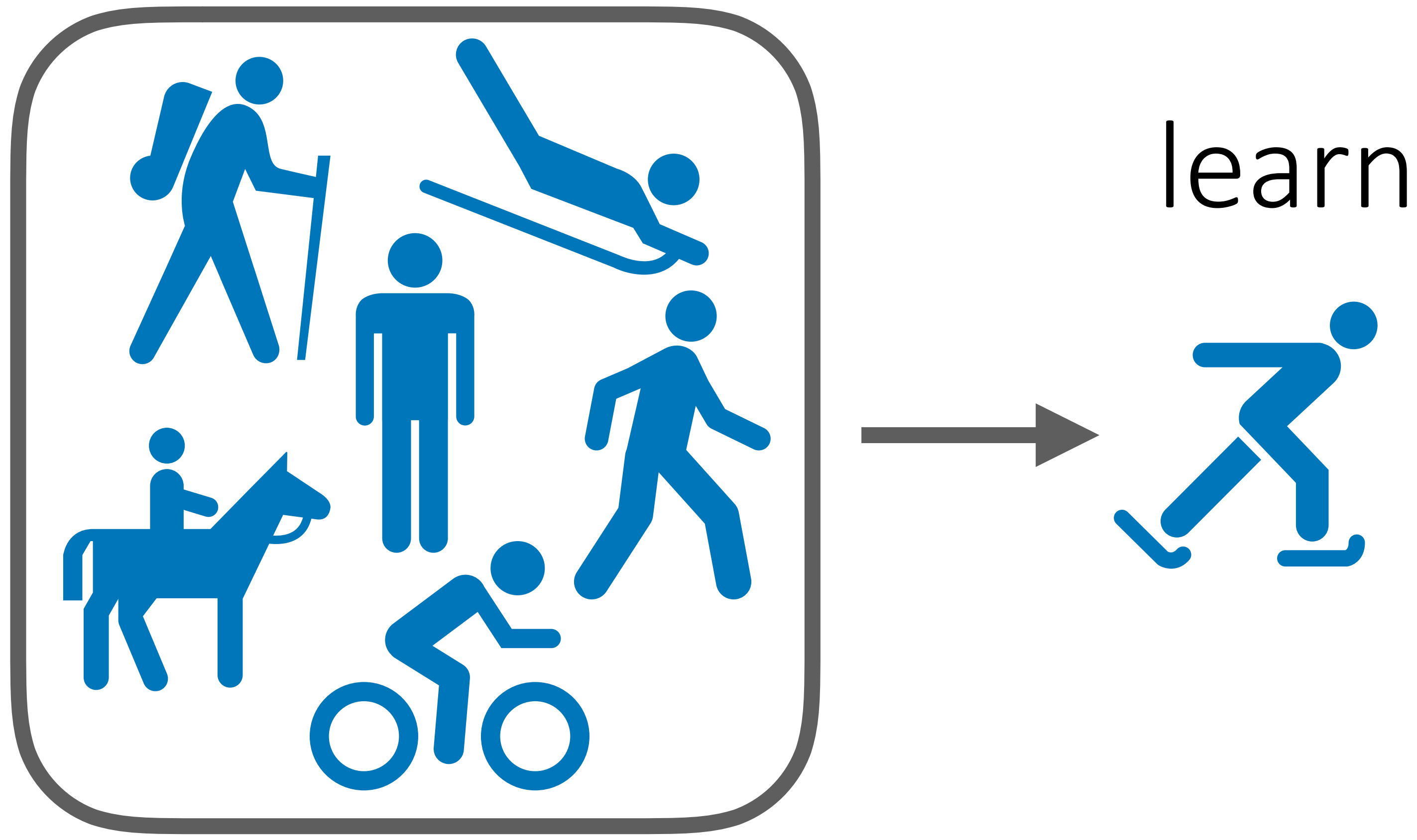


Meta-Learning

(Schmidhuber et al. ’87, Bengio et al. ’92)

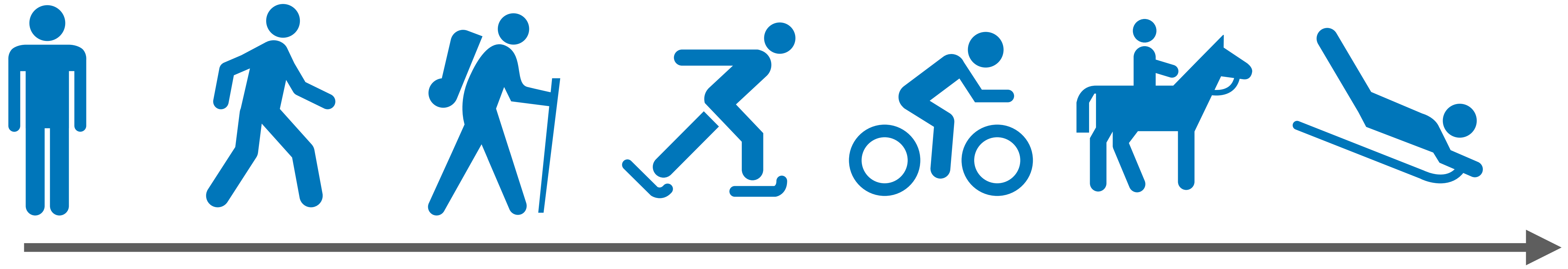
Given i.i.d. task distribution,
learn a new task efficiently

learn



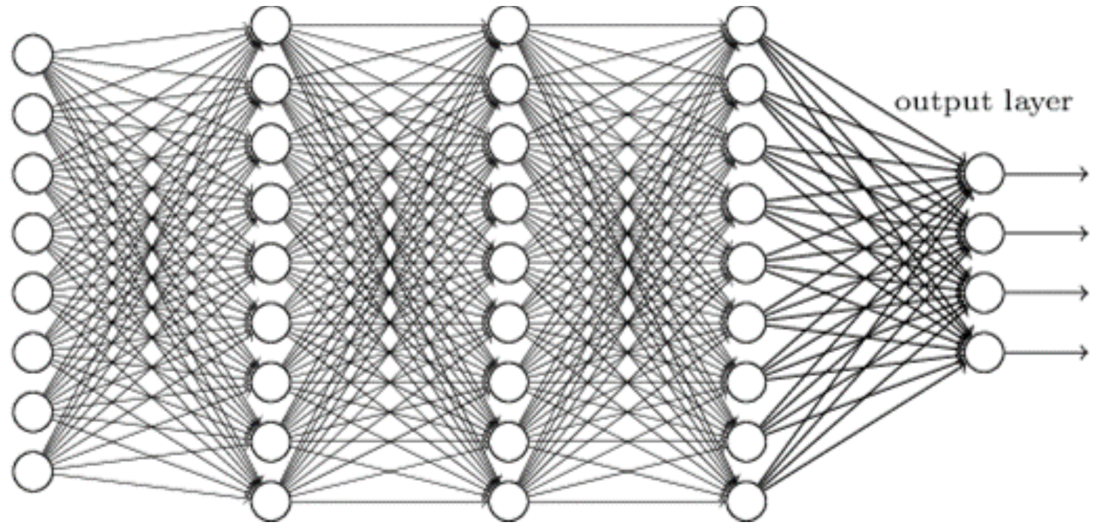
More realistically:

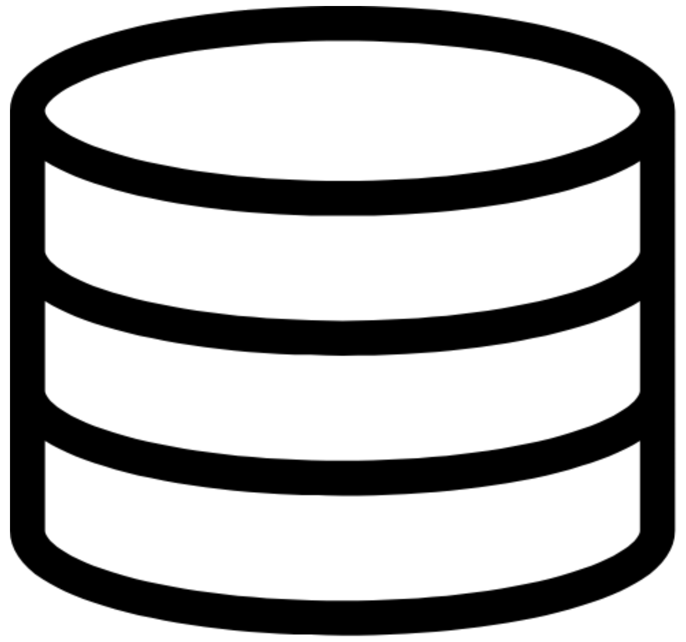
learn learn learn learn learn learn learn



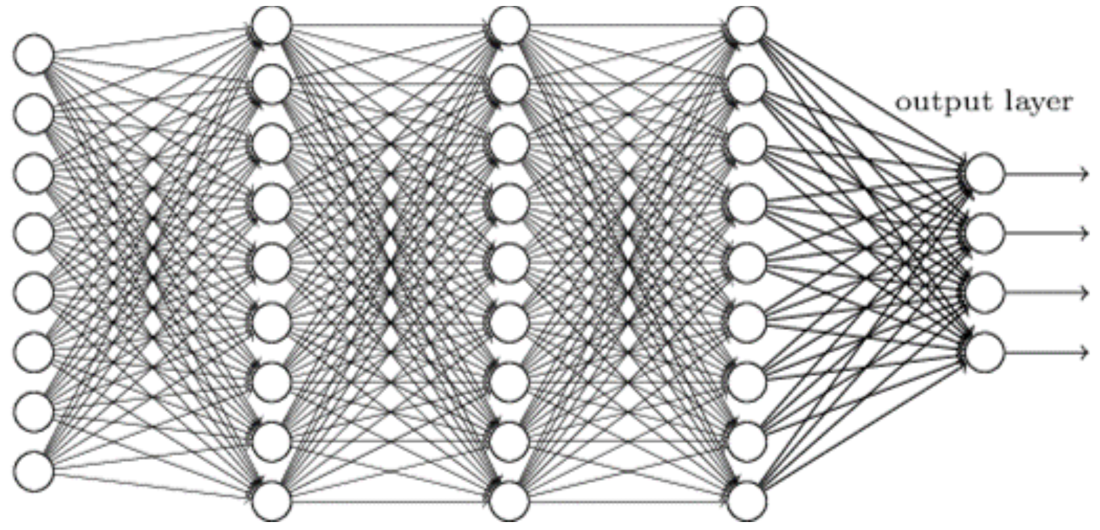
time

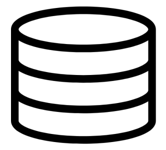
Deep networks + large datasets = 





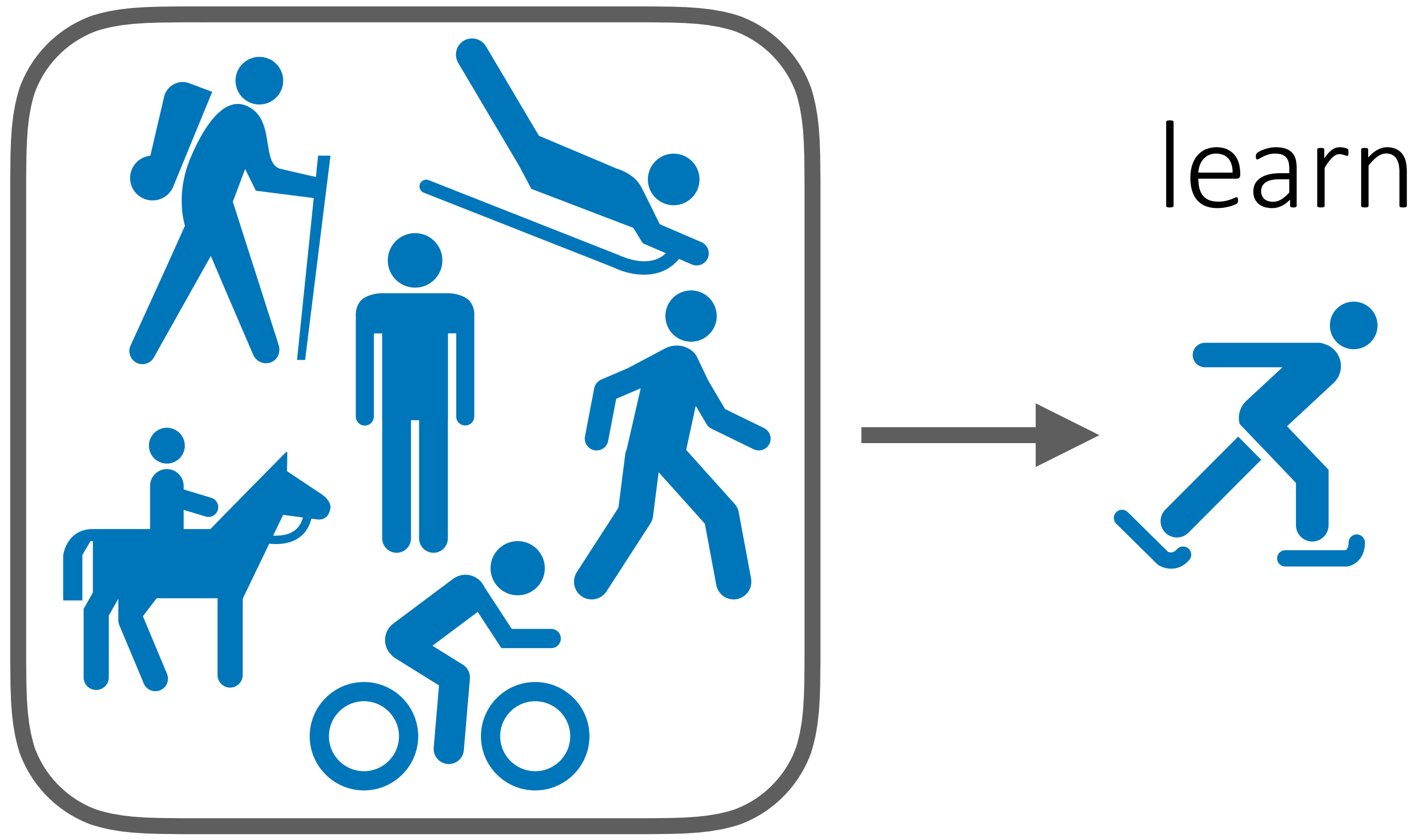
In many practical situations:
Learn new task with only a **few** datapoints



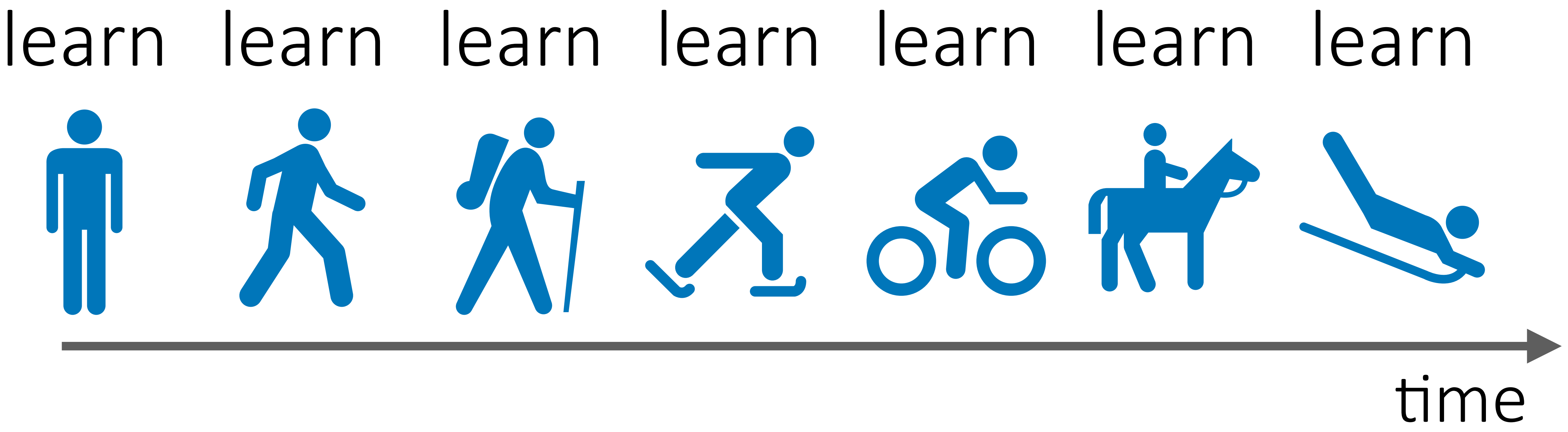


Meta-Learning
(Schmidhuber et al. ’87, Bengio et al. ’92)

Given i.i.d. task distribution,
learn a new task efficiently



More realistically:

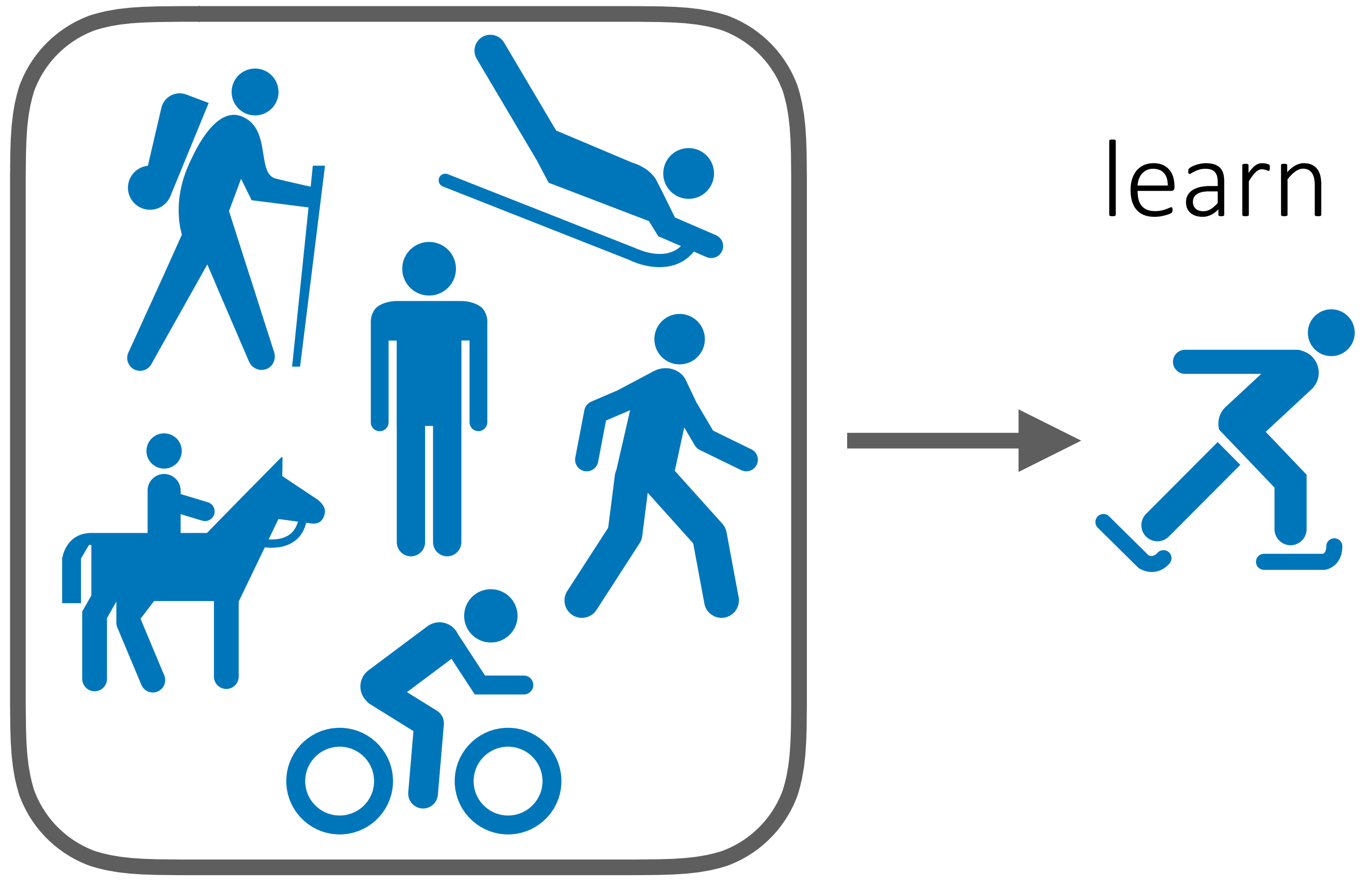


slow learning → rapid learning

Meta-Learning

(Schmidhuber et al. ’87, Bengio et al. ’92)

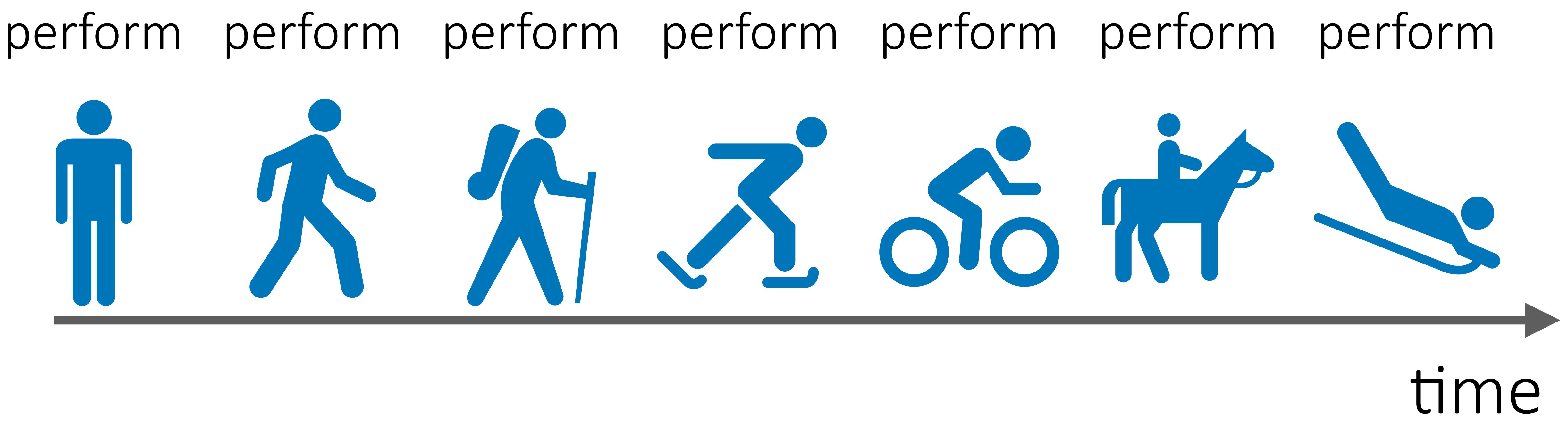
Given i.i.d. task distribution,
learn a new task efficiently



Online Learning

(Hannan ’57, Zinkevich ’03)

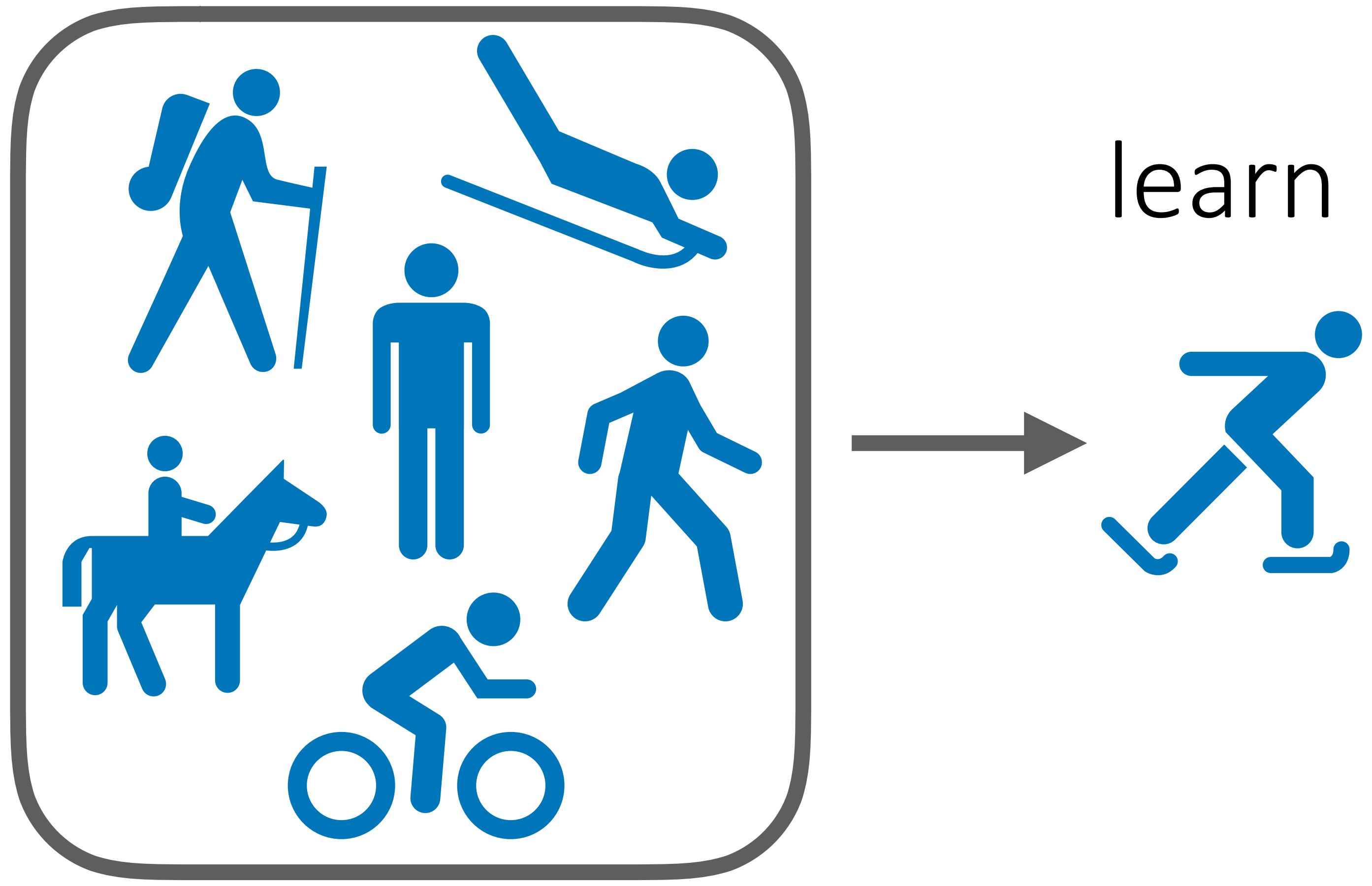
Perform sequence of tasks
while minimizing static regret.



Meta-Learning

(Schmidhuber et al. ’87, Bengio et al. ’92)

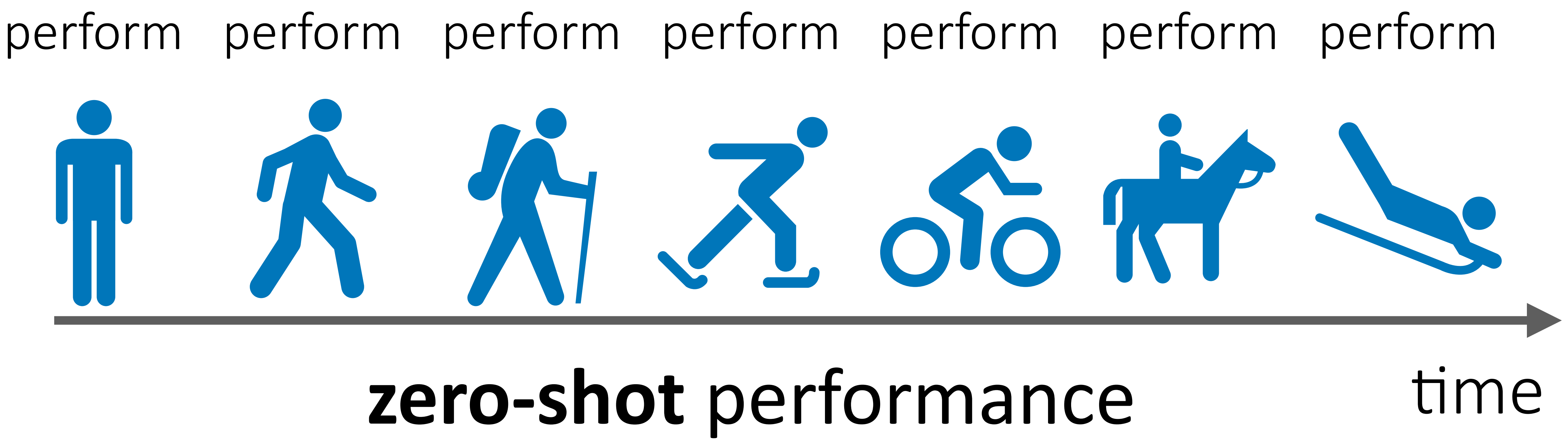
Given i.i.d. task distribution,
learn a new task efficiently



Online Learning

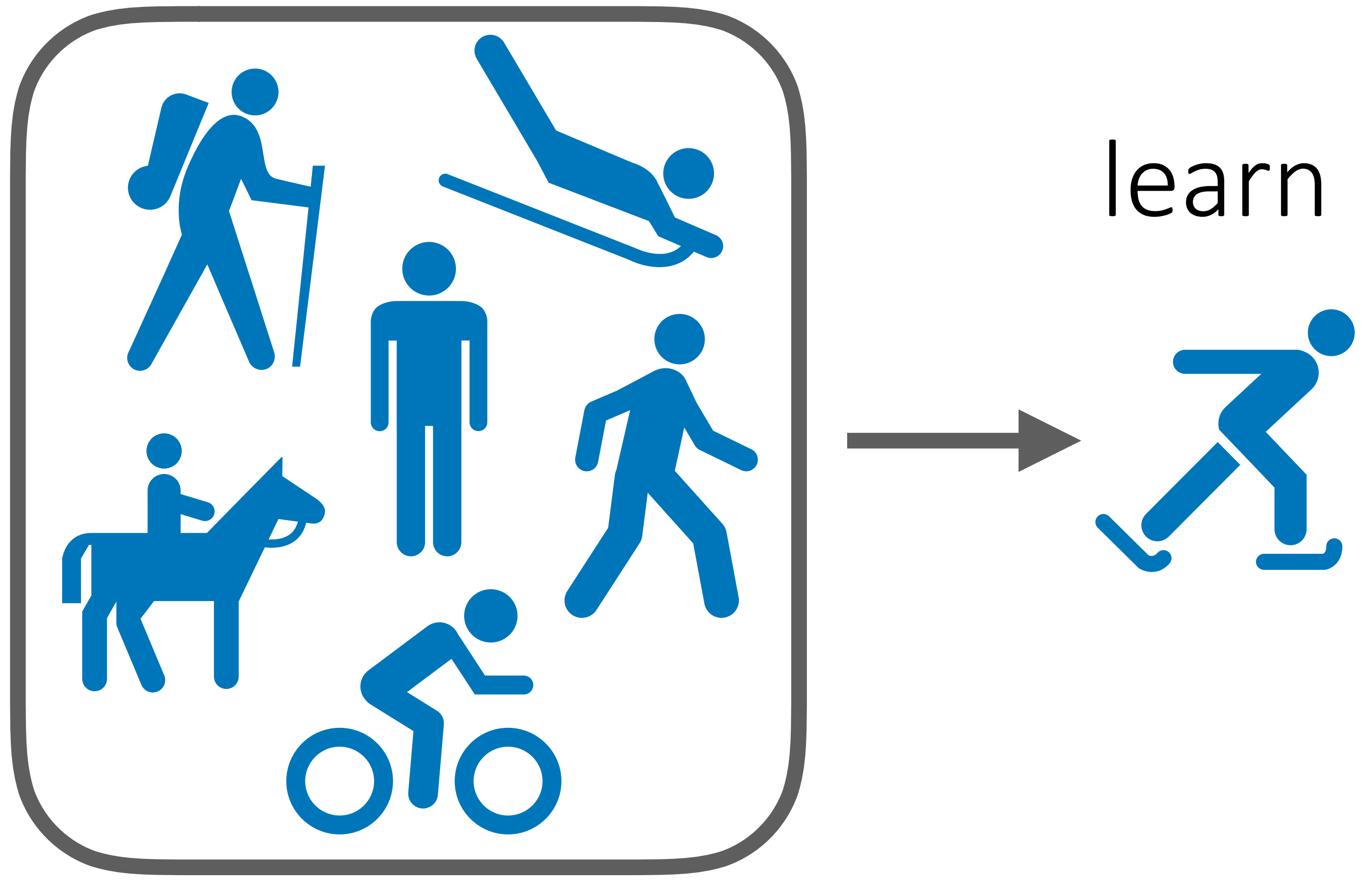
(Hannan ’57, Zinkevich ’03)

Perform sequence of tasks
while minimizing static regret.



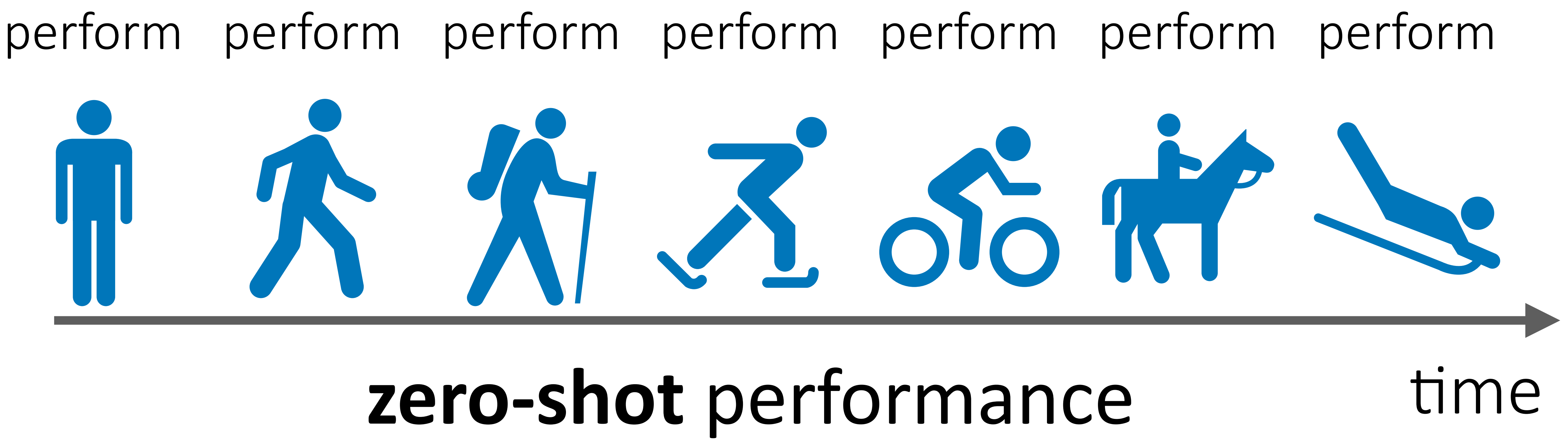
Meta-Learning
(Schmidhuber et al. ’87, Bengio et al. ’92)

Given i.i.d. task distribution, learn a new task efficiently



Online Learning
(Hannan ’57, Zinkevich ’03)

Perform sequence of tasks while minimizing static regret.



zero-shot performance

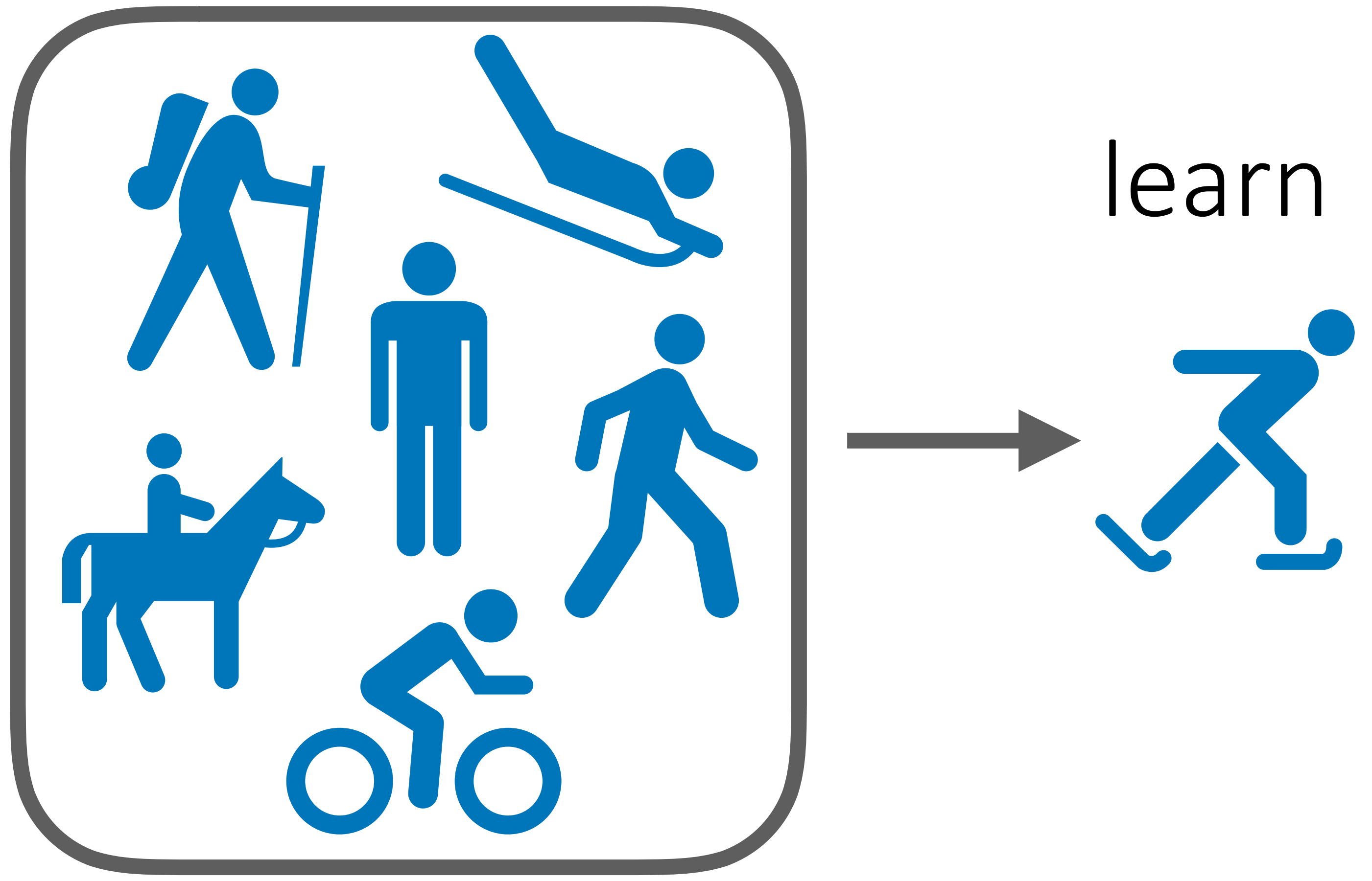
Online Meta-Learning
(this work)

Efficiently learn a sequence of tasks from a non-stationary distribution.

Meta-Learning

(Schmidhuber et al. ’87, Bengio et al. ’92)

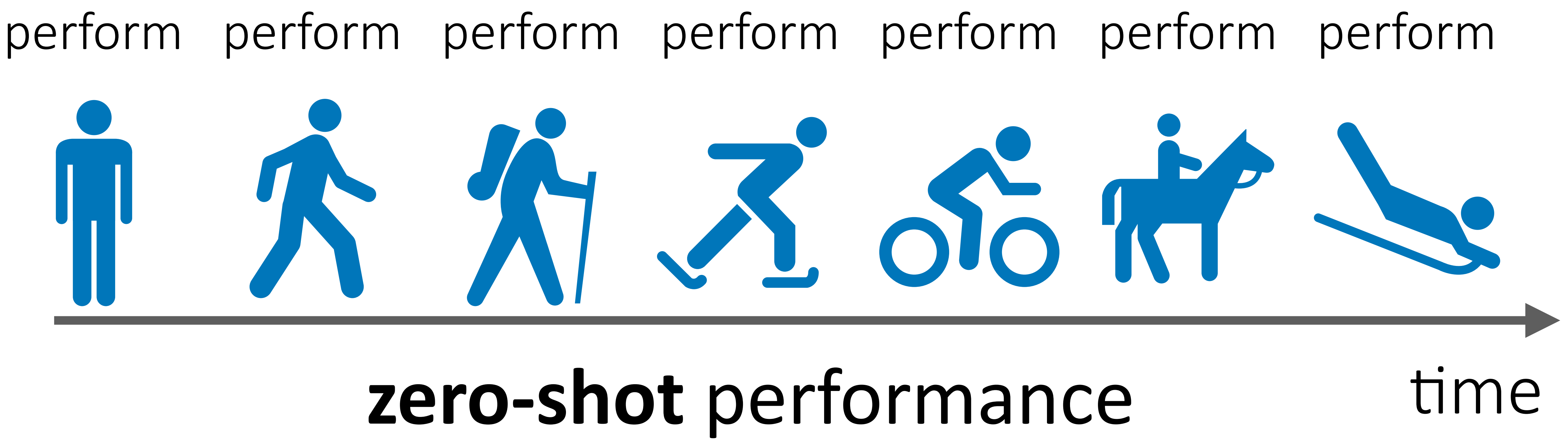
Given i.i.d. task distribution,
learn a new task efficiently



Online Learning

(Hannan ’57, Zinkevich ’03)

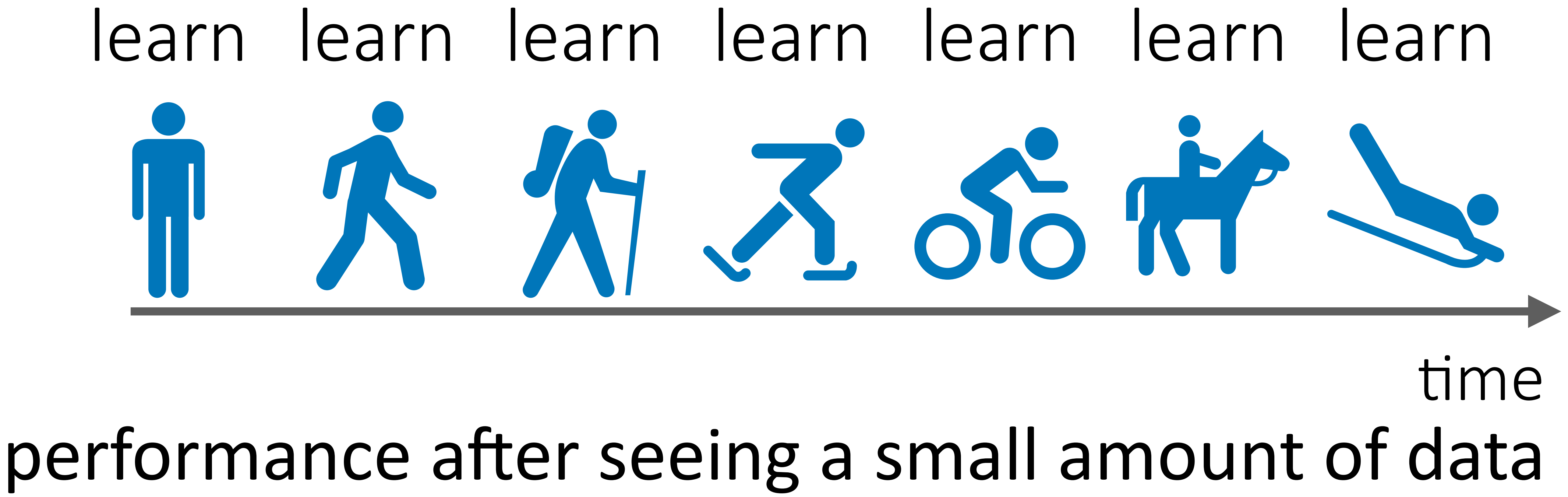
Perform sequence of tasks
while minimizing static regret.



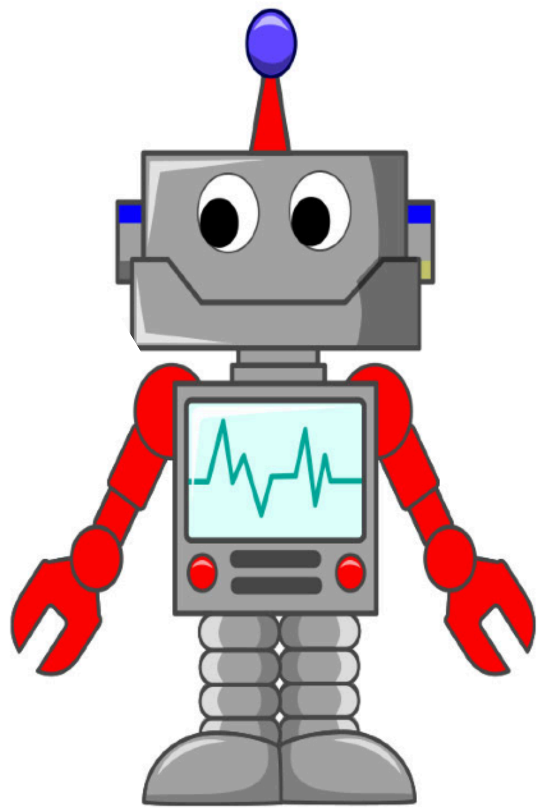
Online Meta-Learning

(this work)

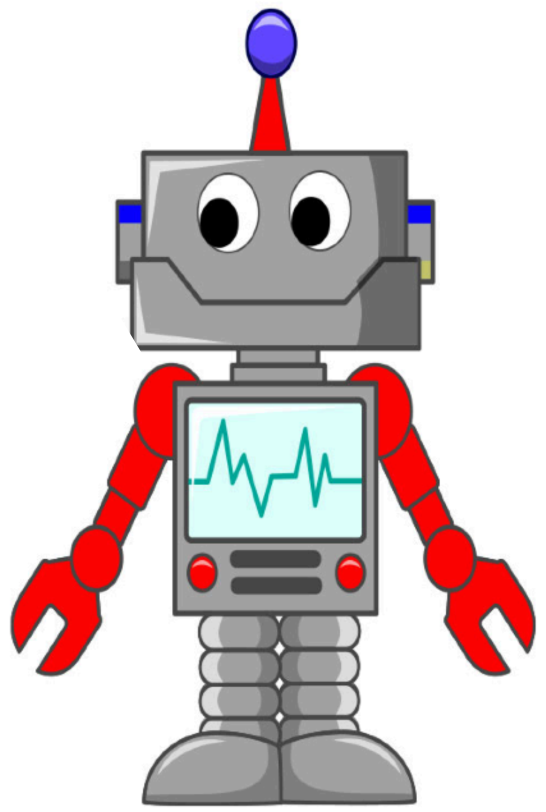
Efficiently learn a sequence of tasks
from a non-stationary distribution.



The Online Meta-Learning Setting



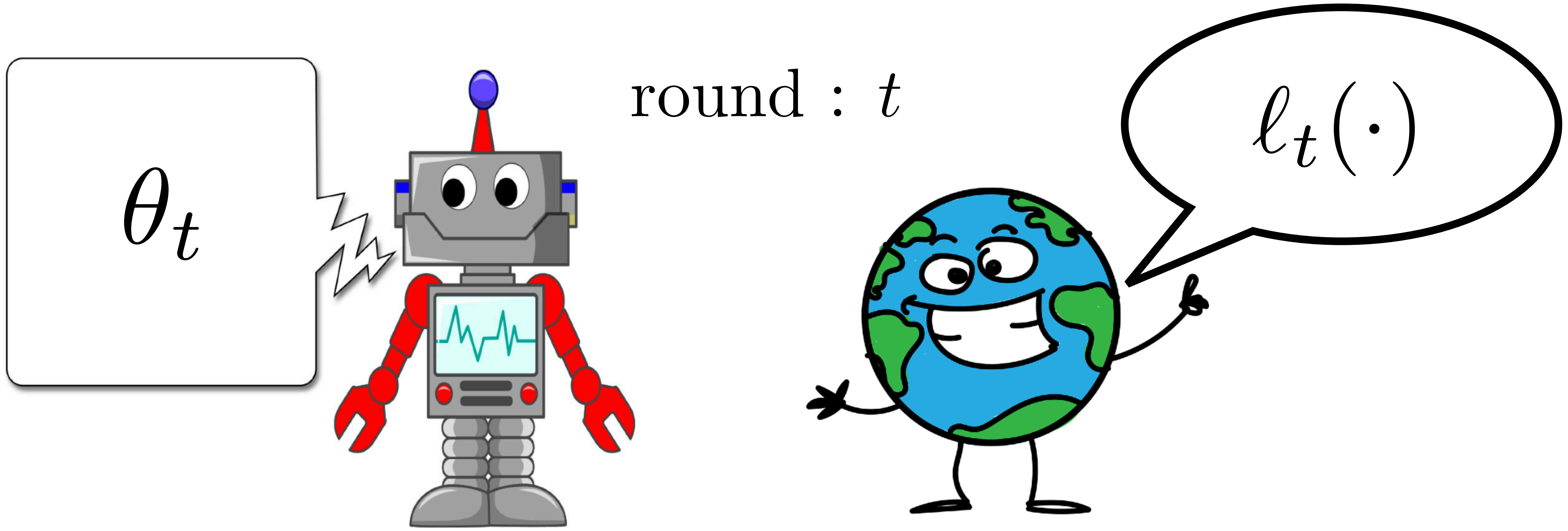
The Online Meta-Learning Setting



Space of parameters $\theta \in \Theta \subseteq \mathbb{R}^d$ and loss functions $\ell : \Theta \to \mathbb{R}$

For round $t \in \{1, 2, \ldots \infty\}$:

The Online Meta-Learning Setting

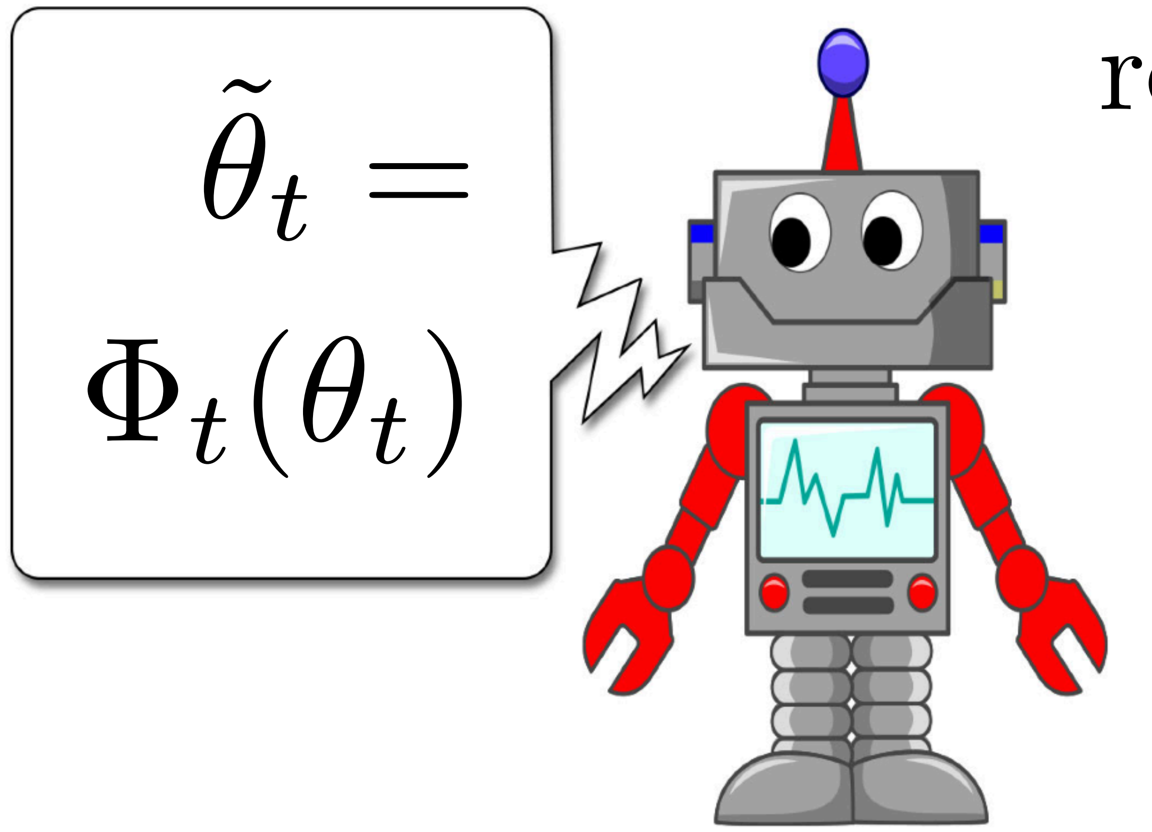


Space of parameters $\theta \in \Theta \subseteq \mathbb{R}^d$ and loss functions $\ell : \Theta \to \mathbb{R}$

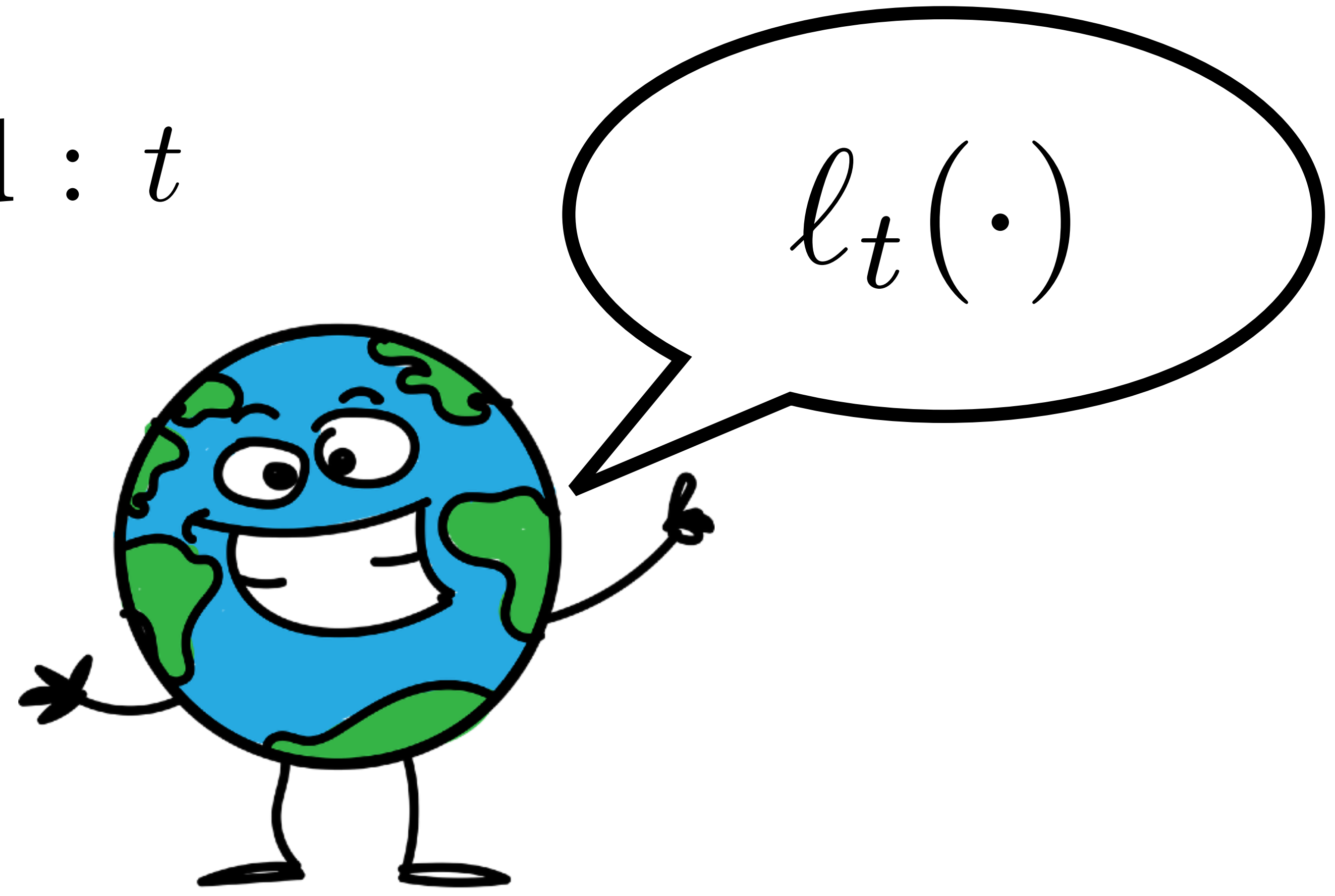
For round $t \in \{1, 2, \ldots \infty\}$:

1. World picks a loss function $\ell_t(\cdot)$
2. Agent should pick θ_t without knowledge of ℓ_t

The Online Meta-Learning Setting



round : t

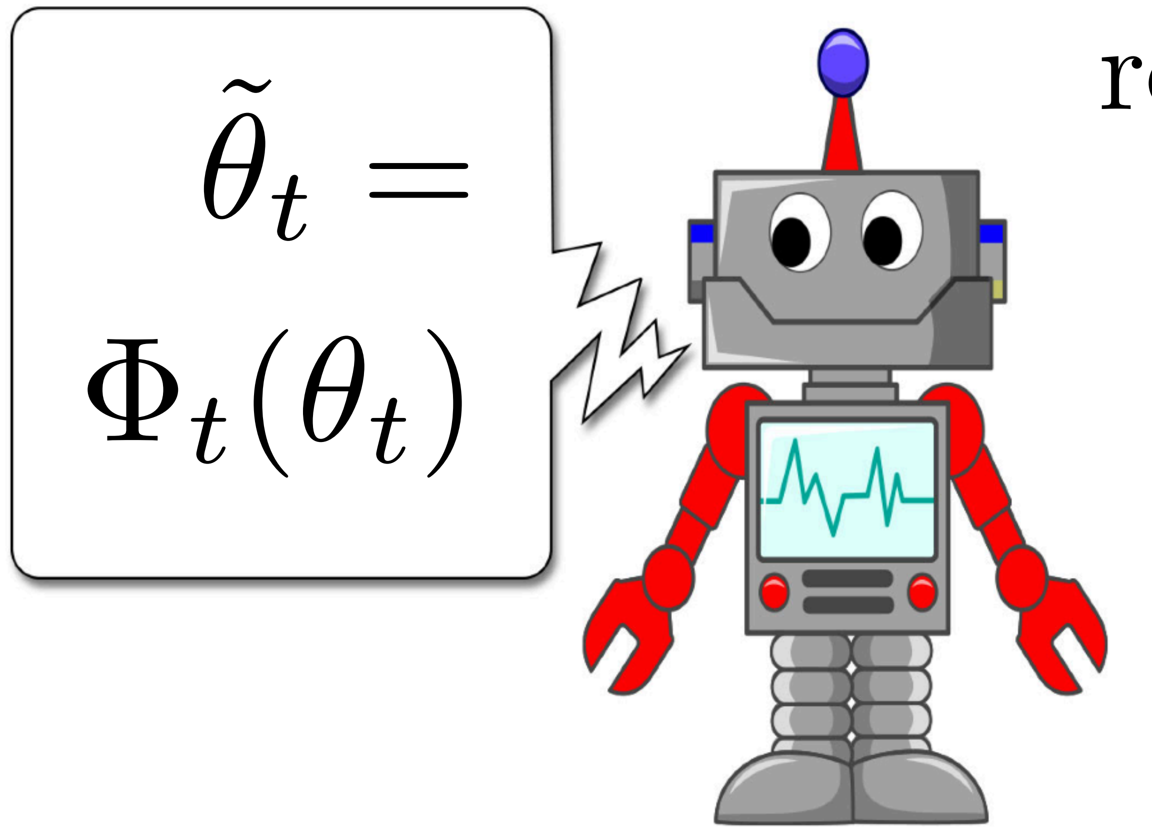


Space of parameters $\theta \in \Theta \subseteq \mathbb{R}^d$ and loss functions $\ell : \Theta \to \mathbb{R}$

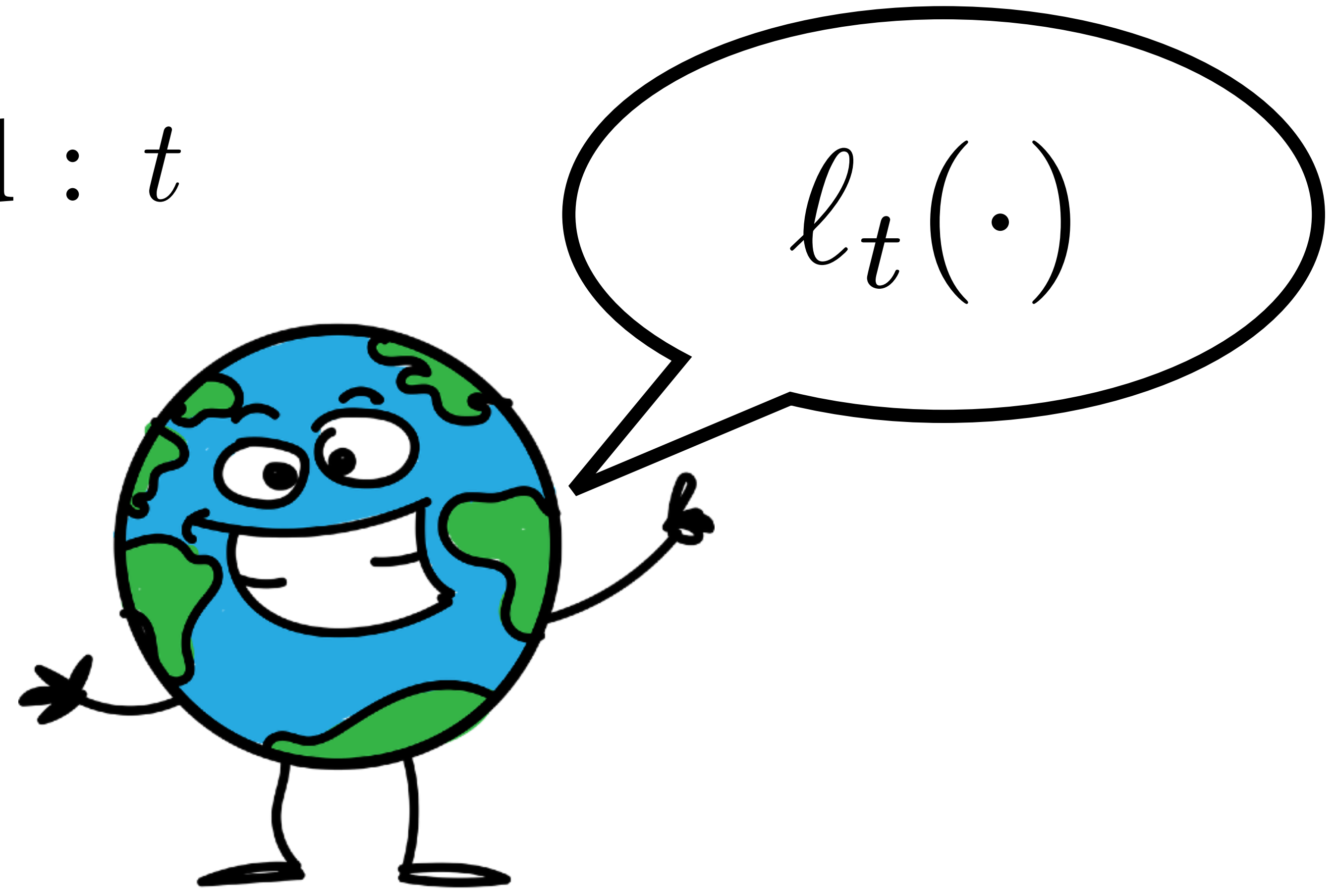
For round $t \in \{1, 2, \ldots \infty\}$:

1. World picks a loss function $\ell_t(\cdot)$
2. Agent should pick θ_t without knowledge of ℓ_t
3. Agent uses update procedure $\Phi_t : \Theta \to \Theta$, and obtains $\tilde{\theta}_t = \Phi_t(\theta_t)$

The Online Meta-Learning Setting



round : t



Space of parameters $\theta \in \Theta \subseteq \mathbb{R}^d$ and loss functions $\ell : \Theta \rightarrow \mathbb{R}$

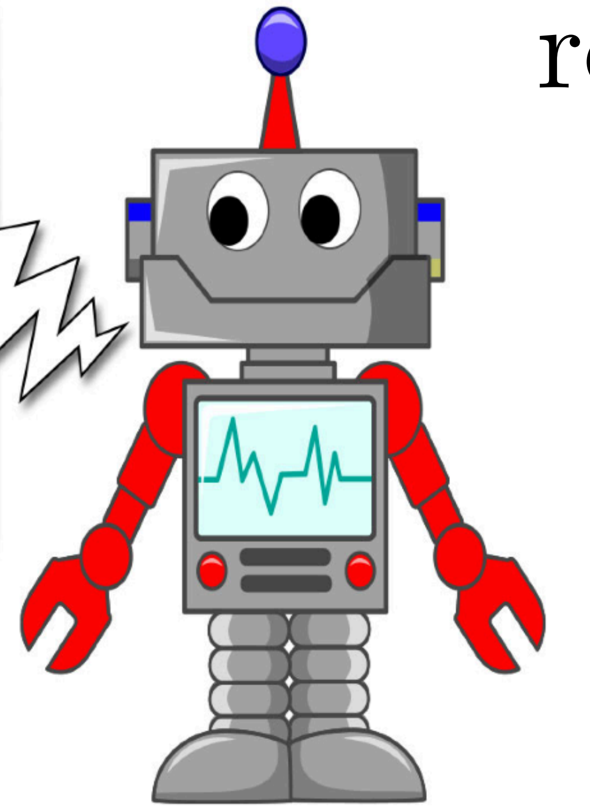
For round $t \in \{1, 2, \ldots \infty\}$:

$$\tilde{\theta}_t = \theta_t - \alpha \nabla \hat{\ell}_t(\theta_t)$$

1. World picks a loss function $\ell_t(\cdot)$
2. Agent should pick θ_t without knowledge of ℓ_t
3. Agent uses update procedure $\Phi_t : \Theta \rightarrow \Theta$, and obtains $\tilde{\theta}_t = \Phi_t(\theta_t)$

The Online Meta-Learning Setting

$\tilde{\theta}_t = \Phi_t(\theta_t)$



round : t

$\ell_t(\cdot)$

Space of parameters $\theta \in \Theta \subseteq \mathbb{R}^d$ and loss functions $\ell : \Theta \rightarrow \mathbb{R}$

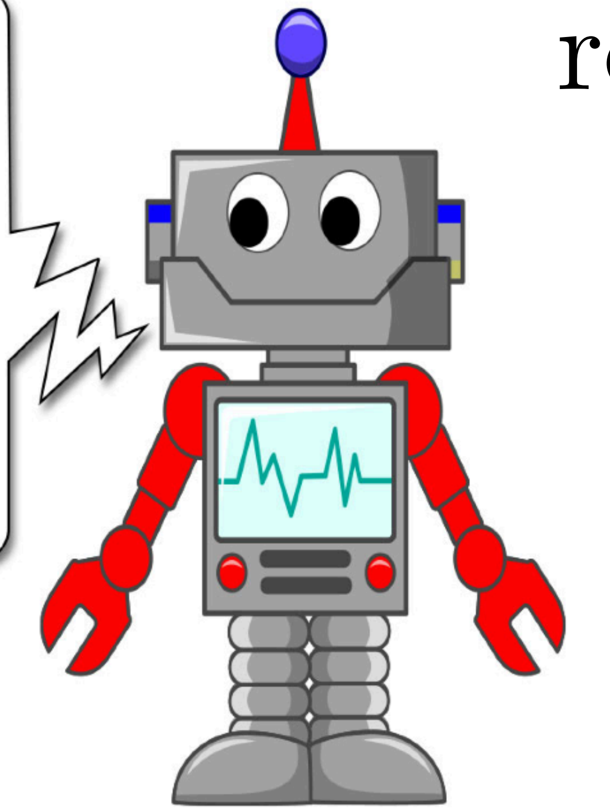
For round $t \in \{1, 2, \ldots \infty\}$:

$\tilde{\theta}_t = \theta_t - \alpha \nabla \hat{\ell}_t(\theta_t)$

1. World picks a loss function $\ell_t(\cdot)$
2. Agent should pick θ_t without knowledge of ℓ_t
3. Agent uses update procedure $\Phi_t : \Theta \rightarrow \Theta$, and obtains $\tilde{\theta}_t = \Phi_t(\theta_t)$
4. Agent suffers $\ell_t(\tilde{\theta}_t)$ for the round

The Online Meta-Learning Setting

$\tilde{\theta}_t = \Phi_t(\theta_t)$



round : t

$\ell_t(\cdot)$

Space of parameters $\theta \in \Theta \subseteq \mathbb{R}^d$ and loss functions $\ell : \Theta \rightarrow \mathbb{R}$

For round $t \in \{1, 2, \ldots \infty\}$:

$\tilde{\theta}_t = \theta_t - \alpha \nabla \hat{\ell}_t(\theta_t)$

1. World picks a loss function $\ell_t(\cdot)$
2. Agent should pick θ_t without knowledge of ℓ_t
3. Agent uses update procedure $\Phi_t : \Theta \rightarrow \Theta$, and obtains $\tilde{\theta}_t = \Phi_t(\theta_t)$
4. Agent suffers $\ell_t(\tilde{\theta}_t)$ for the round

Goal: Learning algorithm with sub-linear

Loss of algorithm

Loss of best algorithm in hindsight

$$\text{Regret}_T := \sum_{t=1}^{T} \ell_t(\Phi_t(\theta_t)) - \min_{\theta \in \Theta} \sum_{t=1}^{T} \ell_t(\Phi_t(\theta))$$

Follow the Meta-Leader (FTML) :

$$\theta_{t+1} = \arg\min_{\theta} \sum_{t=1}^{T} \ell_t(\Phi_t(\theta))$$

Can be implemented with MAML

Follow the Meta-Leader (FTML) : $$\theta_{t+1} = \arg\min_{\theta} \sum_{t=1}^{T} \ell_t(\Phi_t(\theta))$$

Can be implemented with MAML

Theorem (Informal): If $\{\ell_t(\cdot), \hat{\ell}_t(\cdot)\}\ \forall t$ are C^2-smooth and strongly convex, the sequence of models $\{\theta_1, \theta_2, \ldots, \theta_T\}$ returned by FTML has the property:

$$\text{Regret}_T := \sum_{t=1}^{T} \ell_t(\Phi_t(\theta_t)) - \min_{\theta \in \Theta} \sum_{t=1}^{T} \ell_t(\Phi_t(\theta)) = O(\log T)$$

Follow the Meta-Leader (FTML) : $$\theta_{t+1} = \arg\min_{\theta} \sum_{t=1}^{T} \ell_t(\Phi_t(\theta))$$

Can be implemented with MAML

Theorem (Informal): If $\{\ell_t(\cdot), \hat{\ell}_t(\cdot)\}\ \forall t$ are C^2-smooth and strongly convex, the sequence of models $\{\theta_1, \theta_2, \ldots, \theta_T\}$ returned by FTML has the property:

$$\text{Regret}_T := \sum_{t=1}^{T} \ell_t(\Phi_t(\theta_t)) - \min_{\theta \in \Theta} \sum_{t=1}^{T} \ell_t(\Phi_t(\theta)) = O(\log T)$$

$$\implies \text{Avg. Regret} = \frac{\text{Regret}_T}{T} \to 0 \text{ as } T \to \infty$$

Learning in a sequential non-stationary setting, but still competitive with best meta-learner in hindsight!

FTML: practical instantiation of our approach, extending MAML[1]
meta-train on all data so far, fine-tune on current task

[1] Finn et al. ICML '17

FTML: practical instantiation of our approach, extending MAML[1]
meta-train on all data so far, fine-tune on current task

Experiment with **sequences of tasks**:

[1] Finn et al. ICML '17

FTML: practical instantiation of our approach, extending MAML[1]
meta-train on all data so far, fine-tune on current task

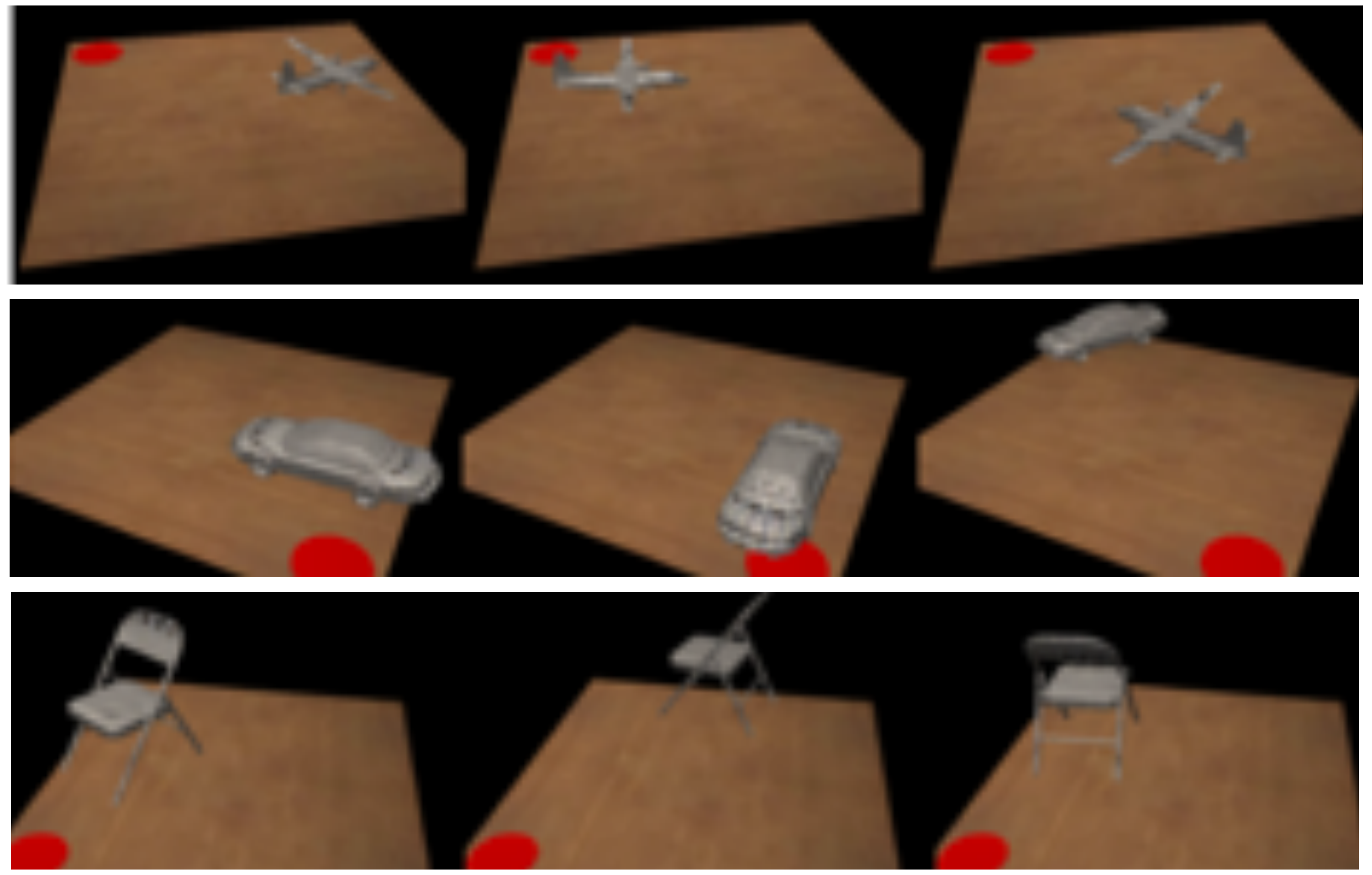
Experiment with **sequences of tasks**:

- Colored, rotated, scaled **MNIST**

[1] Finn et al. ICML '17

FTML: practical instantiation of our approach, extending MAML[1]
meta-train on all data so far, fine-tune on current task

Example pose prediction tasks



Experiment with **sequences of tasks**:

- Colored, rotated, scaled **MNIST**
- **3D object pose prediction**

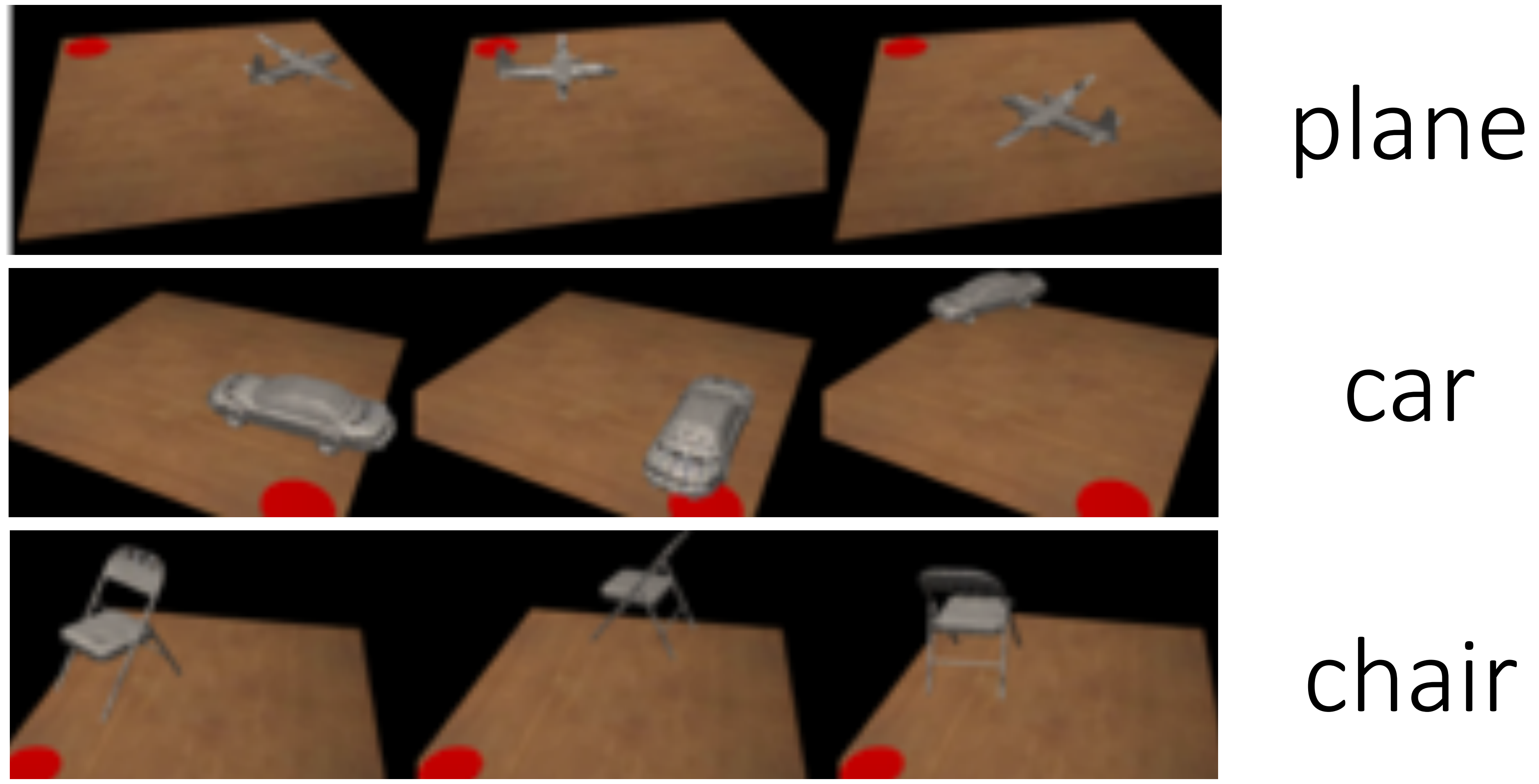
[1] Finn et al. ICML '17

FTML: practical instantiation of our approach, extending MAML[1]
meta-train on all data so far, fine-tune on current task

Experiment with **sequences of tasks**:

- Colored, rotated, scaled **MNIST**
- **3D object pose prediction**
- **CIFAR-100** classification

Example pose prediction tasks



[1] Finn et al. ICML '17

Experiments

Experiments

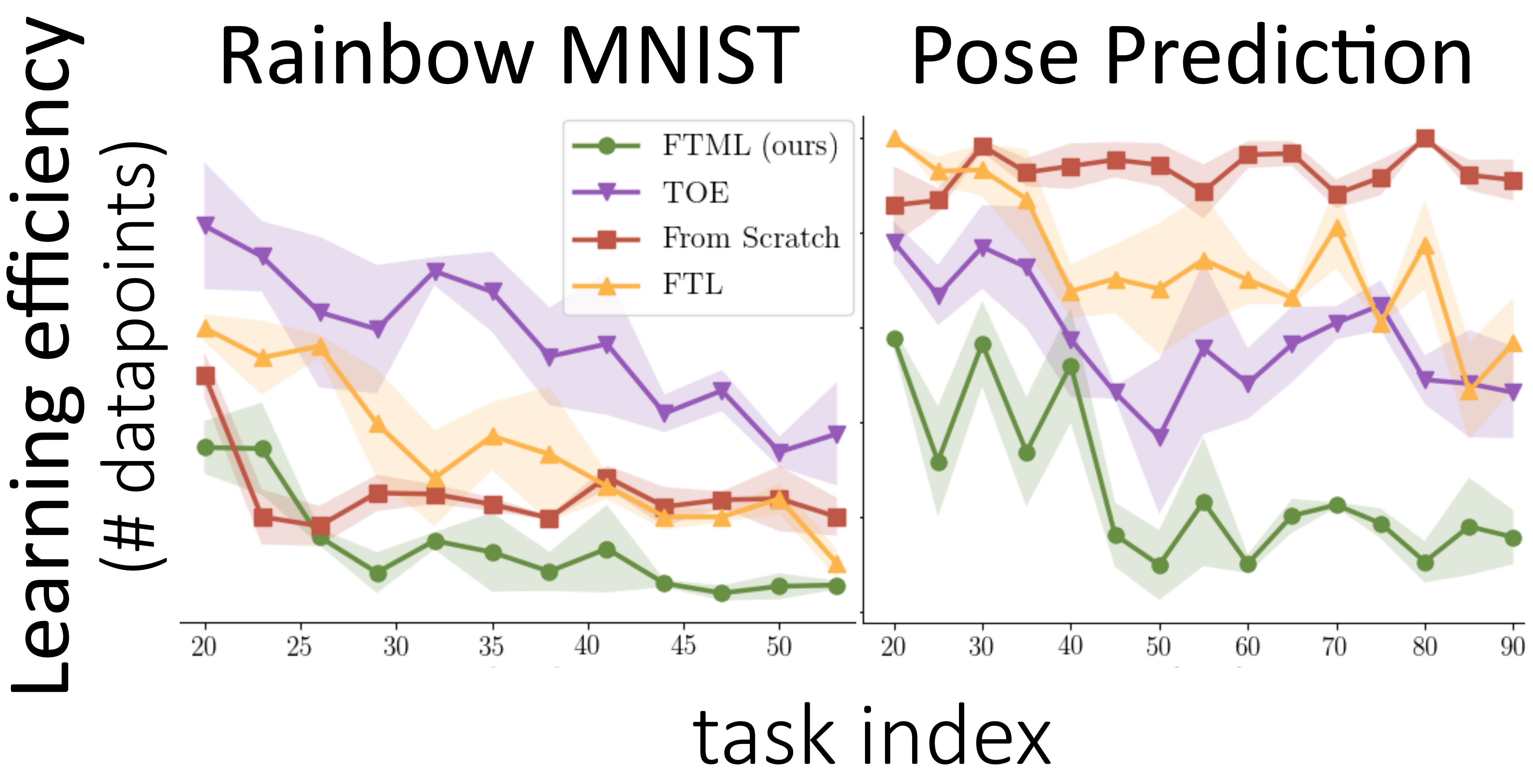
Learning efficiency
(# datapoints)

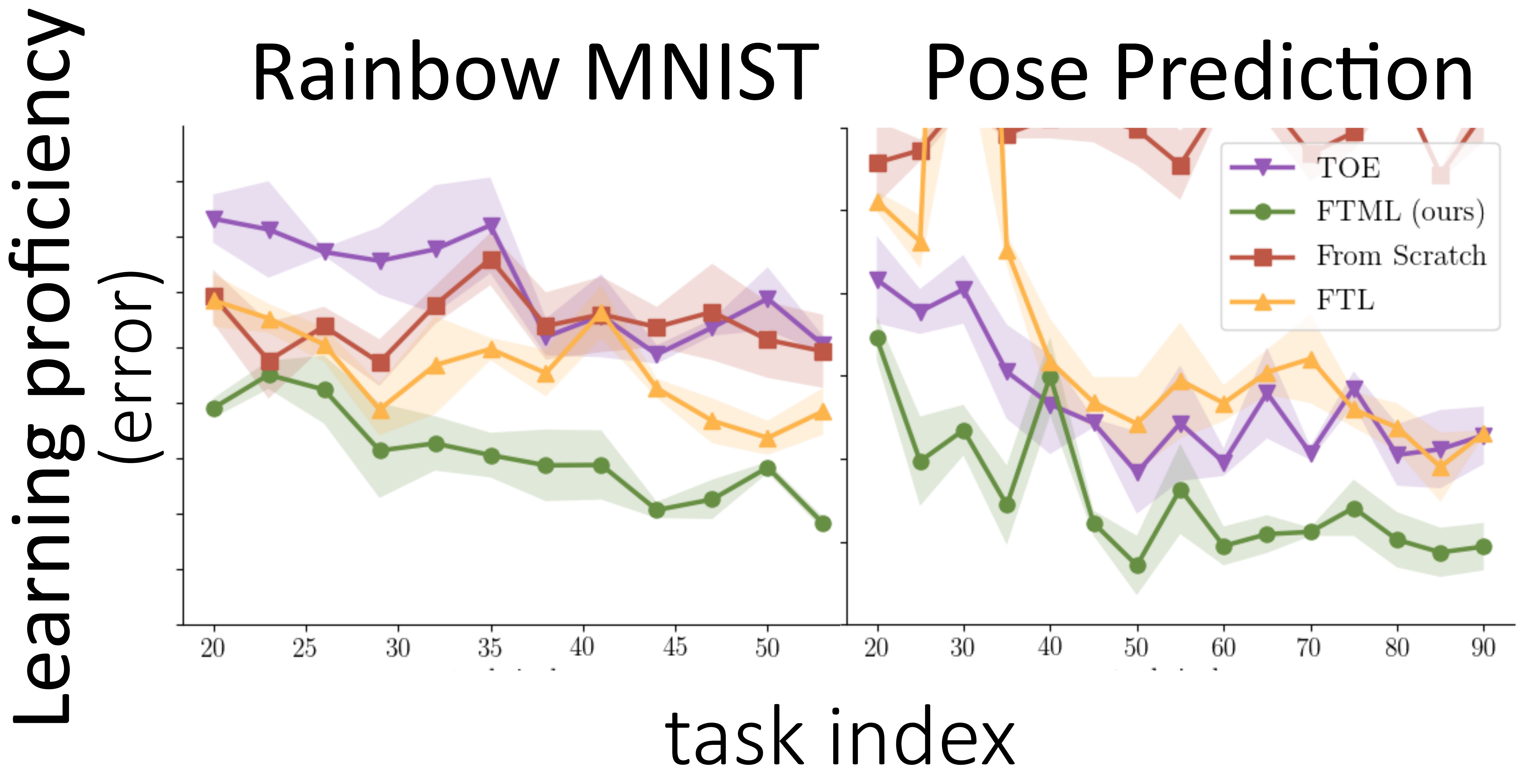
task index

Learning proficiency
(error)

task index

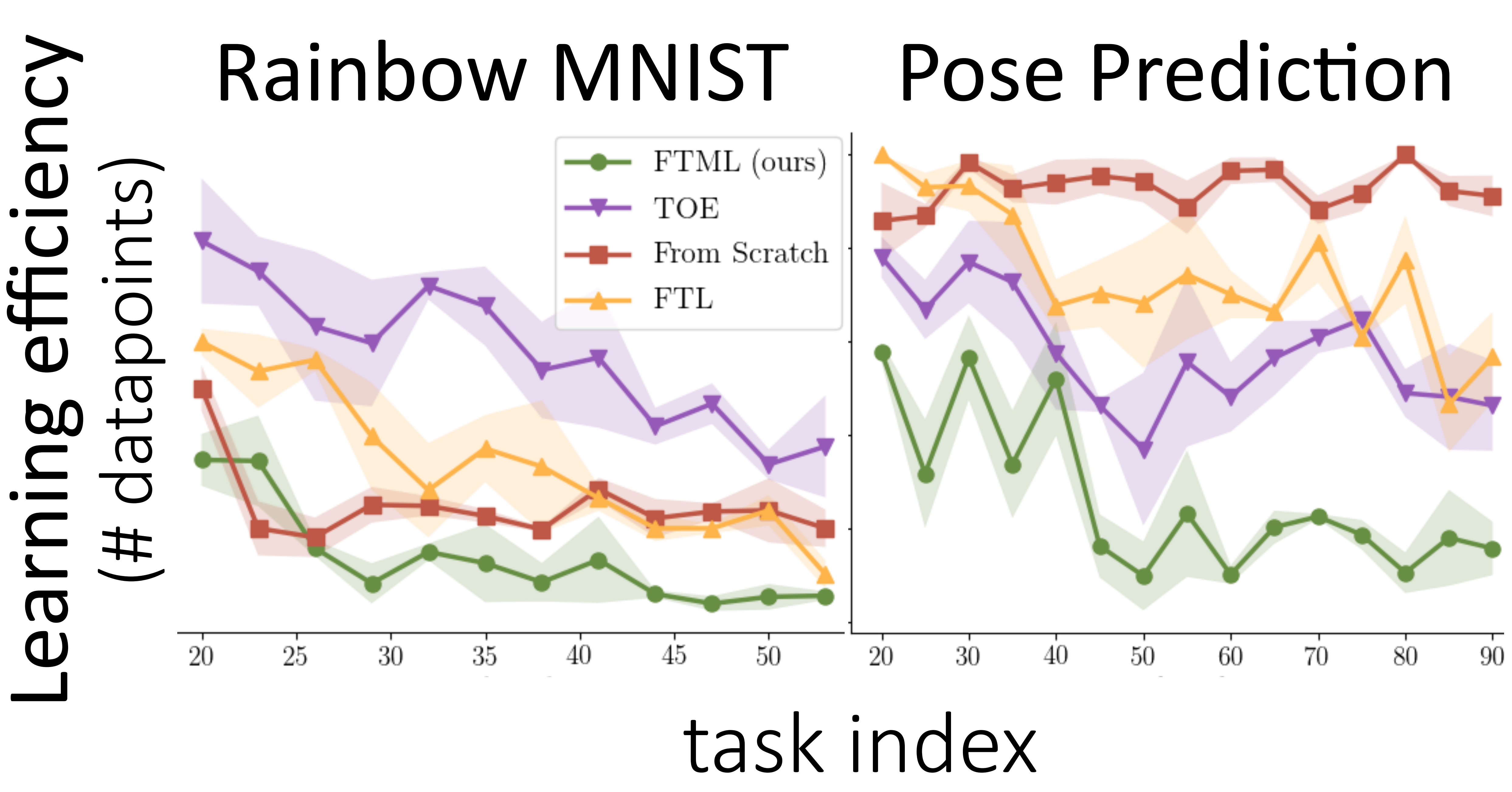
Experiments

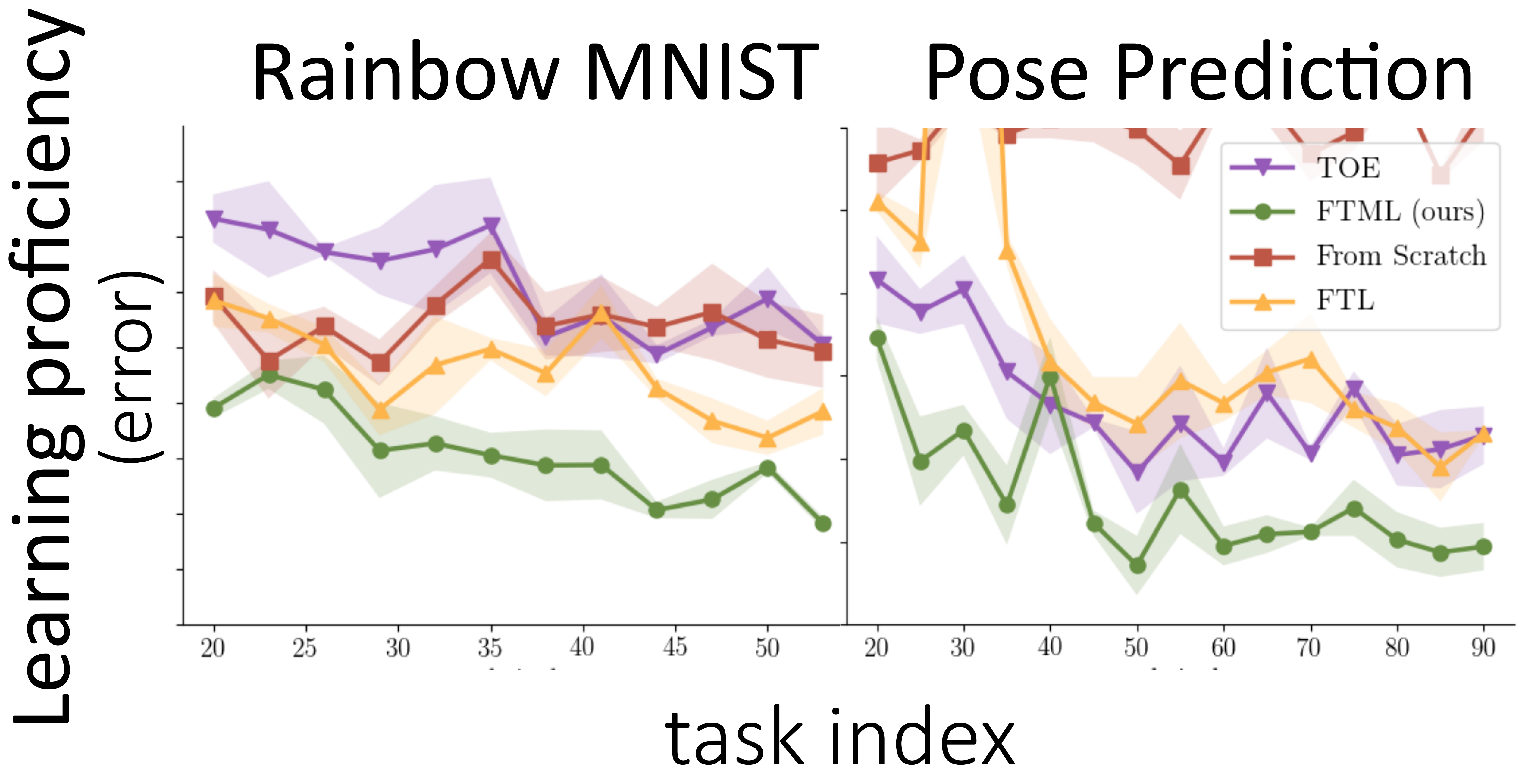




FTML (ours)

Experiments





FTML (ours)

learns each new task faster & **with greater proficiency**,

Experiments

FTML (ours)

learns each new task faster & **with greater proficiency**,

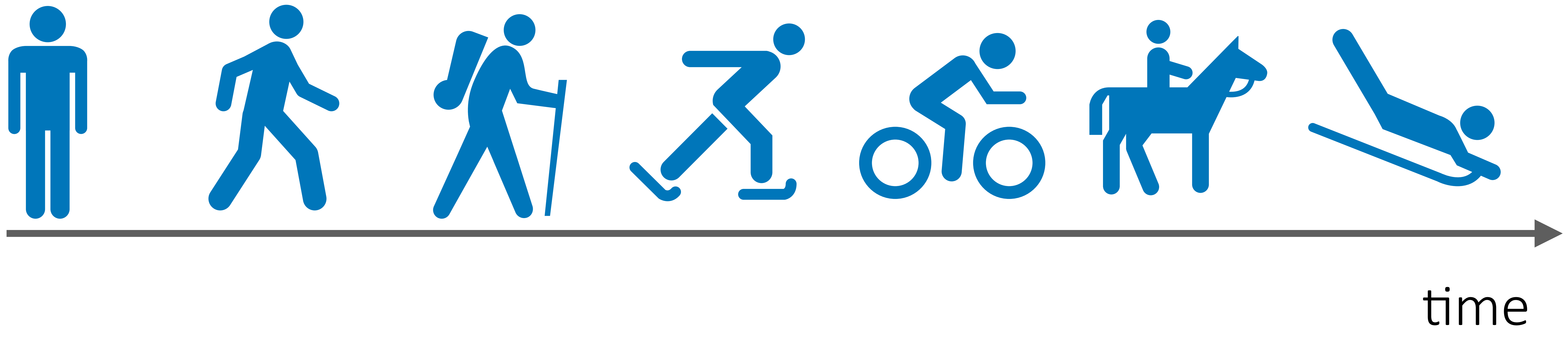
approaches **few-shot learning** regime

Takeaways

Introduced **online meta-learning** problem formulation

Meta-learning is effective in **non-stationary settings**

Similar guarantees to online learning, but **better empirical performance**



For more, come see us at **poster #5**!