Learning to Learn to Communicate

Osvaldo Simeone
Joint work with Sangwoo Park, Sharu Theresa Jose,
Hyeryung Jang, and Joonhyuk Kang

MLCOM 2020, 7/9/2020

Meta-Learning to Communicate

Osvaldo Simeone
Joint work with Sangwoo Park, Sharu Theresa Jose,
Hyeryung Jang, and Joonhyuk Kang

MLCOM 2020, 7/9/2020

Overview

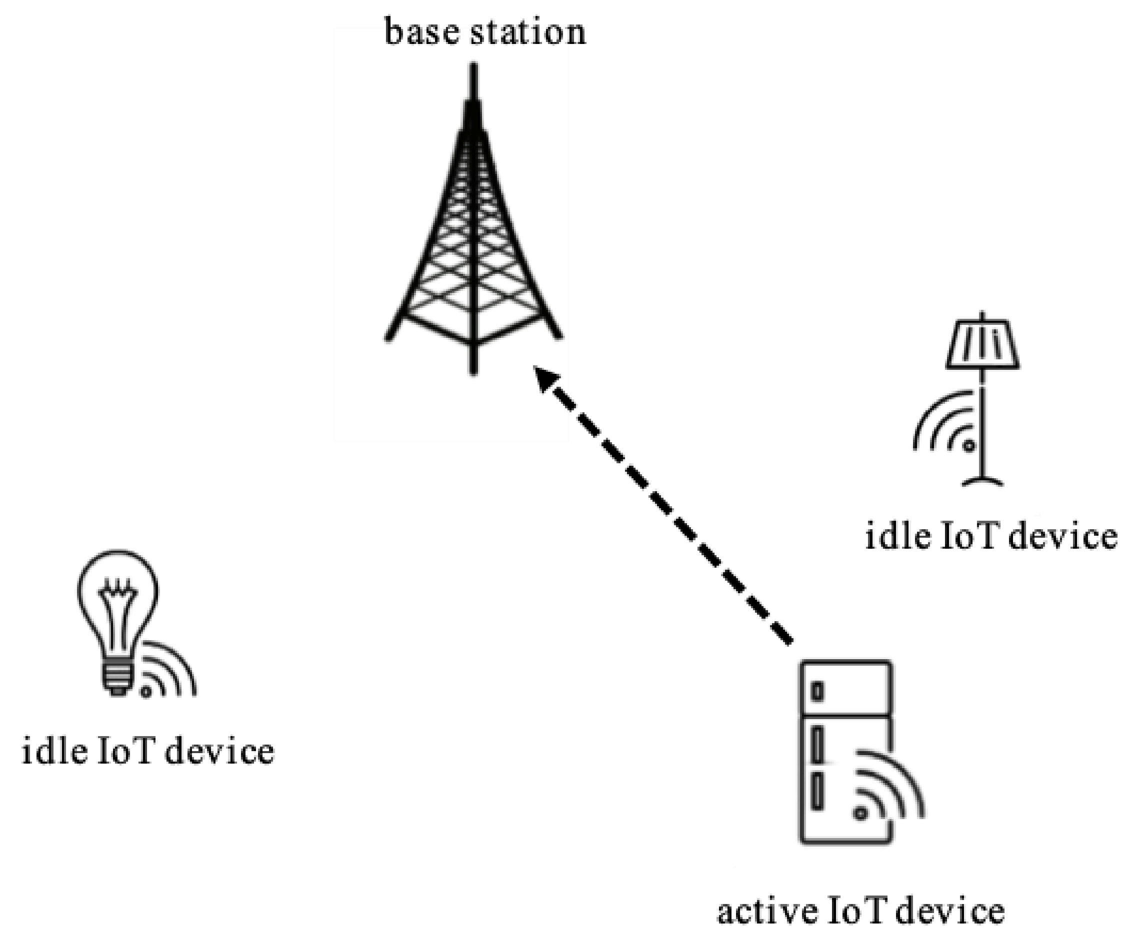
- Motivation
- Meta-learning in a nutshell
- Algorithms and applications

Overview

- Motivation
- Meta-learning in a nutshell
- Algorithms and applications

Motivation

- In the Internet of Things (IoT), devices transmit sporadically using short packets with few pilot symbols.



Motivation

- Conventional model-based approach: estimate the (linear) fading channel and then use it in an optimal coherent demodulator.
- Model deficit (e.g., transmitter's hardware imperfections) → machine learning (ML)

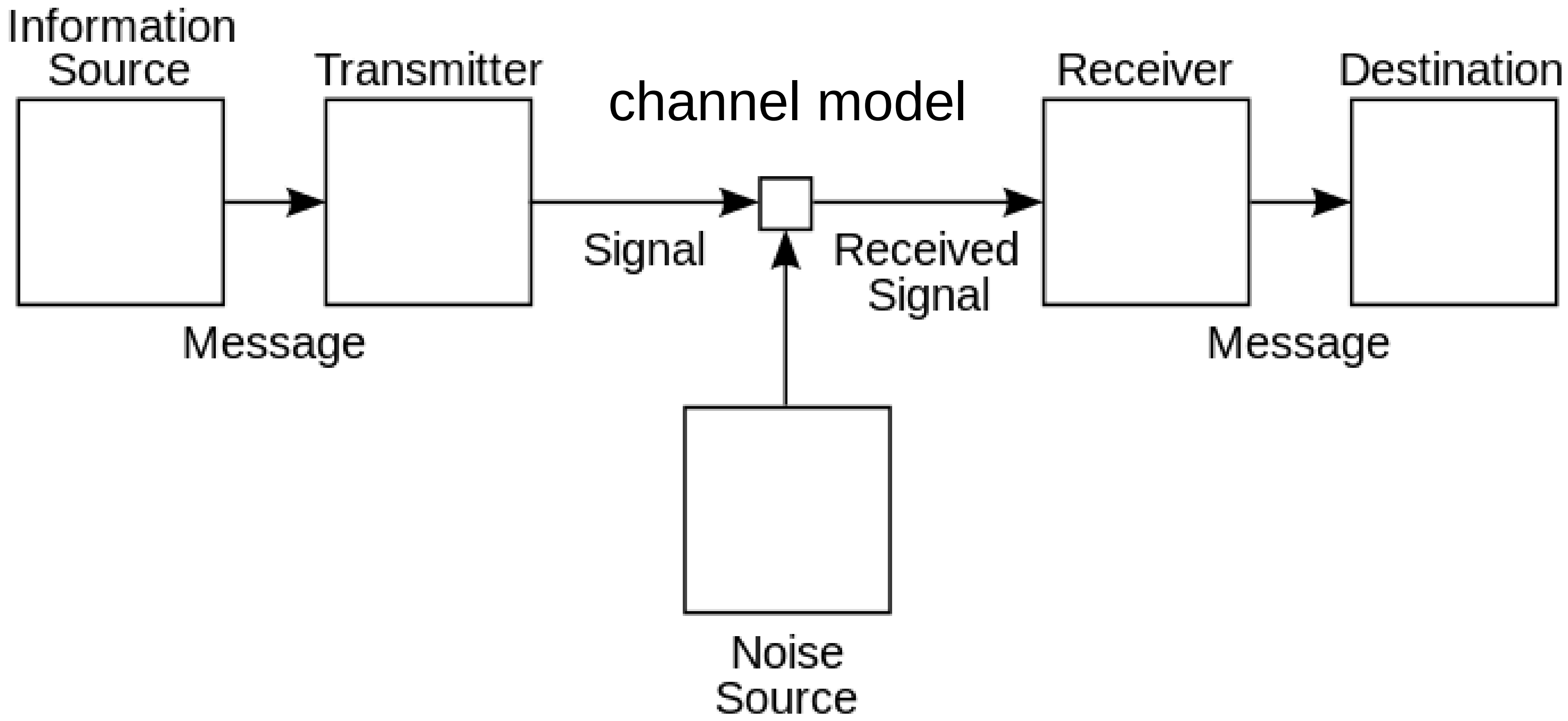
Motivation

- Conventional model-based approach: estimate the (linear) fading channel and then use it in an optimal coherent demodulator.
- Model deficit (e.g., transmitter's hardware imperfections) → machine learning (ML)
- ML requires enough pilot data from each device:

Can ML tools be useful with few pilots per device?
(sample complexity)

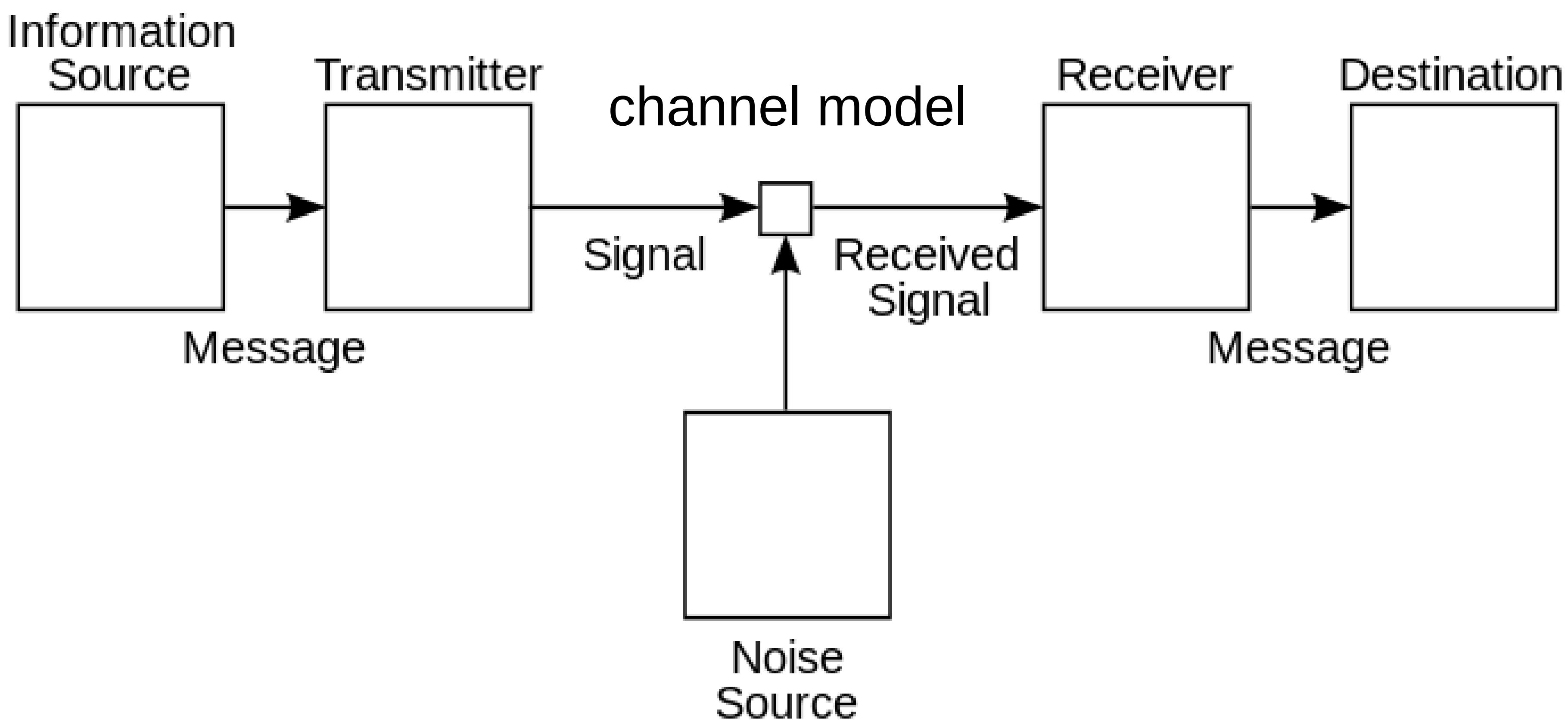
Motivation

- End-to-end training for communication on known channel model to tackle an algorithm deficit



Motivation

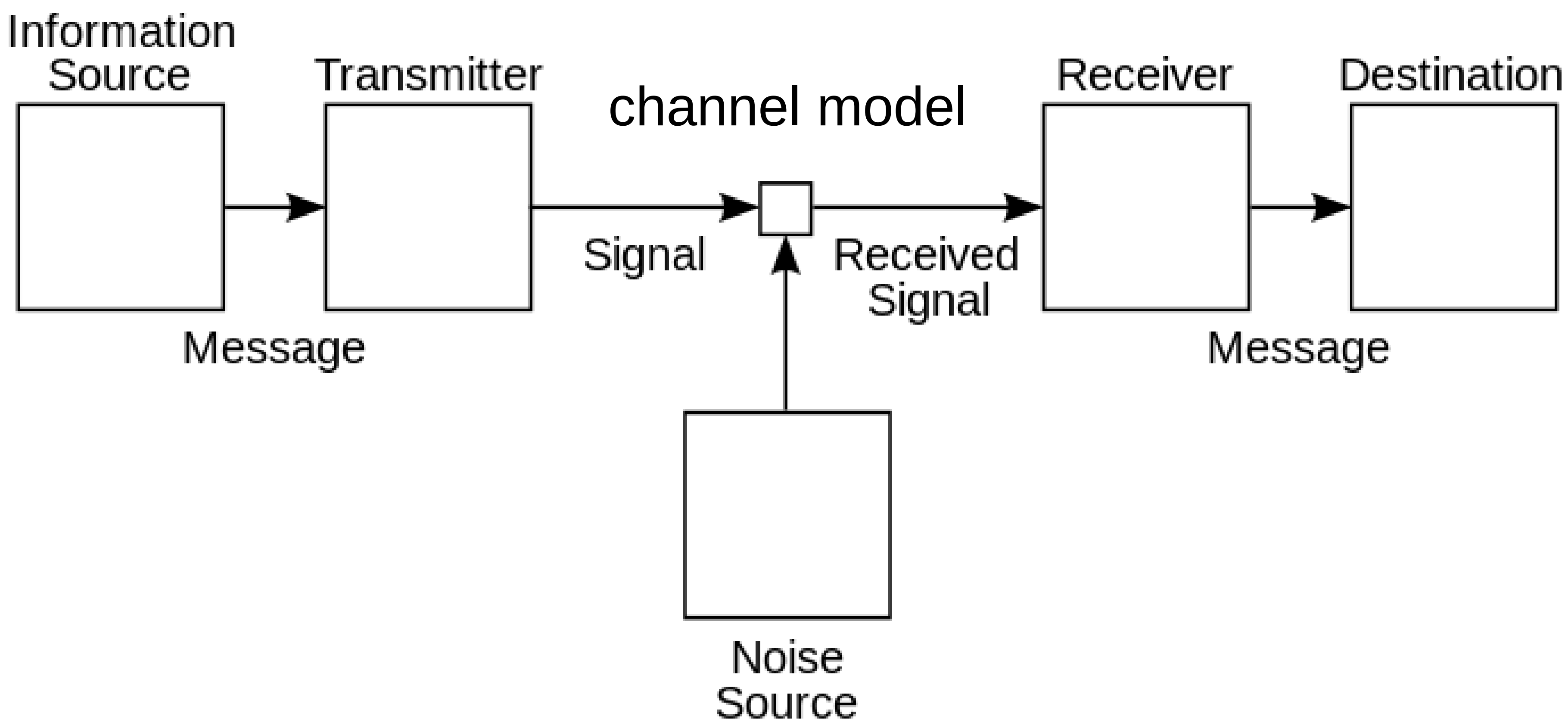
- End-to-end training for communication on known channel model to tackle an algorithm deficit



- Since the channel model is known, data can be generated at will, but training must be redone for each channel realization…

Motivation

- End-to-end training for communication on known channel model to tackle an algorithm deficit



- Since the channel model is known, data can be generated at will, but training must be redone for each channel realization…

Can ML tools be useful with few training iterations per channel realization?
(iteration complexity)

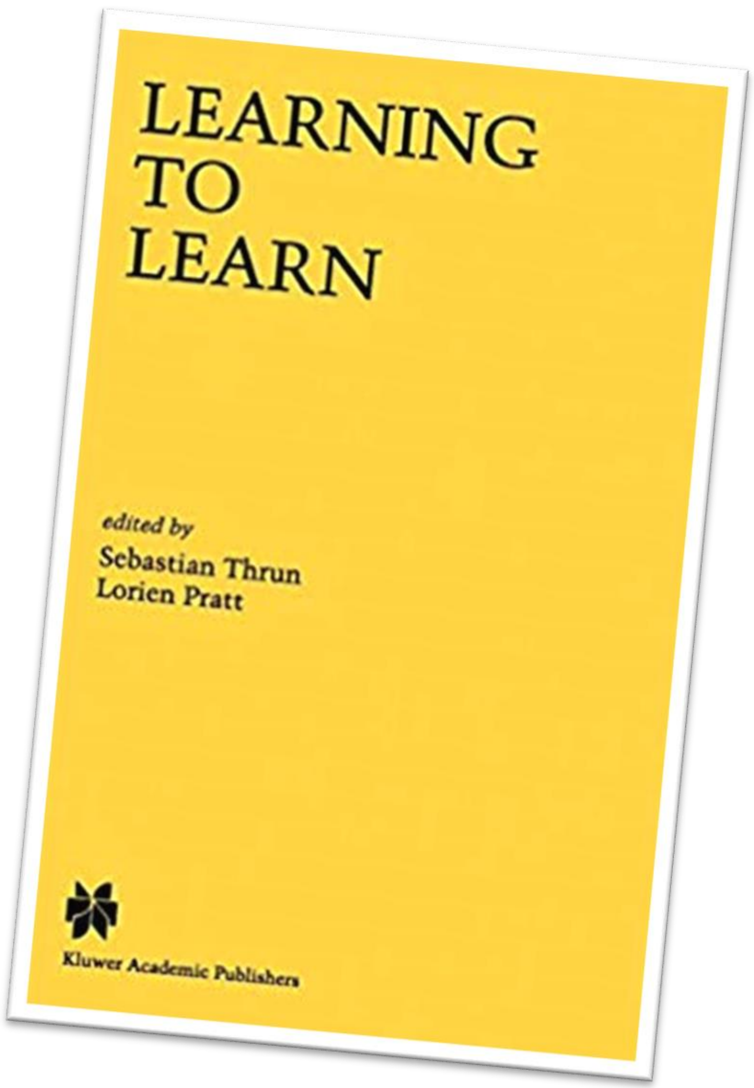
Overview

- Motivation
- Meta-learning in a nutshell
- Algorithms and applications

Google Scholar

meta-learning

Articles Case law



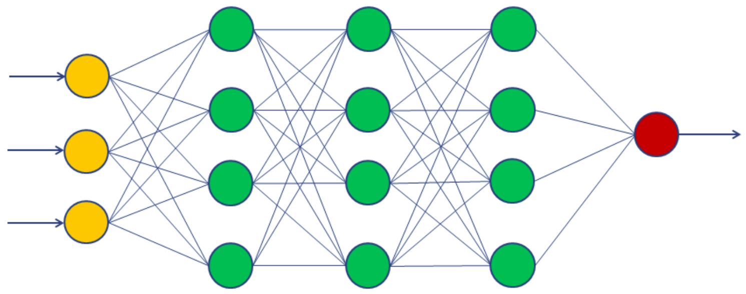
1997

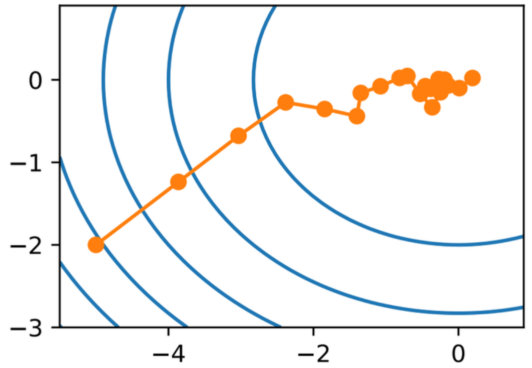
Model-agnostic meta-learning for fast adaptation of deep networks
C Finn, P Abbeel, S Levine - arXiv preprint arXiv:1703.03400, 2017 - arxiv.org
We propose an algorithm for **meta-learning** that is **model-agnostic**, in the sense that it is compatible with any **model** trained with gradient descent and applicable to a variety of different learning problems, including classification, regression, and reinforcement learning …
Cited by 2124 Related articles All 13 versions

Conventional Machine Learning

model class
+ training procedure

inductive
bias

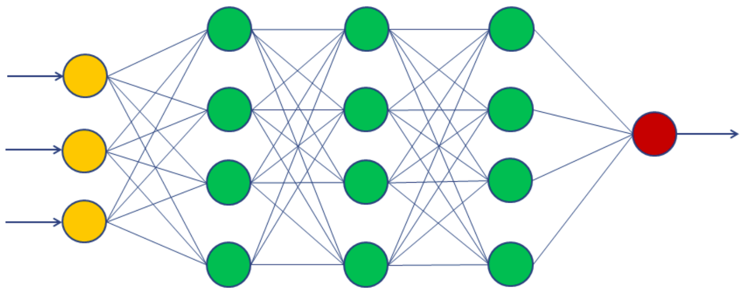


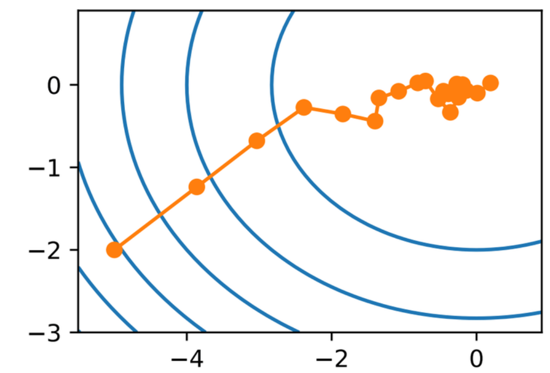


Conventional Machine Learning

model class
+ training procedure

inductive bias



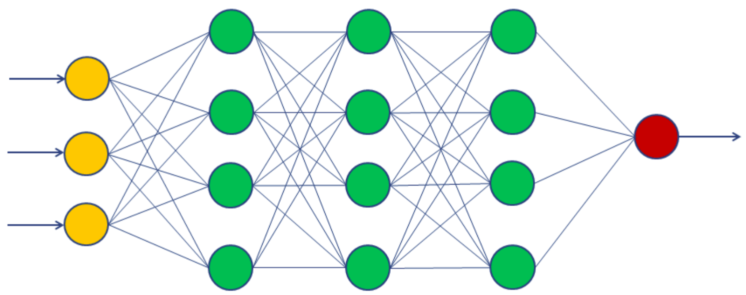


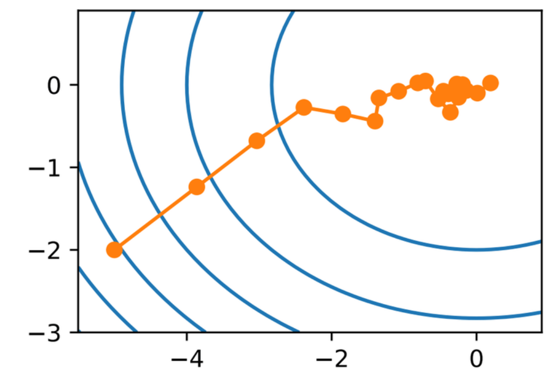
$\mathcal{D}_k^{\mathrm{tr}}$

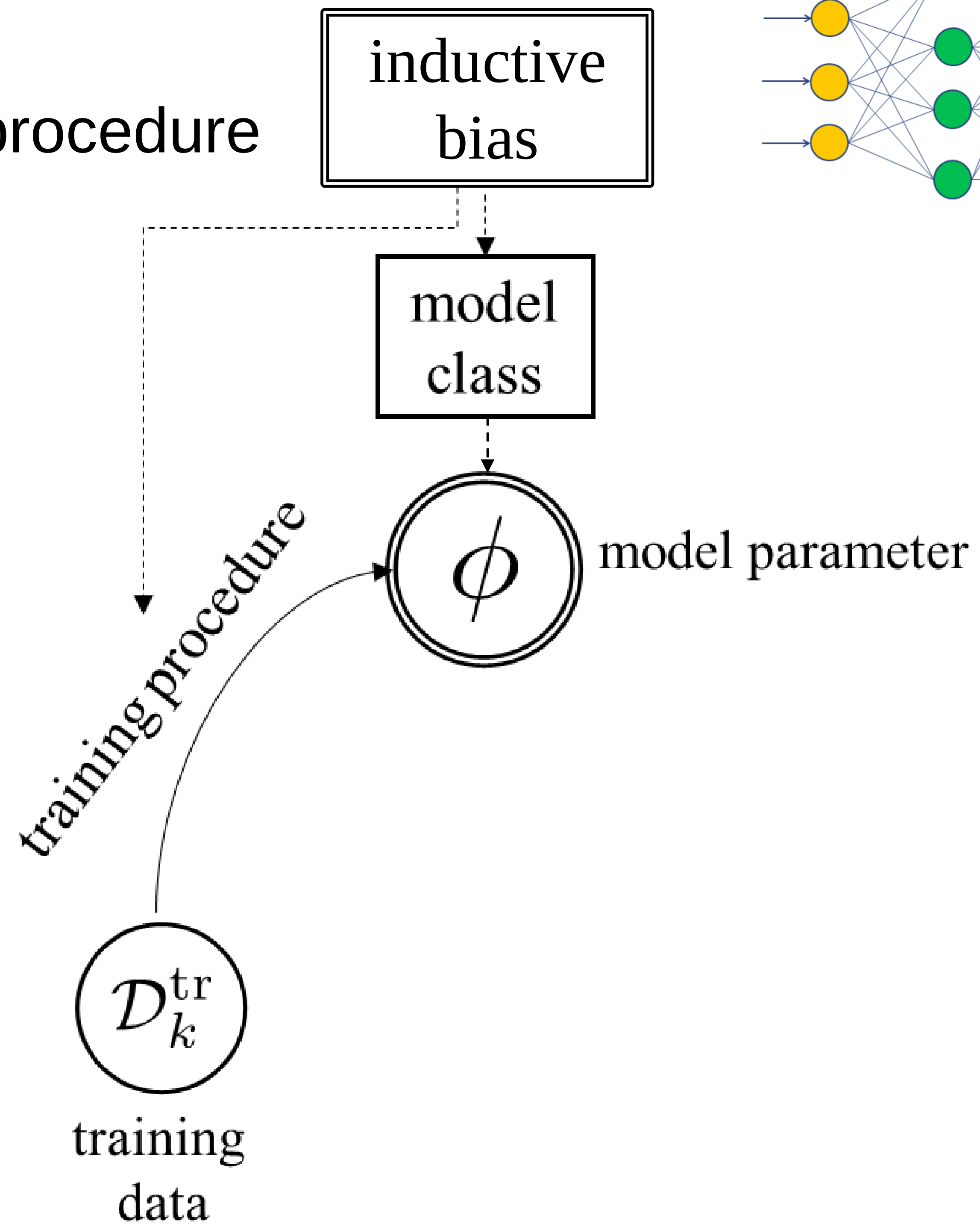
training data

Conventional Machine Learning

model class
+ training procedure

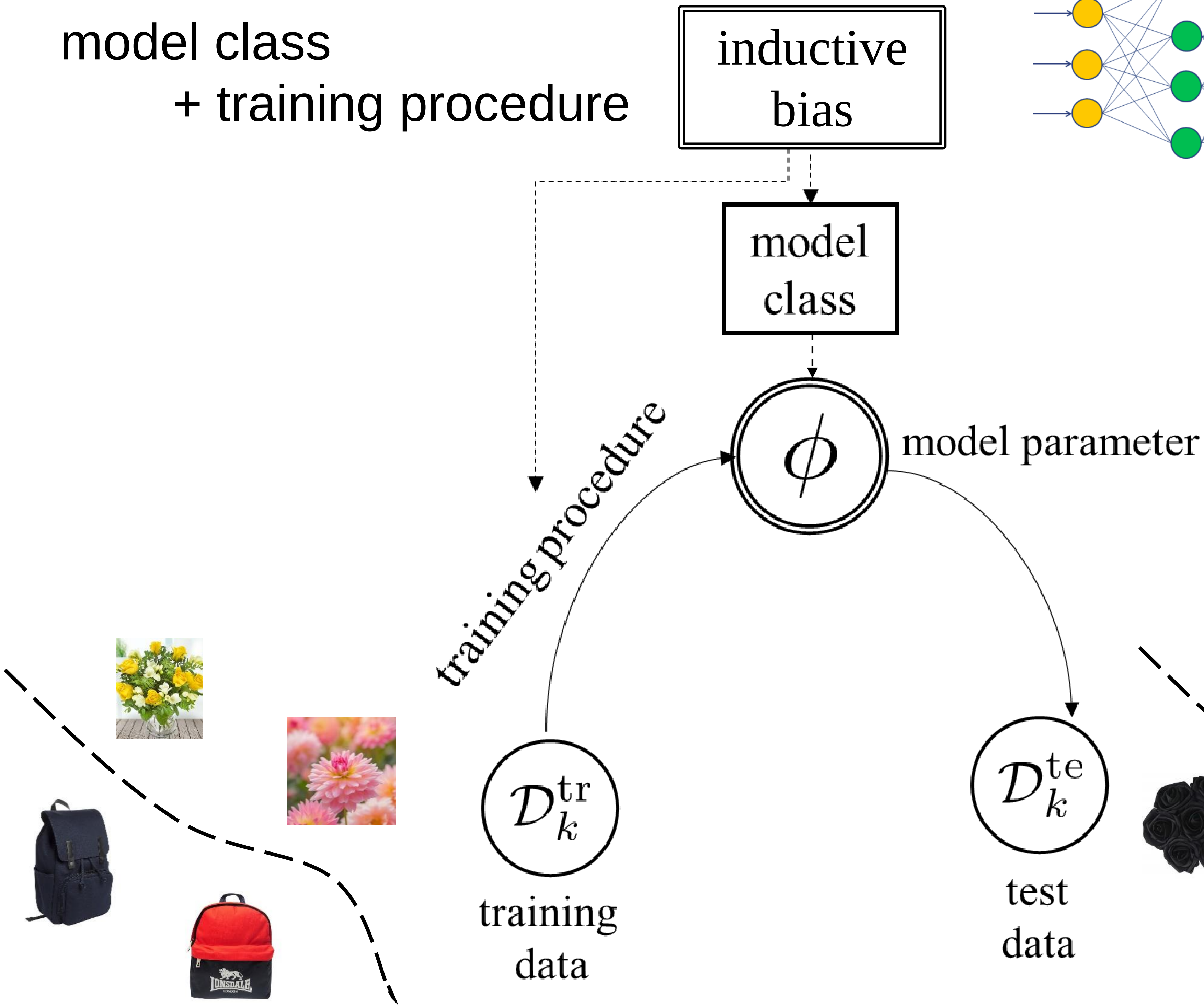


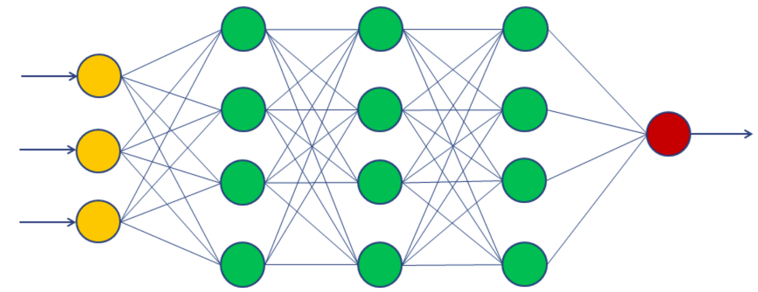


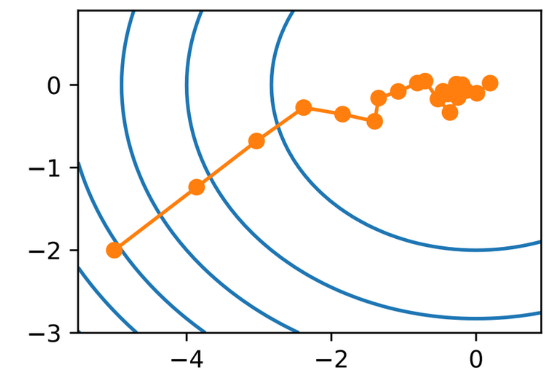


Conventional Machine Learning

model class
+ training procedure

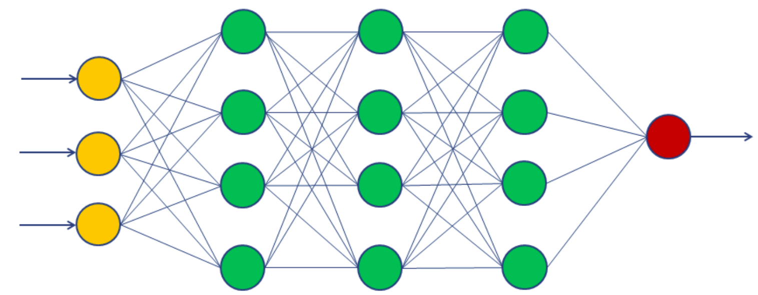


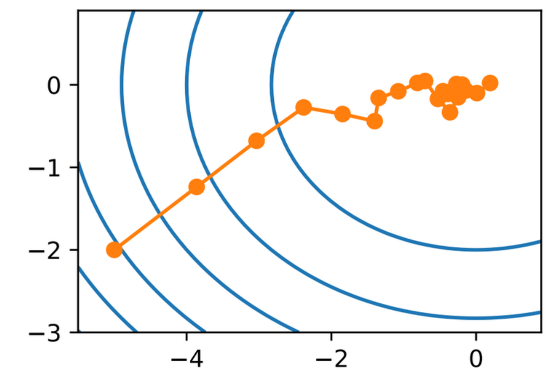


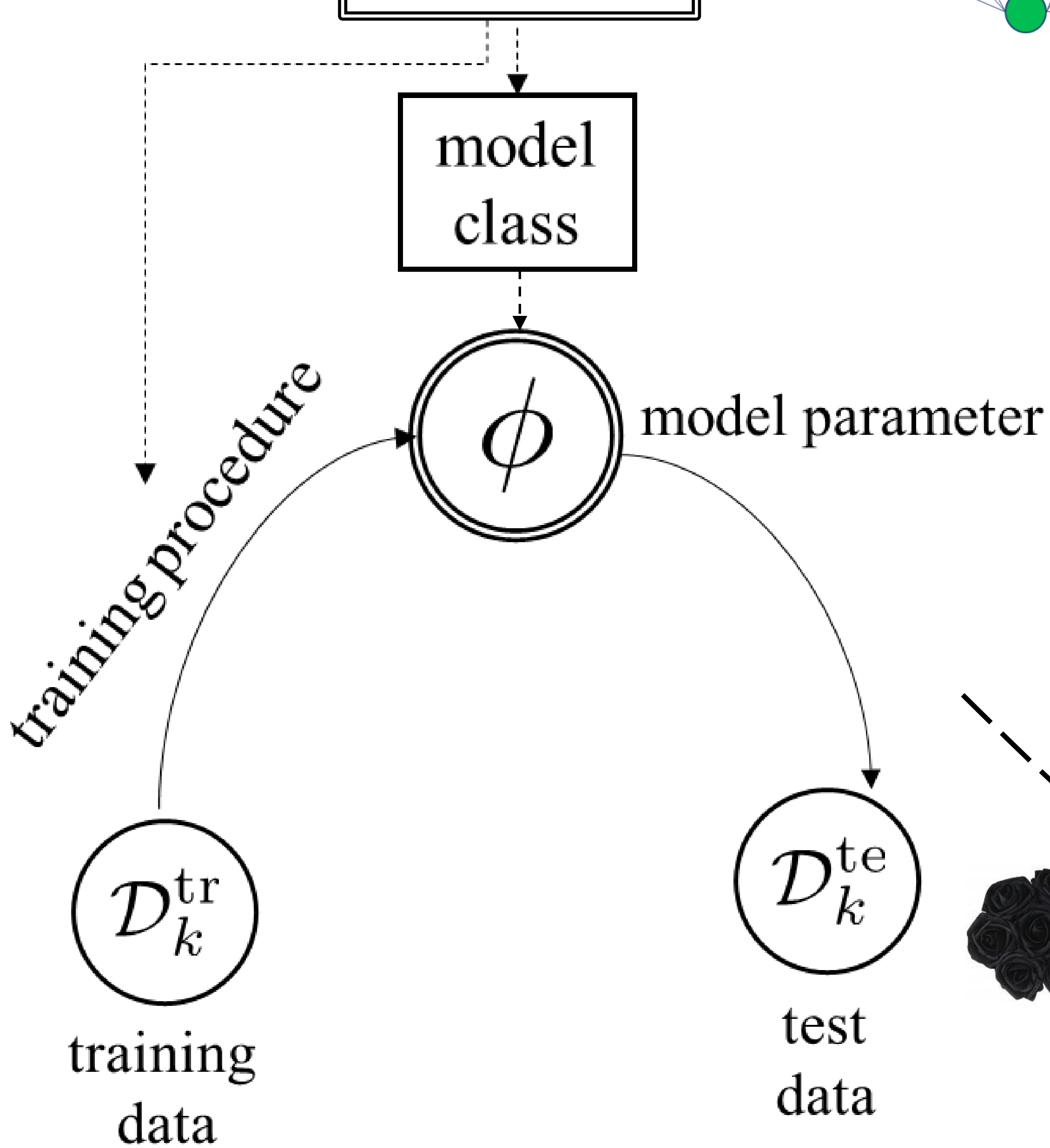


Conventional Machine Learning

model class
+ training procedure





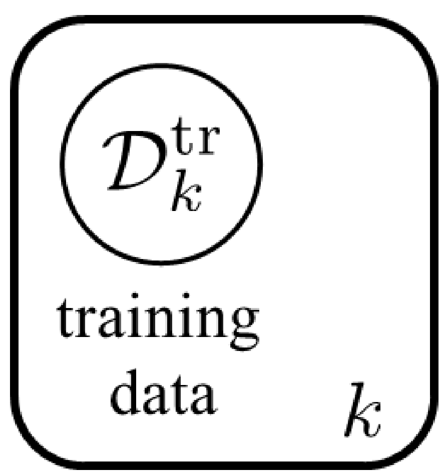


Requires enough data and training time for each new task…

Joint Learning

- Train a single model
  for a class of tasks

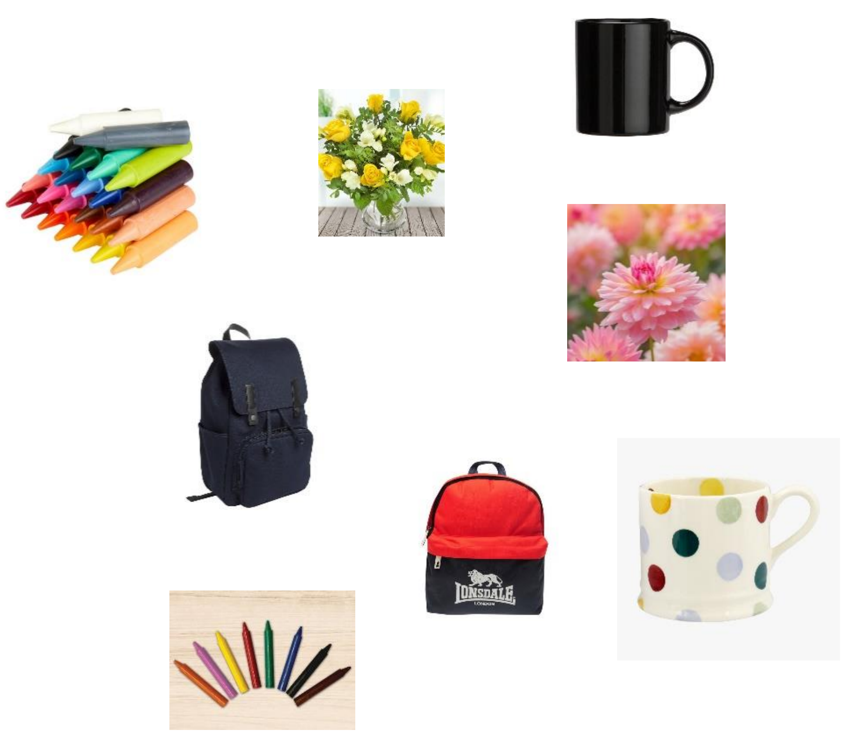
inductive
bias



Joint Learning

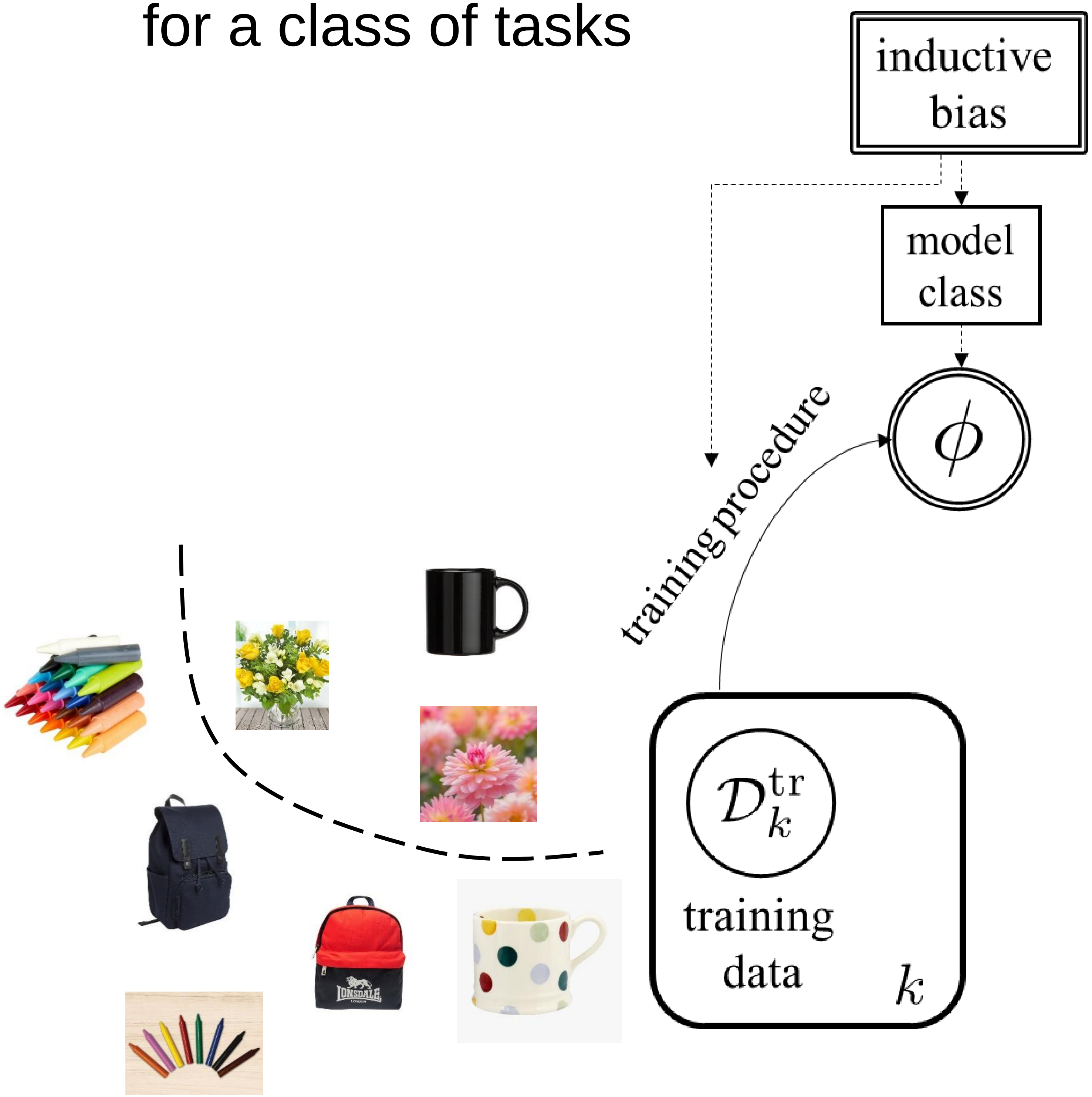
- Train a single model for a class of tasks

inductive bias



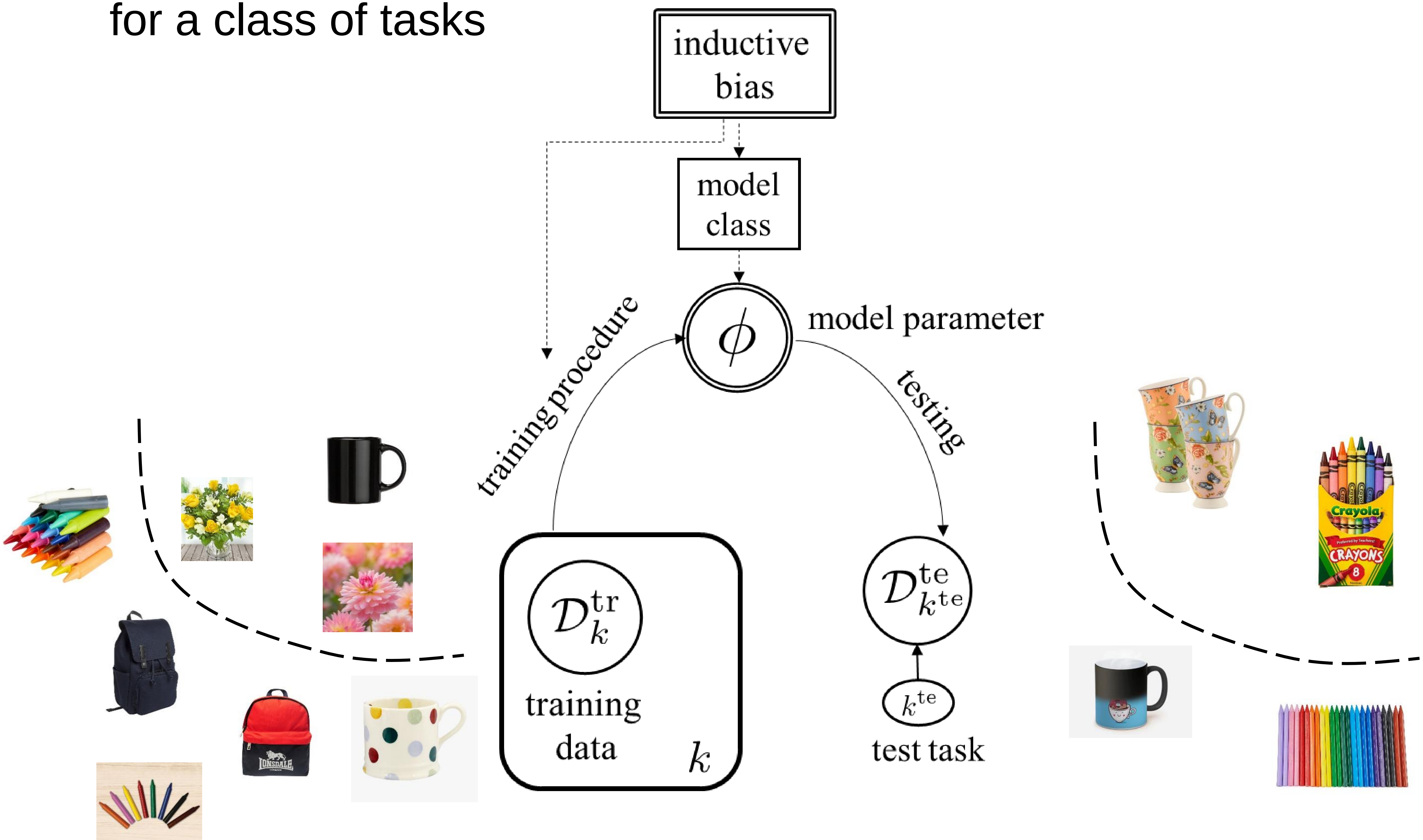
Joint Learning

- Train a single model for a class of tasks



Joint Learning

- Train a single model for a class of tasks

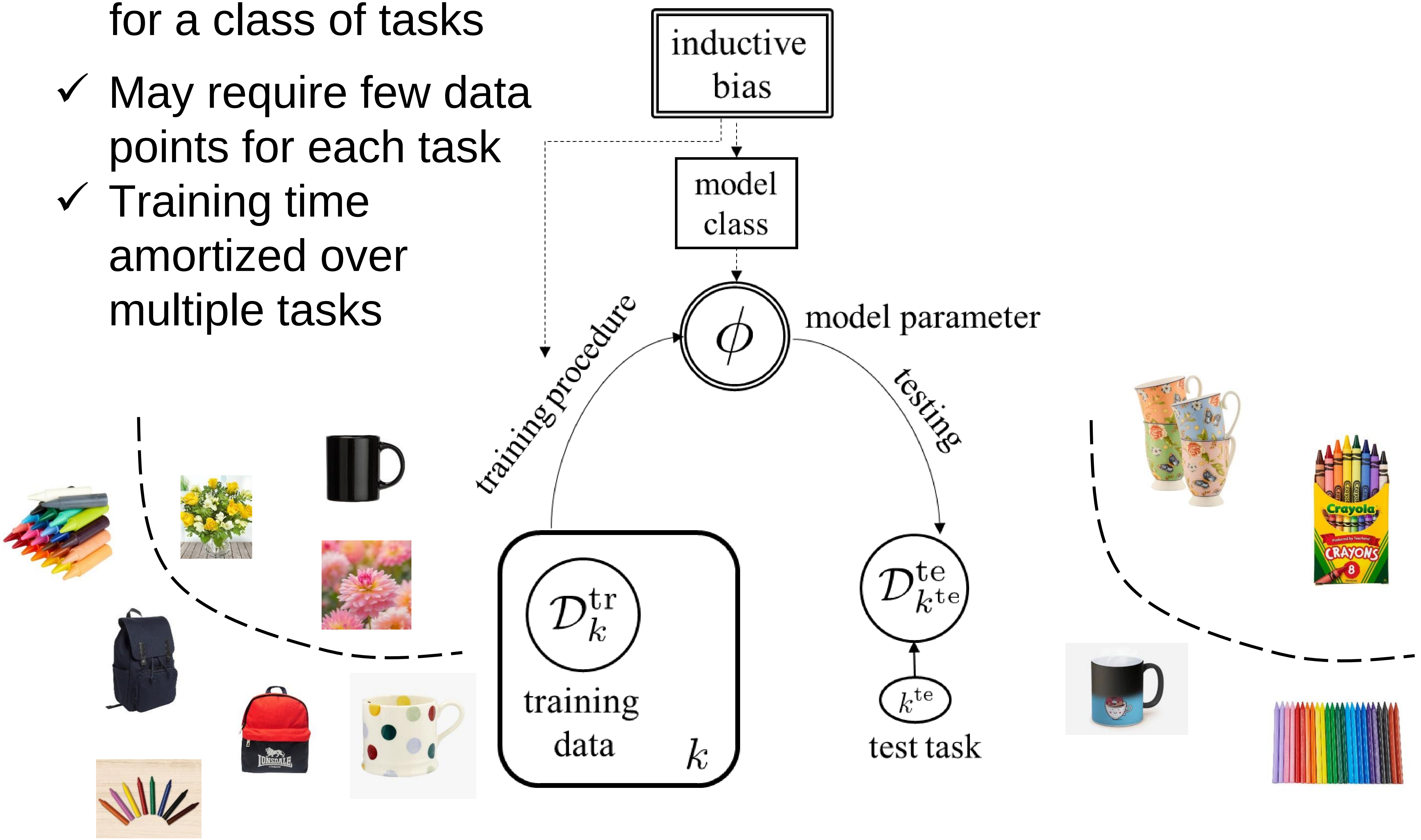


Joint Learning

- Train a single model for a class of tasks

✓ May require few data points for each task

✓ Training time amortized over multiple tasks

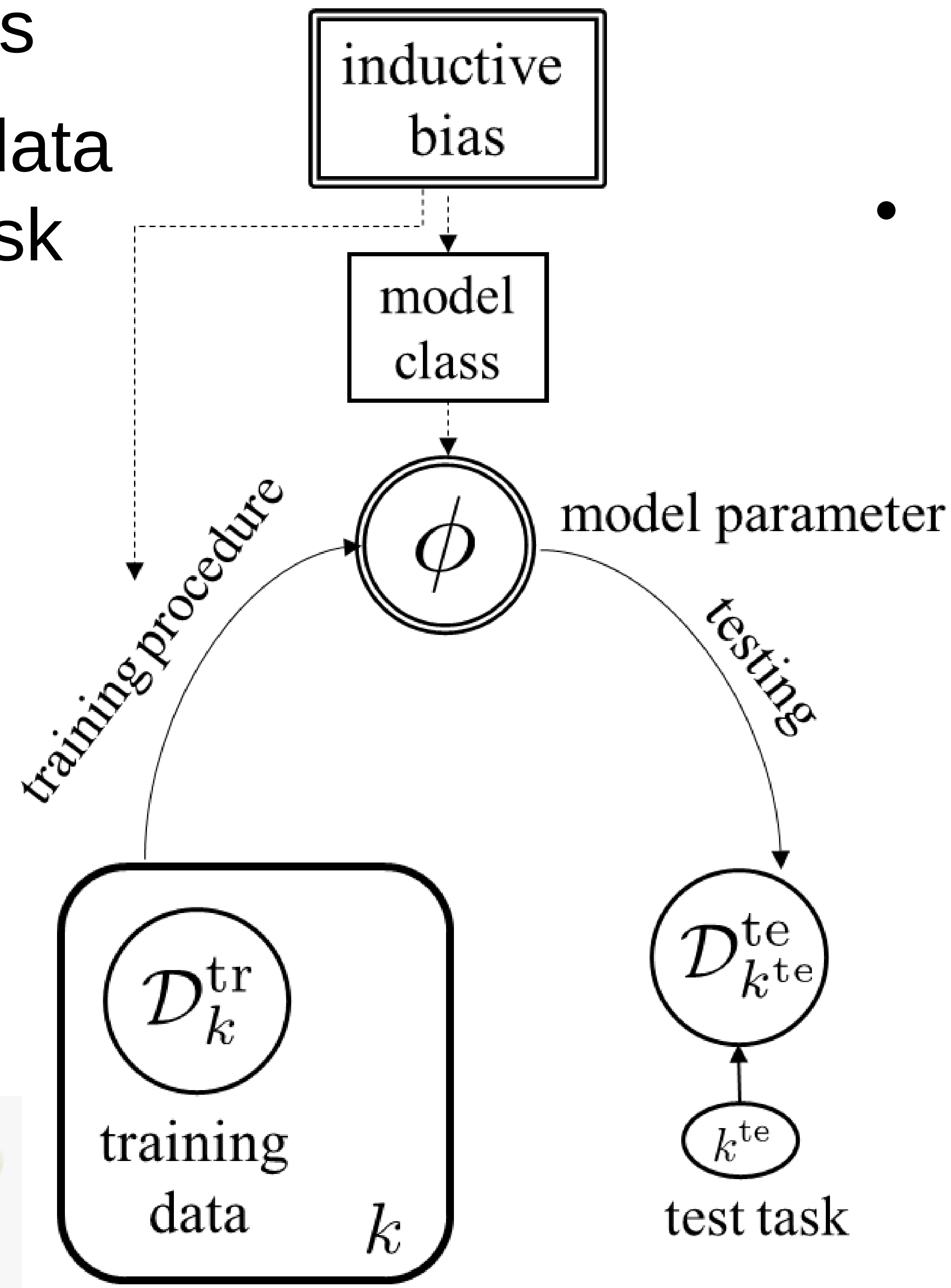


Joint Learning

- Train a single model for a class of tasks

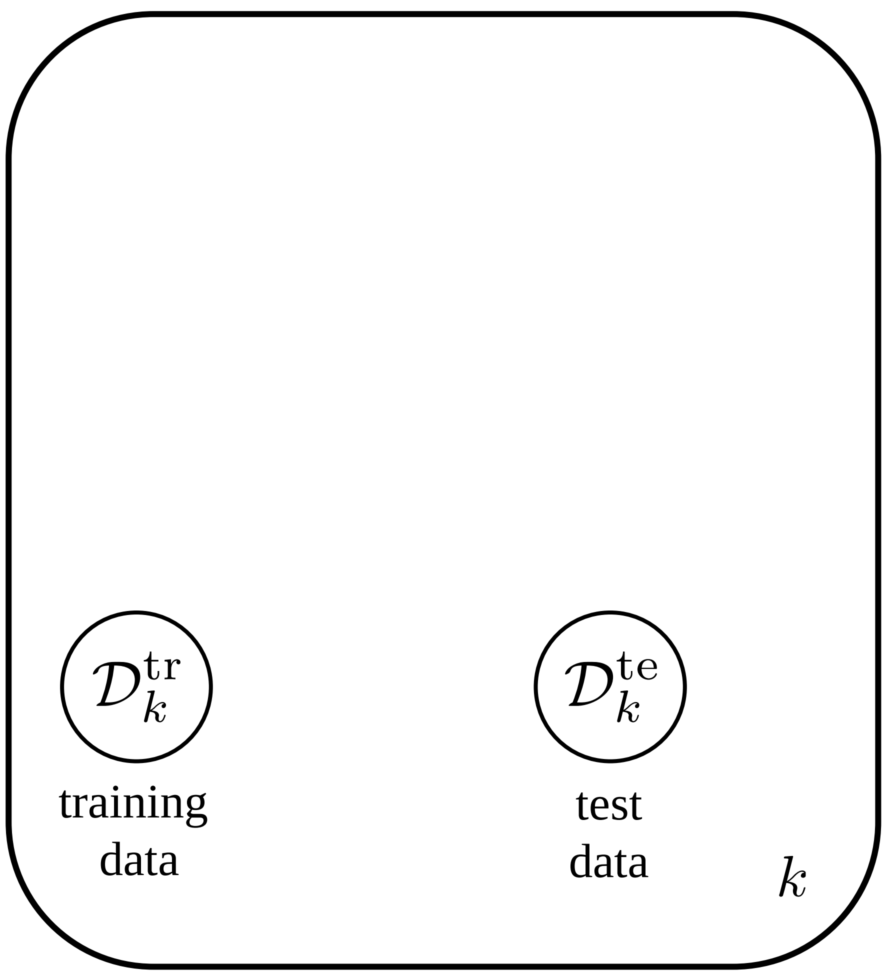
✓ May require few data points for each task

✓ Training time amortized over multiple tasks

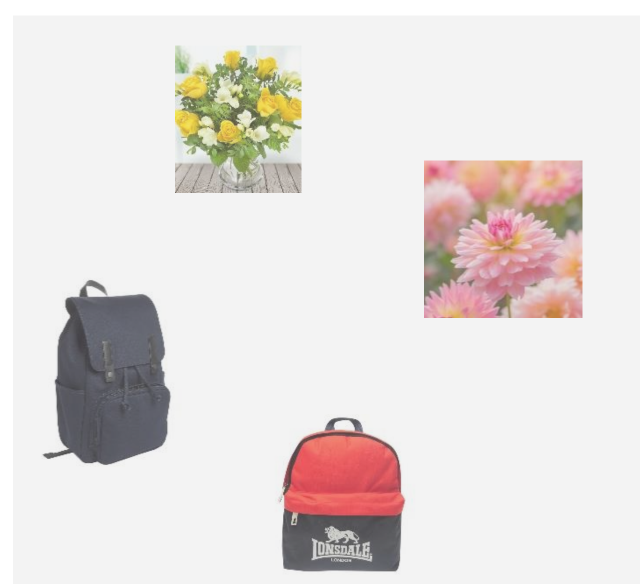


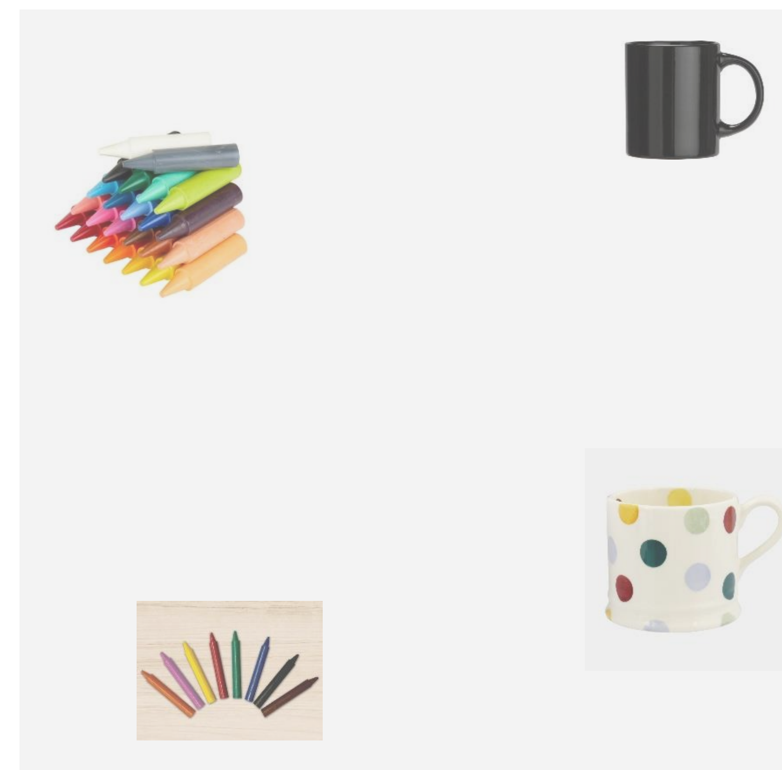
- There may not be a single model that works well on all task
- Unable to adapt to new tasks

Meta-Learning

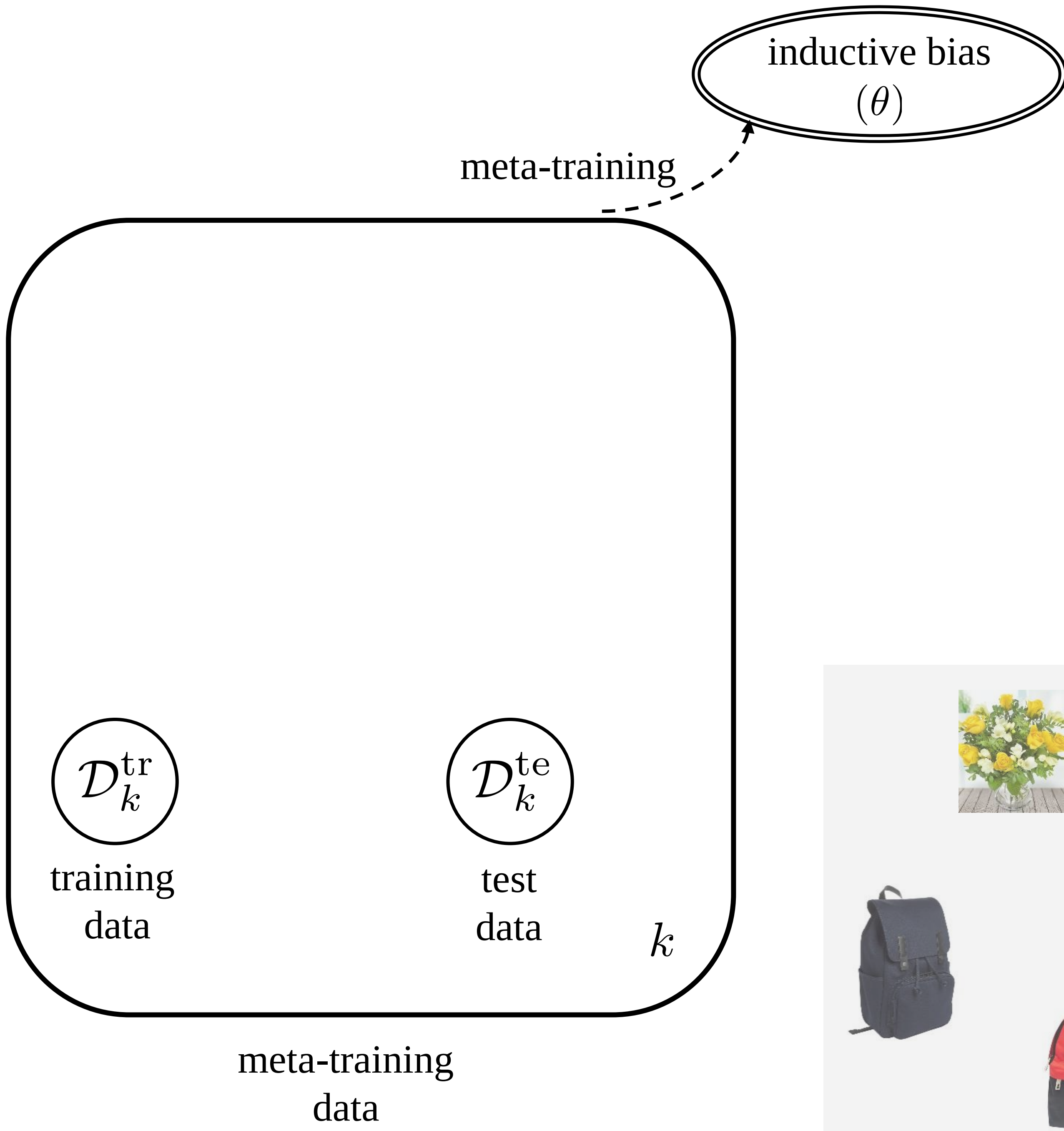


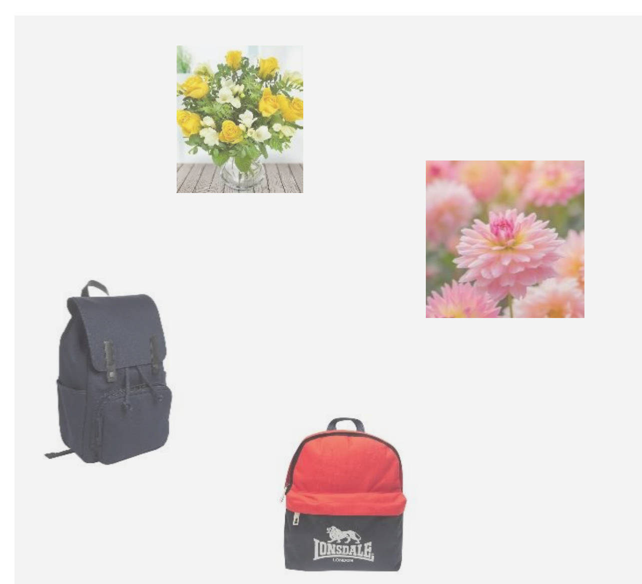
meta-training data

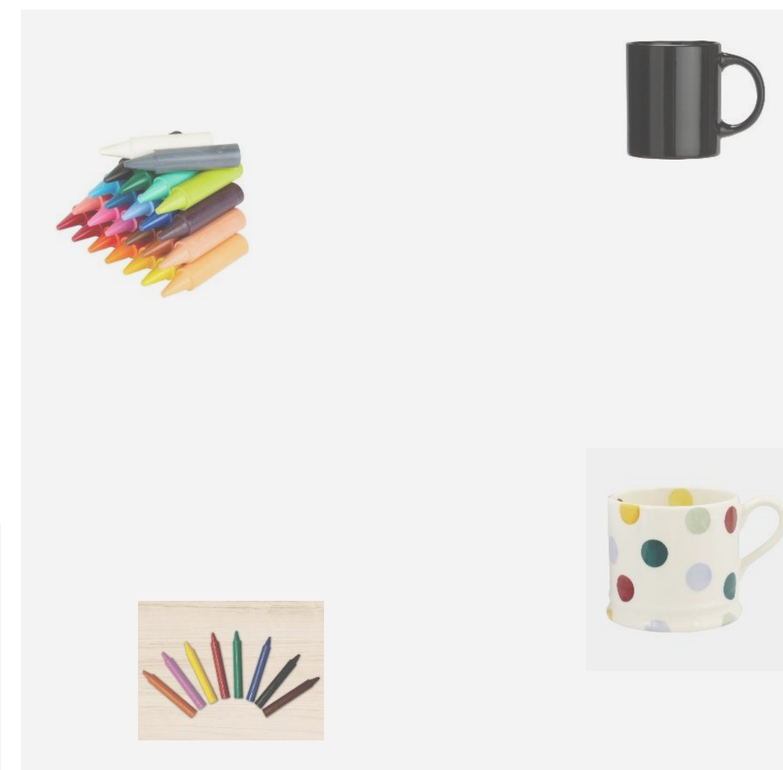




Meta-Learning

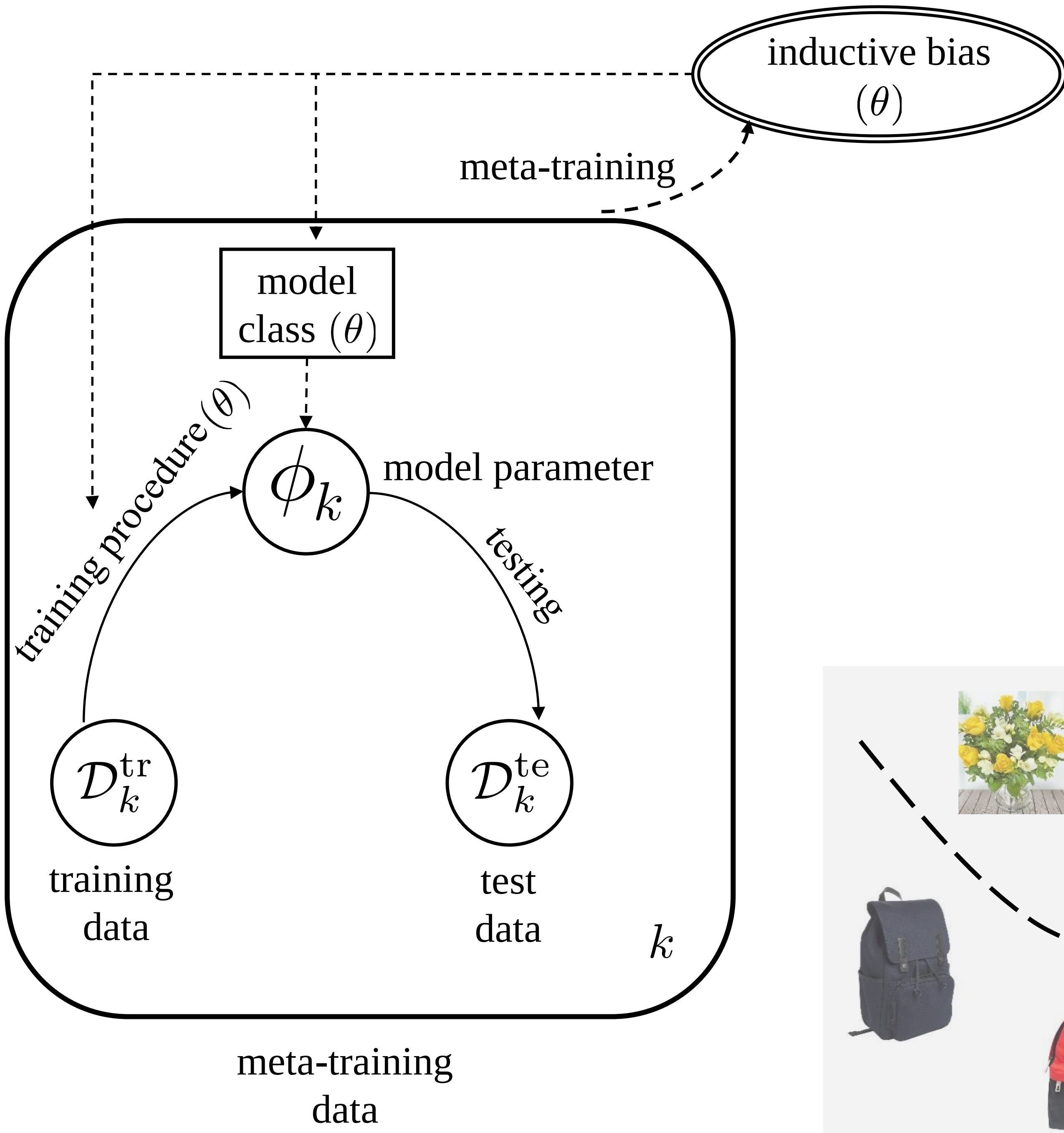


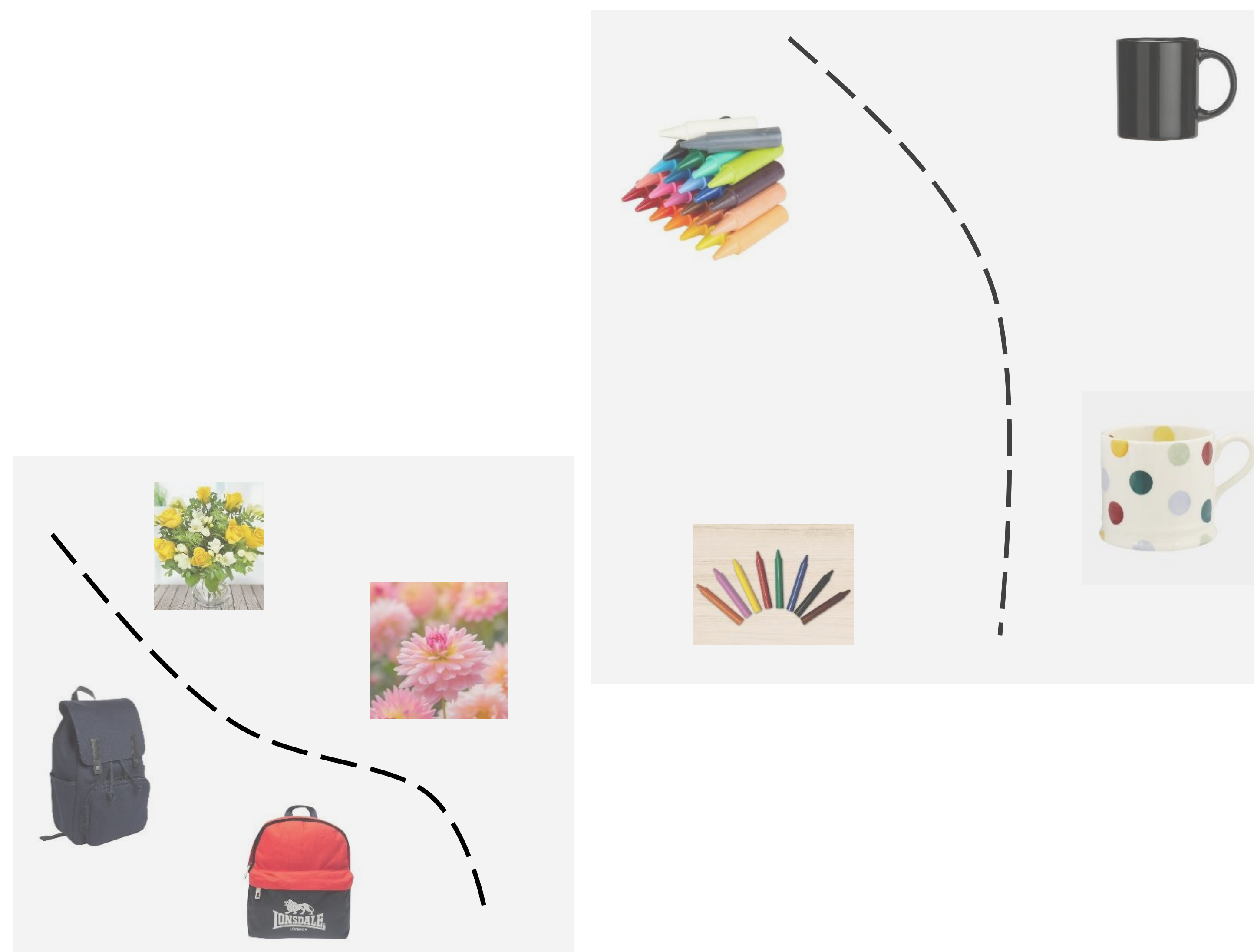




Meta-Learning

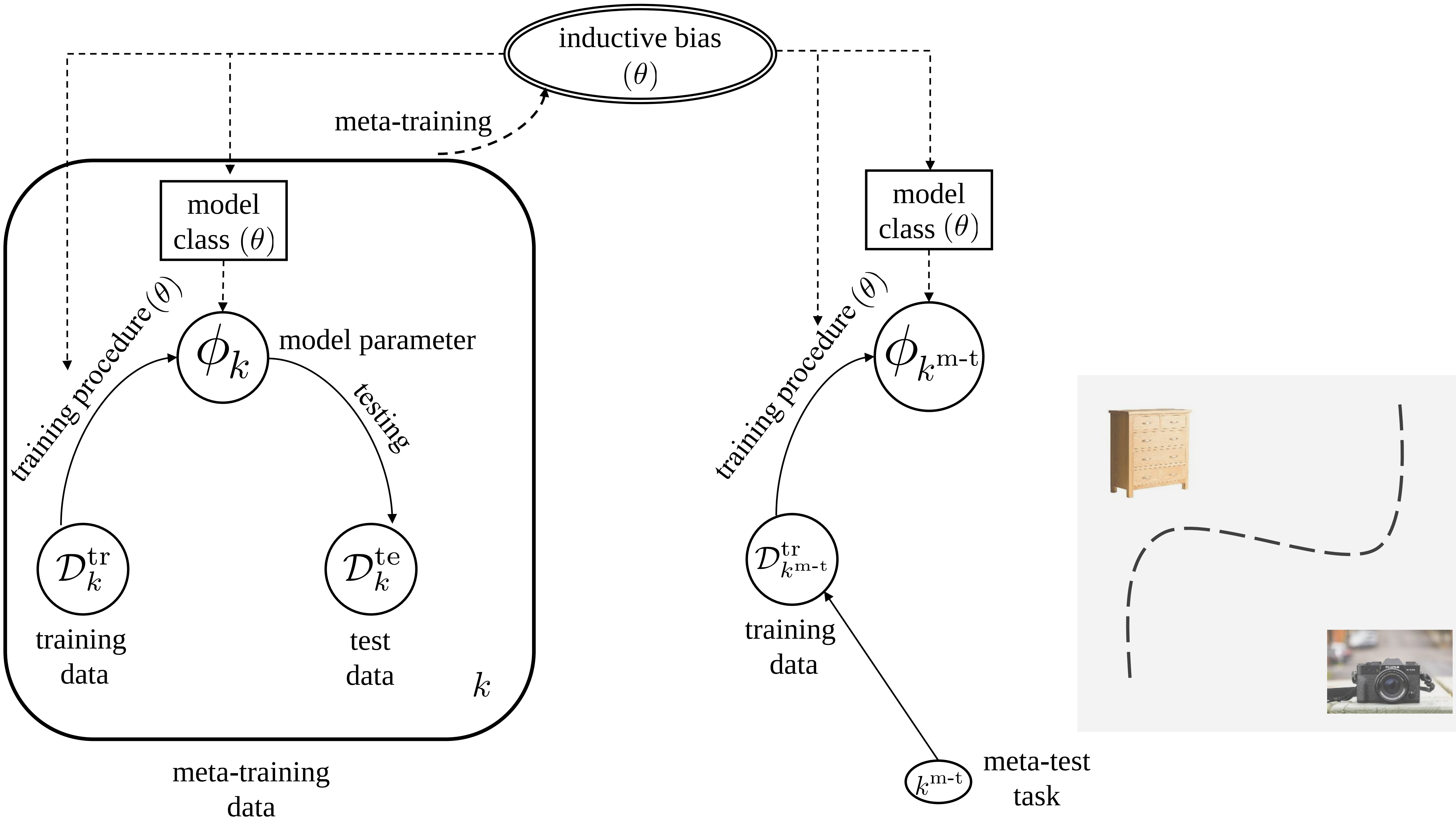
- Meta-learning finds an inductive bias that enables the training of accurate specialized models from few samples and/or with little complexity on each of the meta-training tasks…





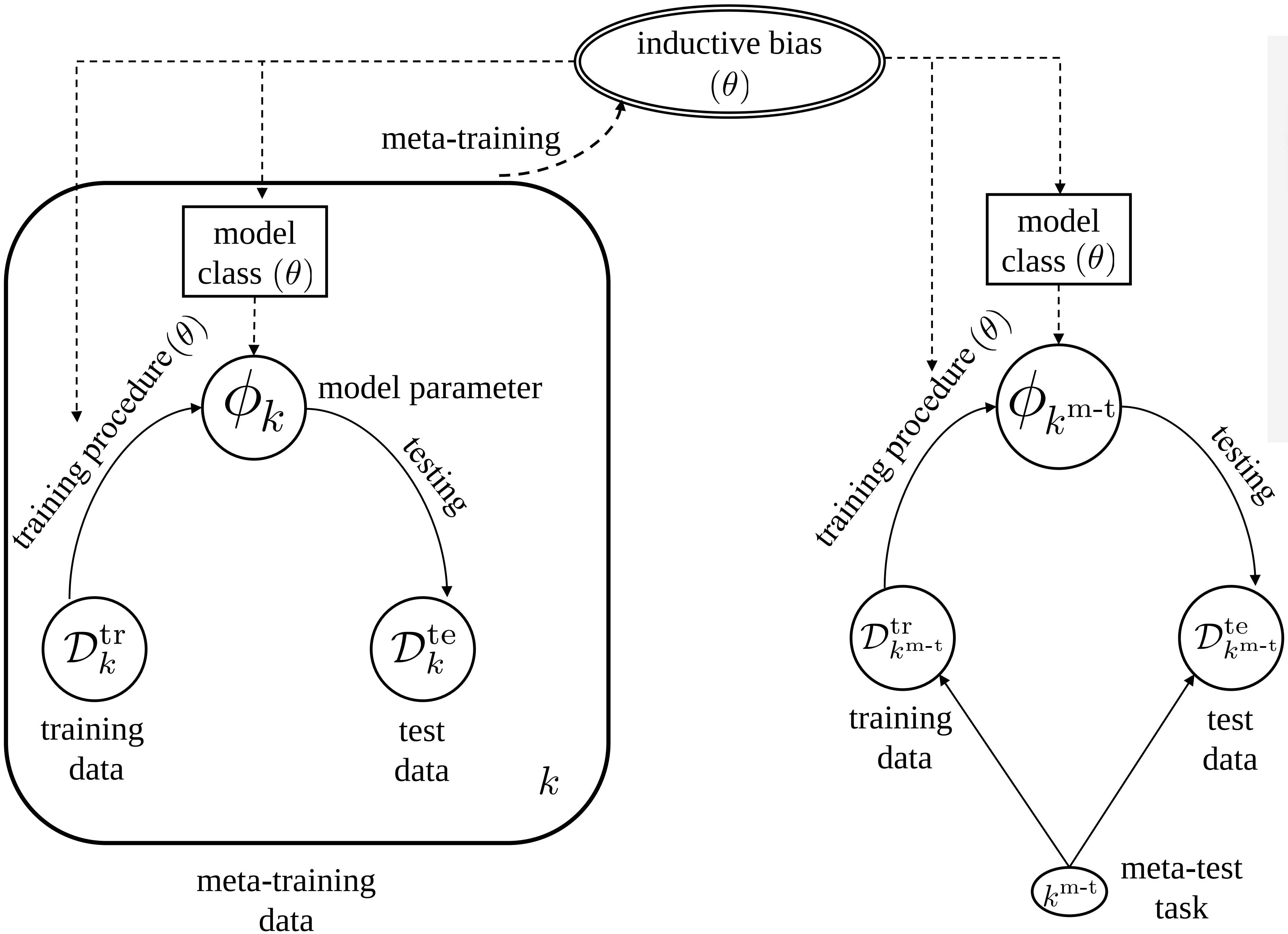
Meta-Learning

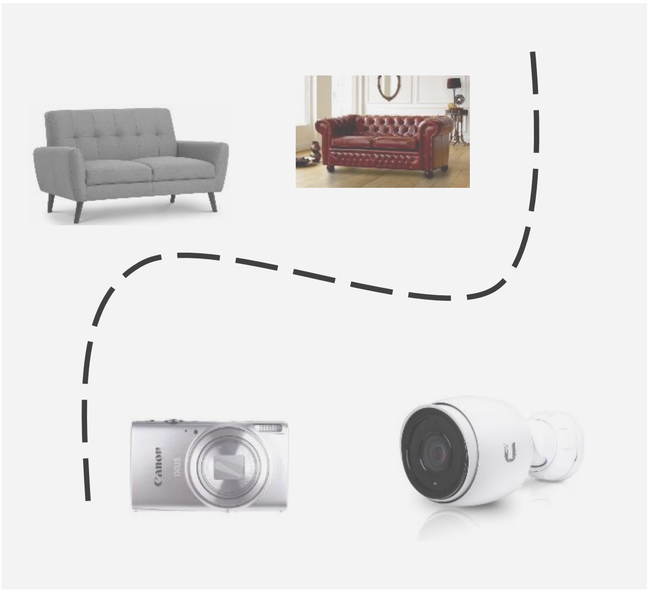
- … so that sample or iteration complexity for training new tasks are reduced



Meta-Learning

- … so that sample or iteration complexity for training new tasks are reduced
- Meta-training learns how to adapt, or how to learn





Meta-Learning

- Meta-learned inductive bias:
  - representation (i.e., feature extraction) [Vinyals et al ’16]
  - use of memory [Santoro et al ‘16]
  - learning rate [Maclaurin et al ’15]
  - non-linear gradient-based updates [Bengio et al ’90] [Wichrowska et al ’17]
  - initialization [Finn et al ’17]

Relationship with Transfer and Multi-Task Learning

	Training	Testing
Transfer Learning	Task 1	Task 2
Multi-task Learning	Task 1 ··· Task N	Task 1 ··· Task N
Meta-learning	Task 1 ··· Task N	Task N+1

Overview

- Motivation
- Meta-learning in a nutshell
- Algorithms and applications

S. Park, H. Jang, O. Simeone, and J. Kang, "Learning how to Demodulate from Few Pilots via Meta-Learning," Proc. IEEE SPAWC 2019.

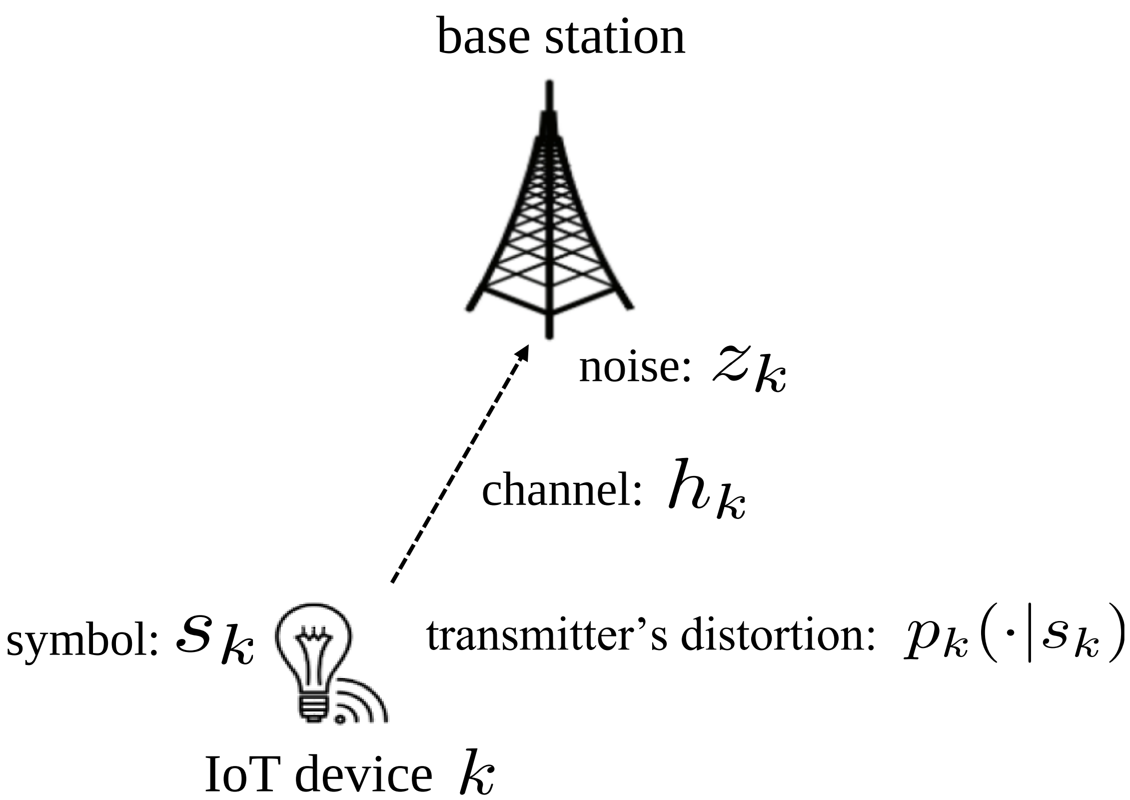
--. “Learning to Demodulate from Few Pilots via Offline and Online Meta-Learning,” arXiv:1908.09049.

--, "Meta-Learning to Communicate: Fast End-to-End Training for Fading Channels,“ Proc. ICASSP 2020

System Model

key offline meta-learning parameters

K: # of meta-training devices	N: # of pilots for meta-training data	$\mathcal{D}$: meta-training dataset
φ: neural network parameters	P: # of pilots for meta-test data	$\mathcal{D}_{\mathrm{T}}$: meta-test dataset



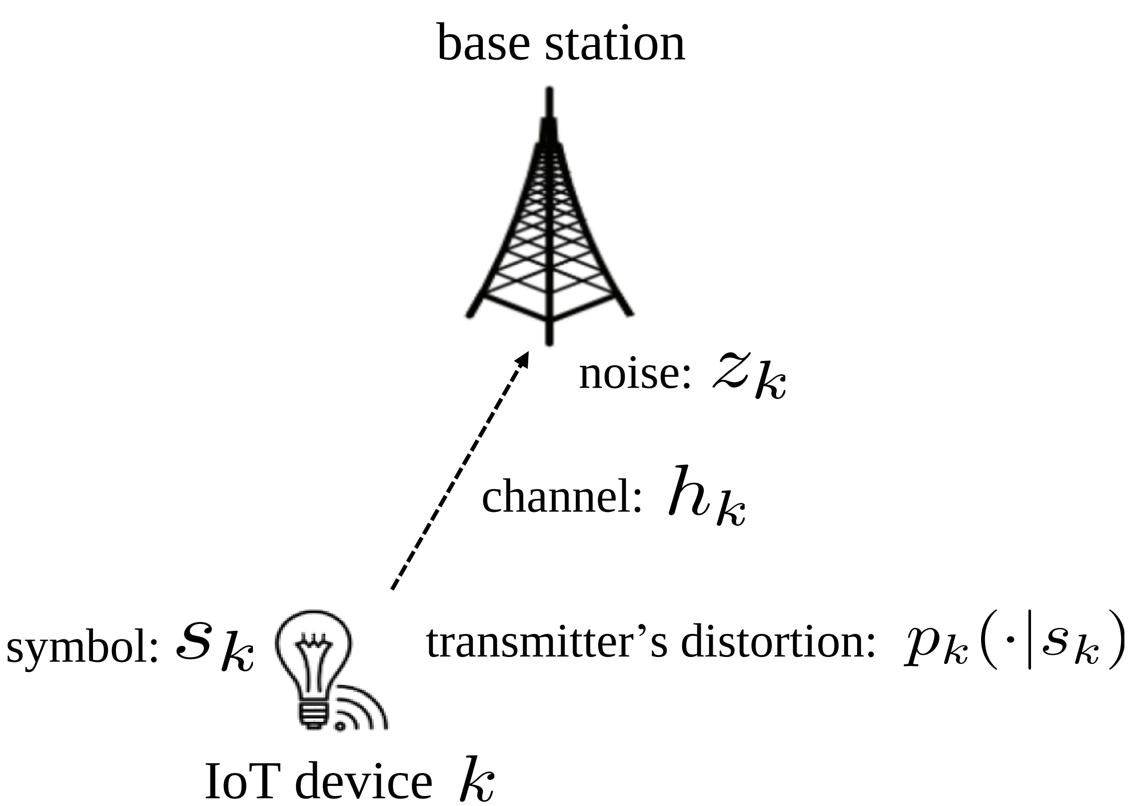
End-to-end channel for a device k

$$y_k = h_k x_k + z_k$$
$$x_k \sim p_k(\cdot|s_k)$$

System Model

key offline meta-learning parameters

K: # of meta-training devices
N: # of pilots for meta-training data
$\mathcal{D}$: meta-training dataset
φ: neural network parameters
P: # of pilots for meta-test data
$\mathcal{D}_{\mathrm{T}}$: meta-test dataset



End-to-end channel for a device k

$$y_k = h_k x_k + z_k$$
$$x_k \sim p_k(\cdot|s_k)$$

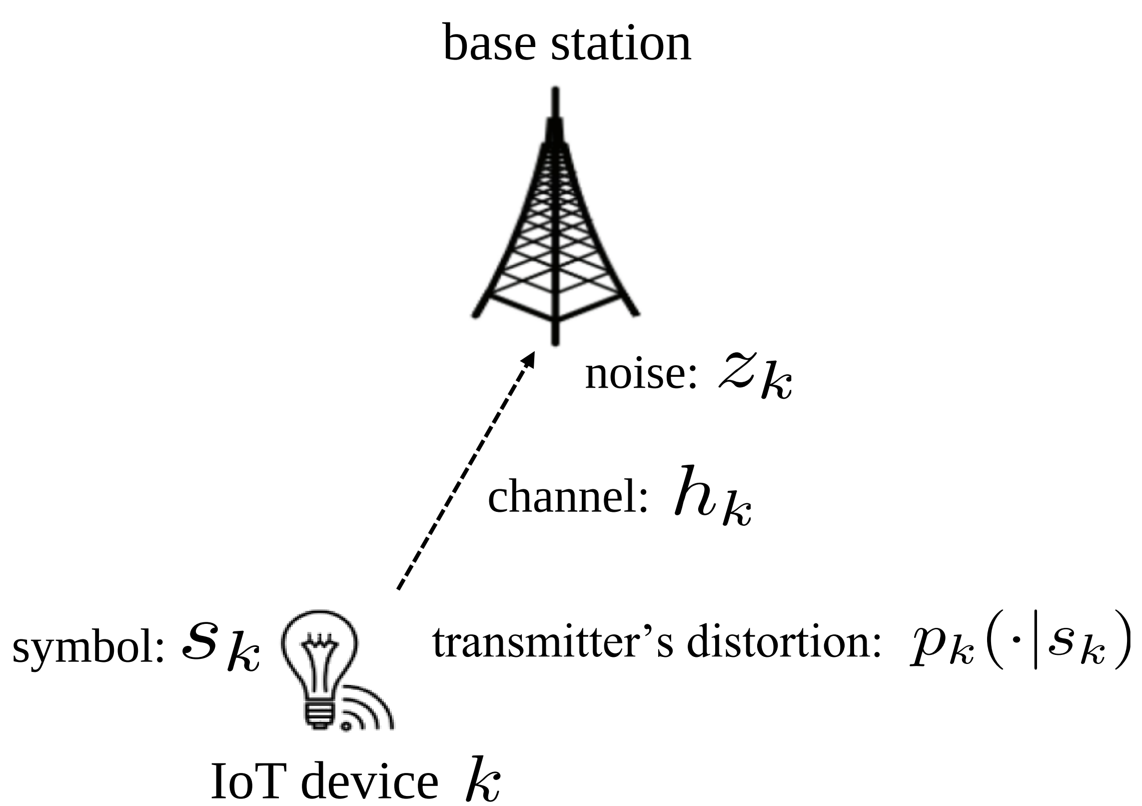
Parameterized demodulator (neural network)

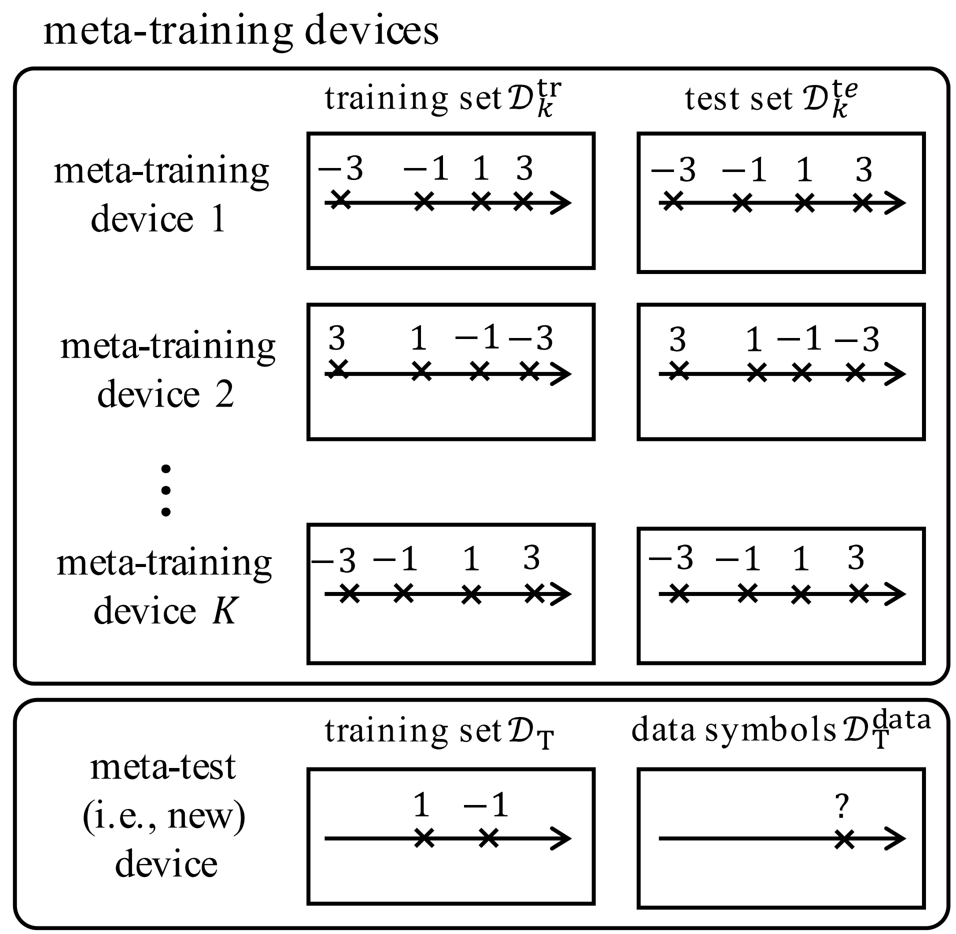
$$p(s|y, \varphi)$$

System Model

key offline meta-learning parameters

K: # of meta-training devices N: # of pilots for meta-training data $\mathcal{D}$: meta-training dataset
φ: neural network parameters P: # of pilots for meta-test data $\mathcal{D}_\mathrm{T}$: meta-test dataset





End-to-end channel for a device k

$$y_k = h_k x_k + z_k$$
$$x_k \sim p_k(\cdot|s_k)$$

Parameterized demodulator (neural network)

$$p(s|y,\varphi)$$

Meta-training dataset

$$\mathcal{D} = \{\mathcal{D}_k\}_{k=1,\ldots,K}$$
$$\mathcal{D}_k = \{(s_k^{(n)}, y_k^{(n)}) : n = 1, \ldots, N\}$$

Meta-test dataset

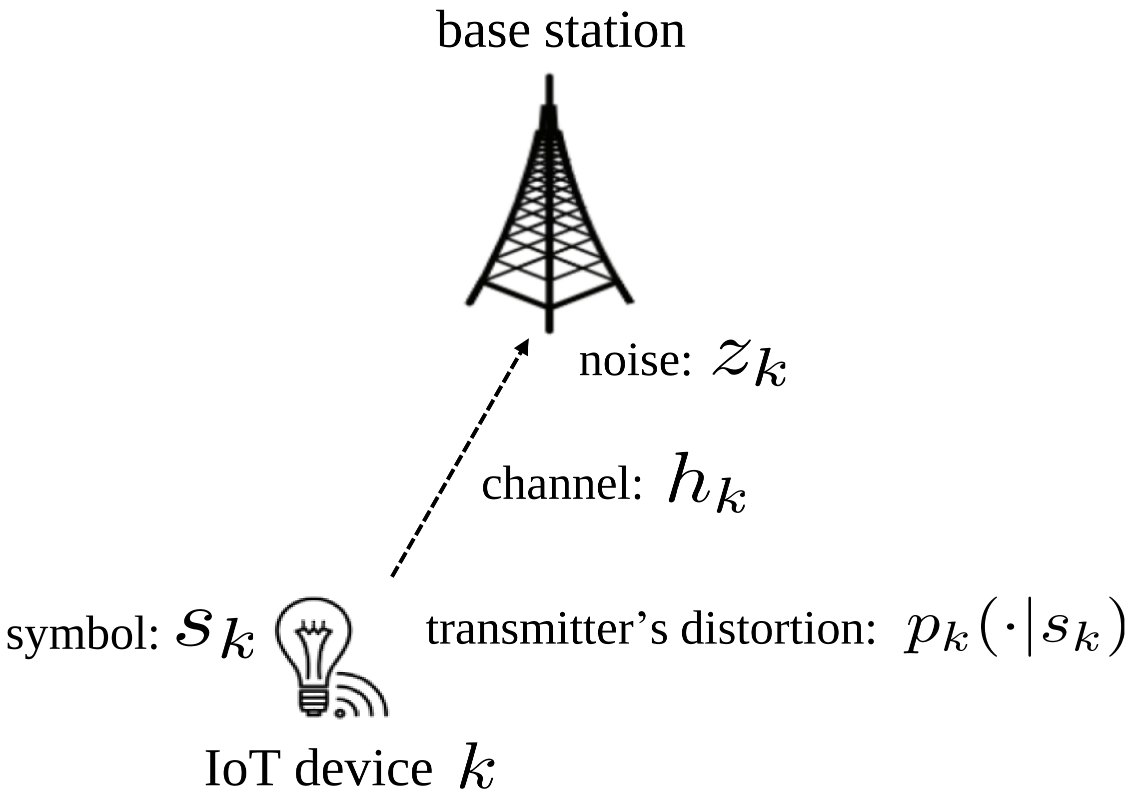
$$\mathcal{D}_\mathrm{T} = \{(s^{(n)}, y^{(n)}) : n = 1, \ldots, P\}$$

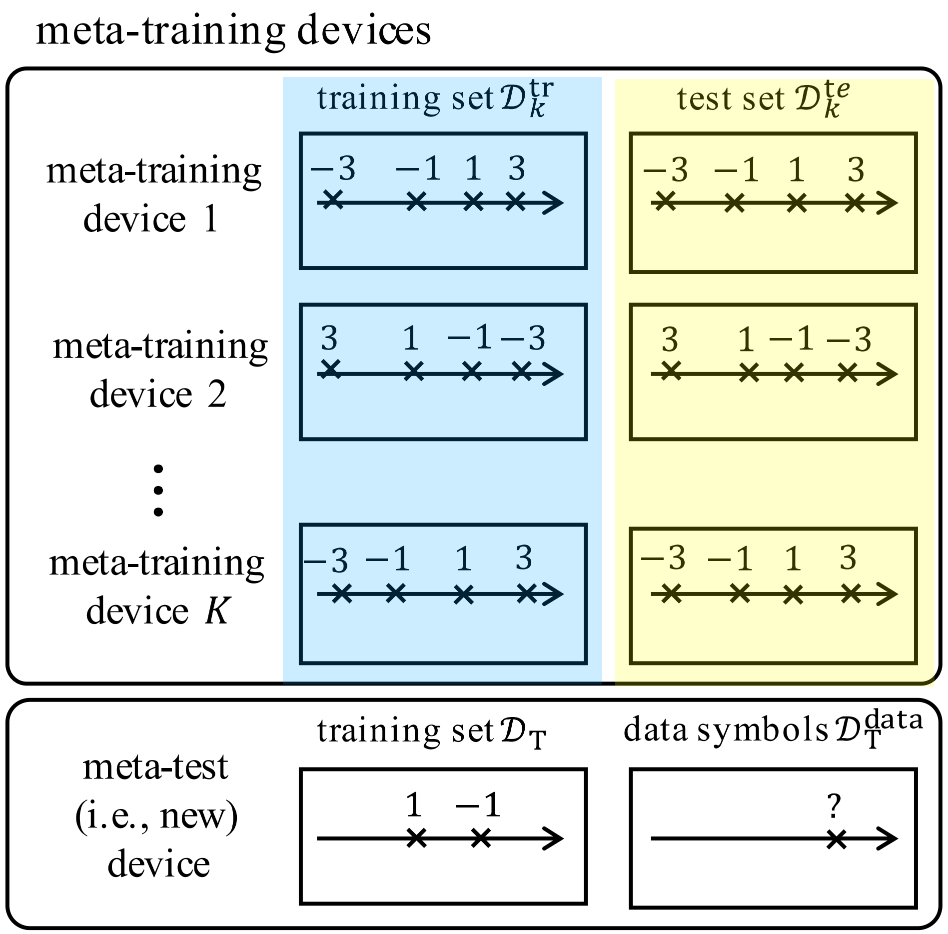
System Model

key offline meta-learning parameters

K: # of meta-training devices — N: # of pilots for meta-training data — $\mathcal{D}$: meta-training dataset

φ: neural network parameters — P: # of pilots for meta-test data — $\mathcal{D}_\mathrm{T}$: meta-test dataset





End-to-end channel for a device k

$$y_k = h_k x_k + z_k$$
$$x_k \sim p_k(\cdot|s_k)$$

Parameterized demodulator (neural network)

$$p(s|y,\varphi)$$

Meta-training dataset

$$\mathcal{D} = \{\mathcal{D}_k\}_{k=1,\dots,K}$$
$$\mathcal{D}_k = \{(s_k^{(n)}, y_k^{(n)}) : n = 1,\dots,N\}$$

Meta-test dataset

$$\mathcal{D}_\mathrm{T} = \{(s^{(n)}, y^{(n)}) : n = 1,\dots,P\}$$

Conventional Training

- Conventional training operates separately on the pilots of each device k.
- Pilots can be used for the supervised learning of a demodulator as a classifier.
- The training procedure aims at minimizing the generalization cross-entropy (surrogate of the probability of error)

$$L_k(\varphi) = \mathbb{E}_{(s,y)\sim p_k}[\underbrace{-\log p(s|y,\varphi)}_{\substack{\text{log-loss} \\ \text{(information-theoretic surprise)}}}]$$

Conventional Training

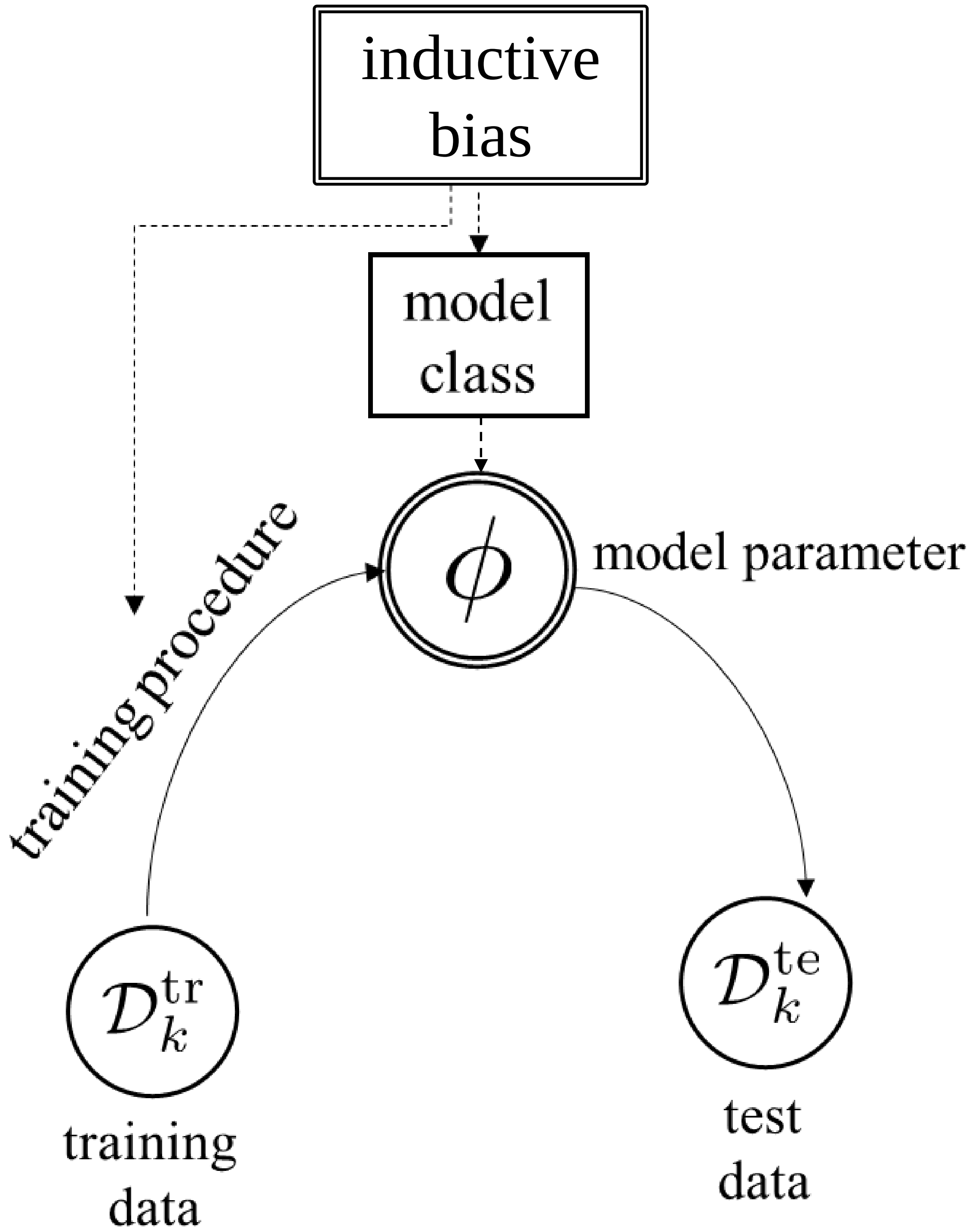
- Given a training data set $\mathcal{D}_k$, the ensemble loss is approximated by the training cross-entropy loss

$$L_{\mathcal{D}_k}(\varphi) = -\sum_{(s,y)\in\mathcal{D}_k} \log p(s|y,\varphi)$$

- Minimization is done via Stochastic GD (SGD)

$$\varphi \leftarrow \varphi - \eta \nabla_{\varphi} \log p(s|y,\varphi)$$

Conventional Training



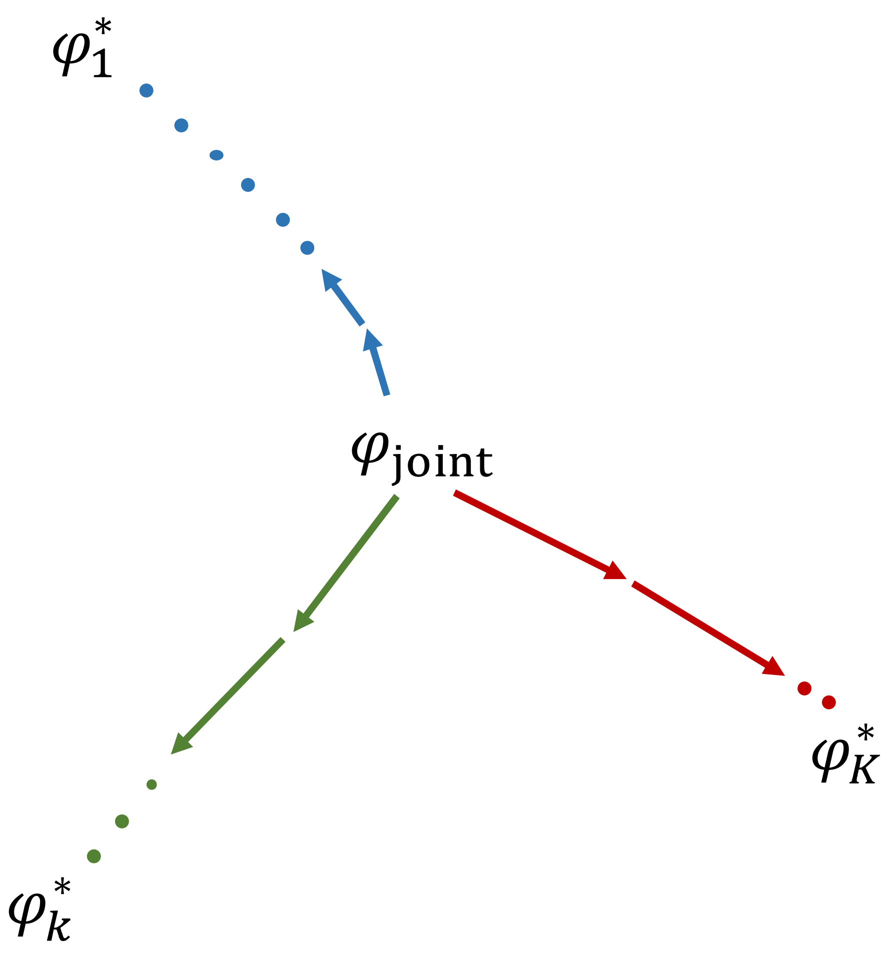
Joint Training

- In order to reduce the amount of data required for the new task, an intuitive solution would be to use joint training.

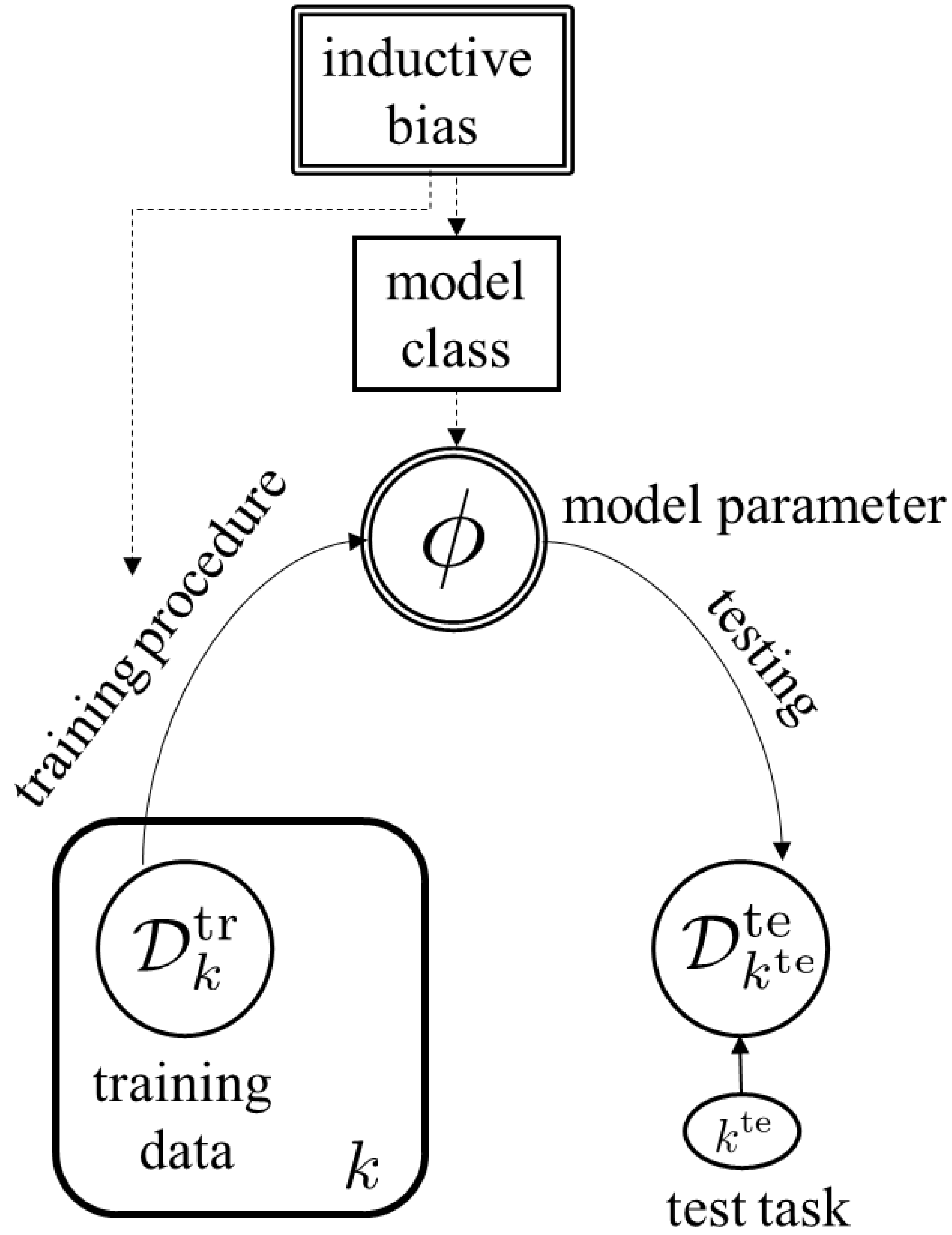
- Based on meta-training data $\mathcal{D}$, joint training minimizes

$$L_{\mathcal{D}}(\varphi) = -\sum_{(s,y)\in\mathcal{D}} \log p(s|y,\varphi)$$

- Joint training finds a single model that should perform well on all meta-training tasks/ devices.



Joint Training



Meta-Learning

- Shared hyperparameter θ → defines the inductive bias

- Task/ device-specific parameter ϕ

Meta-Learning

- Shared hyperparameter θ → defines the inductive bias

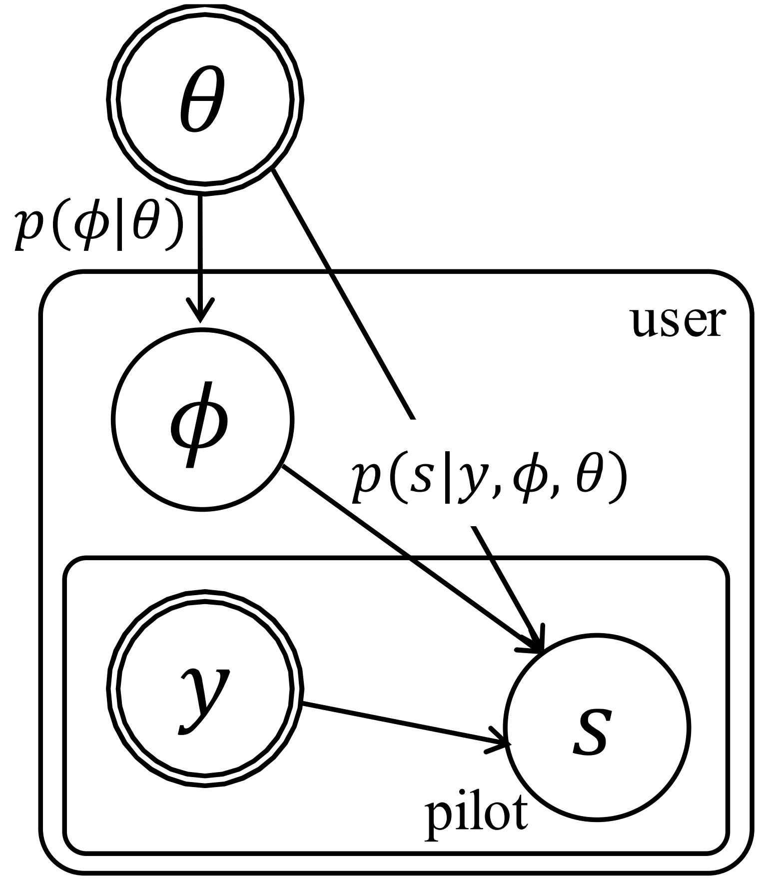
  meta-learned using meta-training data $\mathcal{D}$

- Task/ device-specific parameter ϕ

  from conventional training with inductive bias θ
  using task-specific data

Meta-Learning

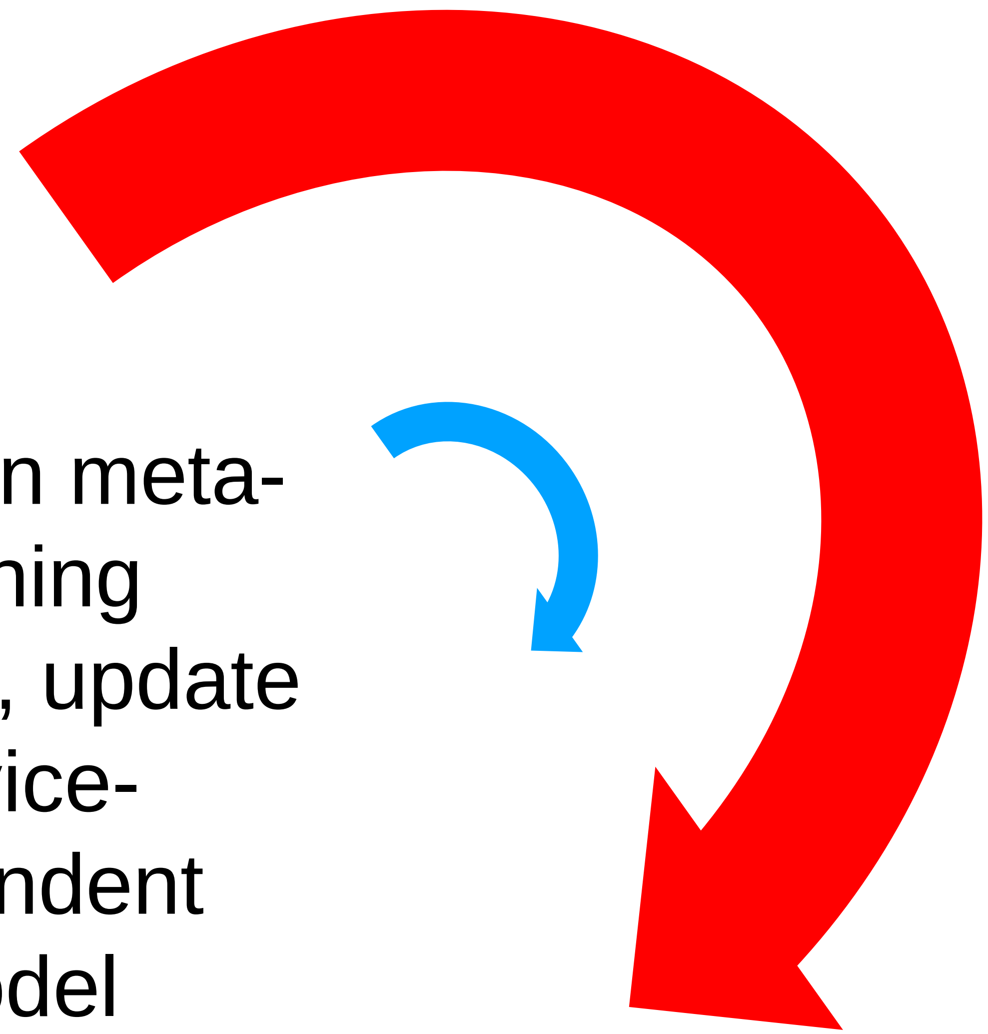
- Meta-learning algorithms can be derived as approximations of Expectation Maximization (EM) for hierarchical probabilistic models [Park et al '19].
- As for EM, they are organized around a nested loop.



Meta-Learning

for given meta-training devices, update device-dependent model parameter ϕ_k

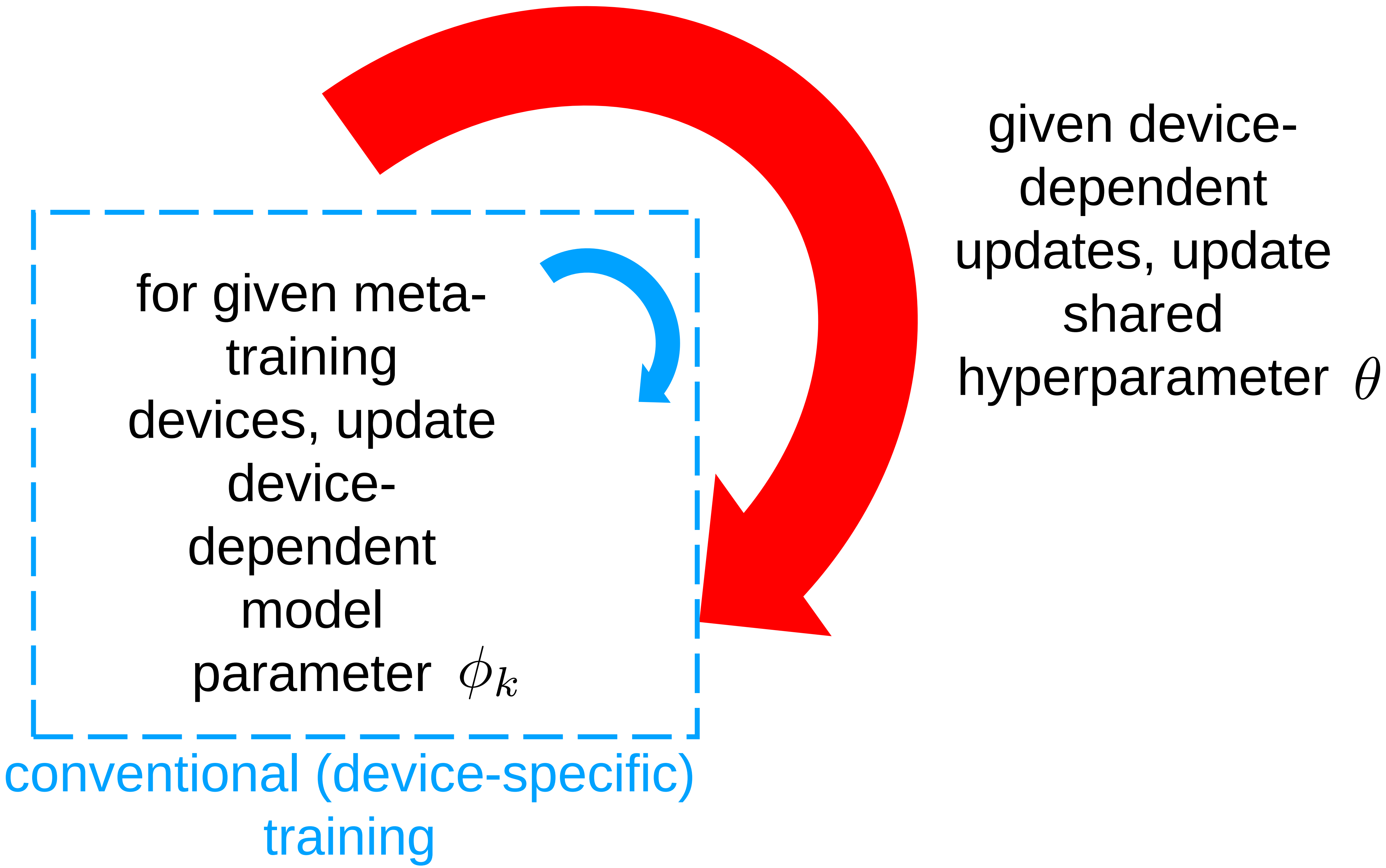
Meta-Learning



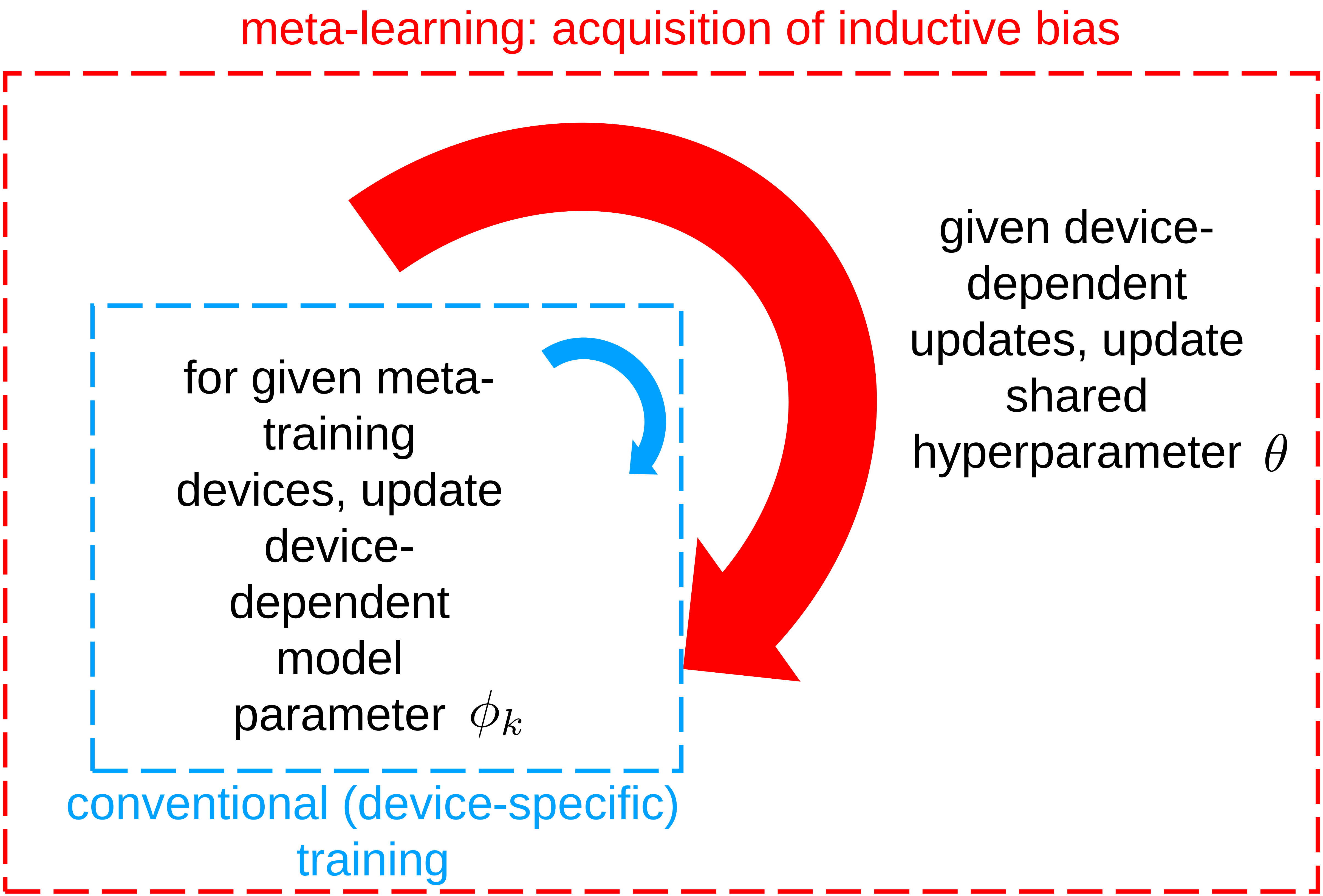
for given meta-training devices, update device-dependent model parameter ϕ_k

given device-dependent updates, update shared hyperparameter θ

Meta-Learning



Meta-Learning



MAML

- Model-Agnostic Meta-Learning [Finn et al '17]
  - Demodulator: $p(s|y,\phi)$
  - Device-specific parameter: $\varphi = \phi$
  - Shared hyperparameter: $\theta =$ initialization of local updates

local SGD
update

$$\phi_k = \boxed{\theta} - \eta \nabla_\theta L_{\mathcal{D}_k^{\mathrm{tr}}}(\theta)$$

(*m* SGD steps, here *m*=1)

MAML

- Model-Agnostic Meta-Learning [Finn et al ’17]
  - Demodulator: $p(s|y, \phi)$
  - Device-specific parameter: $\varphi = \phi$
  - Shared hyperparameter: $\theta =$ initialization of local updates

meta-update

$$\min_{\theta} \sum_{k=1}^{K} L_{\mathcal{D}_k^{\mathrm{te}}}(\phi_k)$$

local SGD update

$$\phi_k = \theta - \eta \nabla_\theta L_{\mathcal{D}_k^{\mathrm{tr}}}(\theta)$$

(*m* SGD steps, here *m*=1)

MAML

- Model-Agnostic Meta-Learning [Finn et al '17]
  - Demodulator: $p(s|y,\phi)$
  - Device-specific parameter: $\varphi = \phi$
  - Shared hyperparameter: $\theta =$ initialization of local updates

meta-update

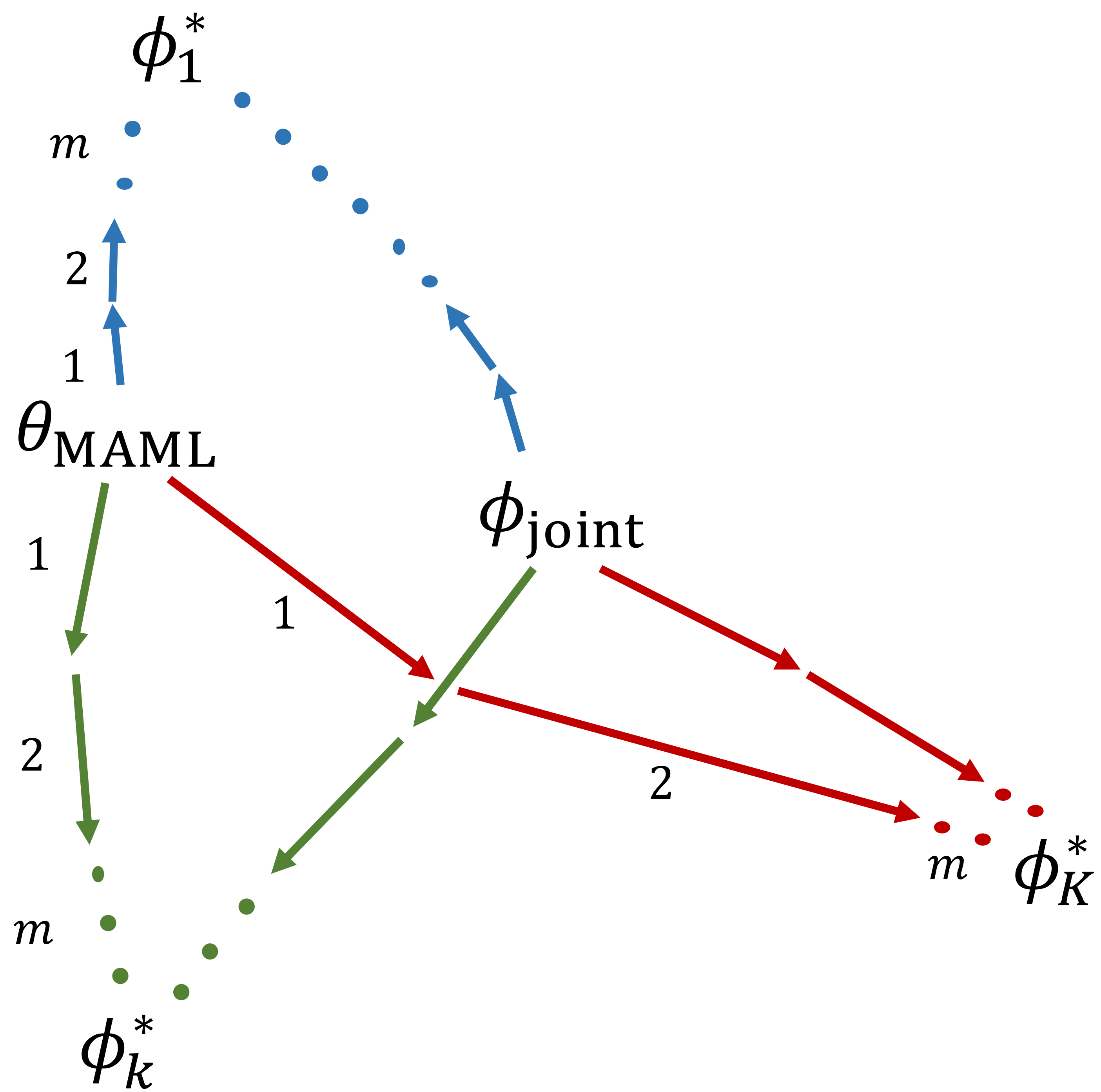
$$\min_{\theta} \sum_{k=1}^{K} L_{\mathcal{D}_k^{\mathrm{te}}}(\phi_k)$$

local SGD update

$$\phi_k = \theta - \eta \nabla_{\theta} L_{\mathcal{D}_k^{\mathrm{tr}}}(\theta)$$

(*m* SGD steps, here *m*=1)

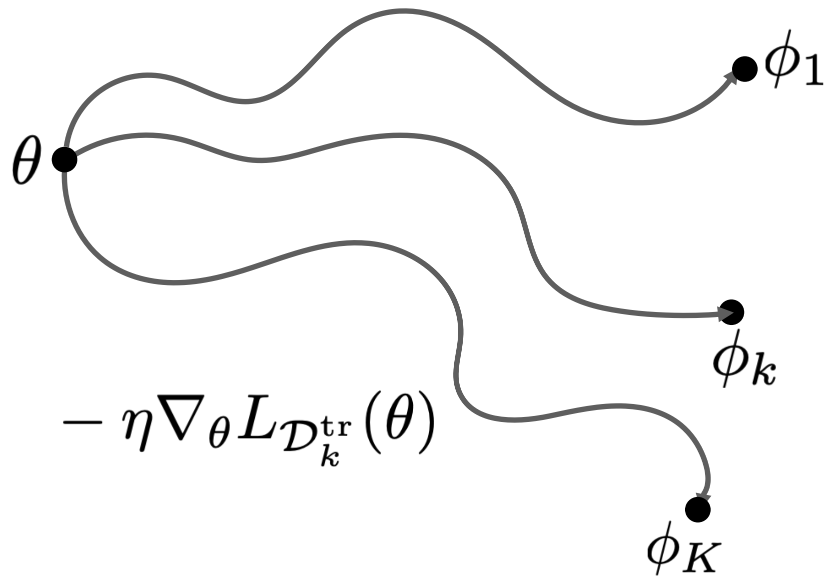
MAML



MAML

- Meta-update

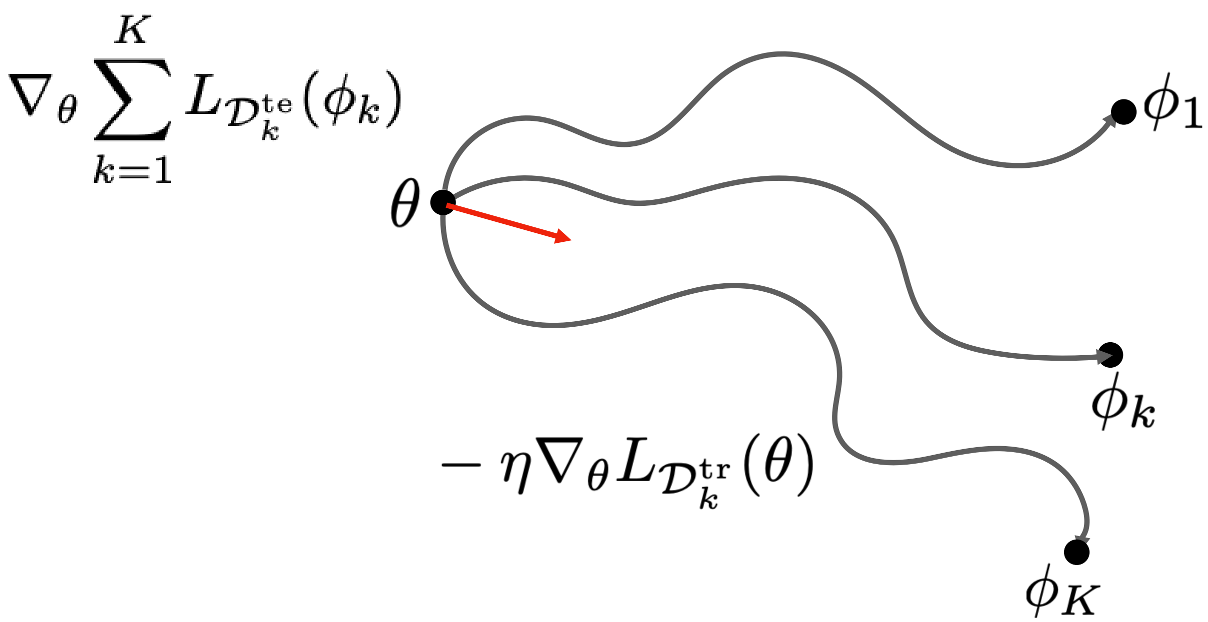
$$\min_{\theta} \sum_{k=1}^{K} L_{\mathcal{D}_k^{\text{te}}}(\phi_k)$$



MAML

- Meta-update

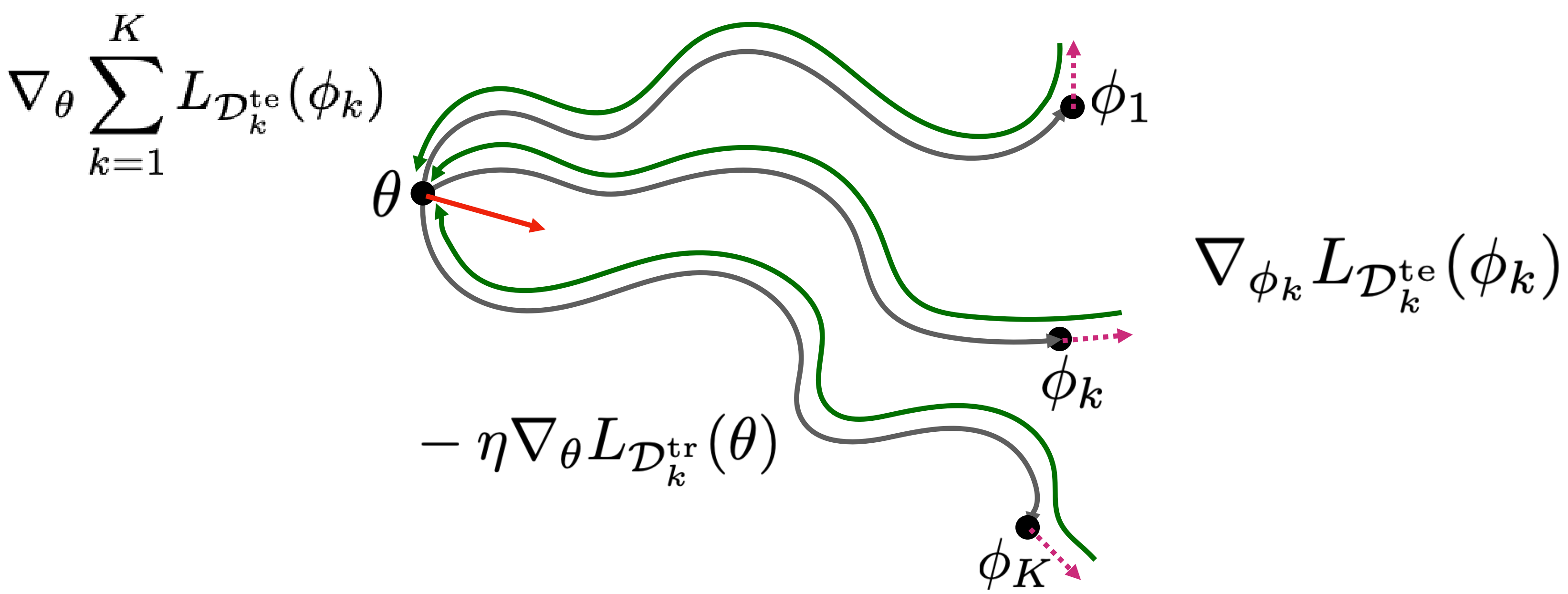
$$\min_{\theta} \sum_{k=1}^{K} L_{\mathcal{D}_k^{\mathrm{te}}}(\phi_k)$$



MAML

- Meta-update

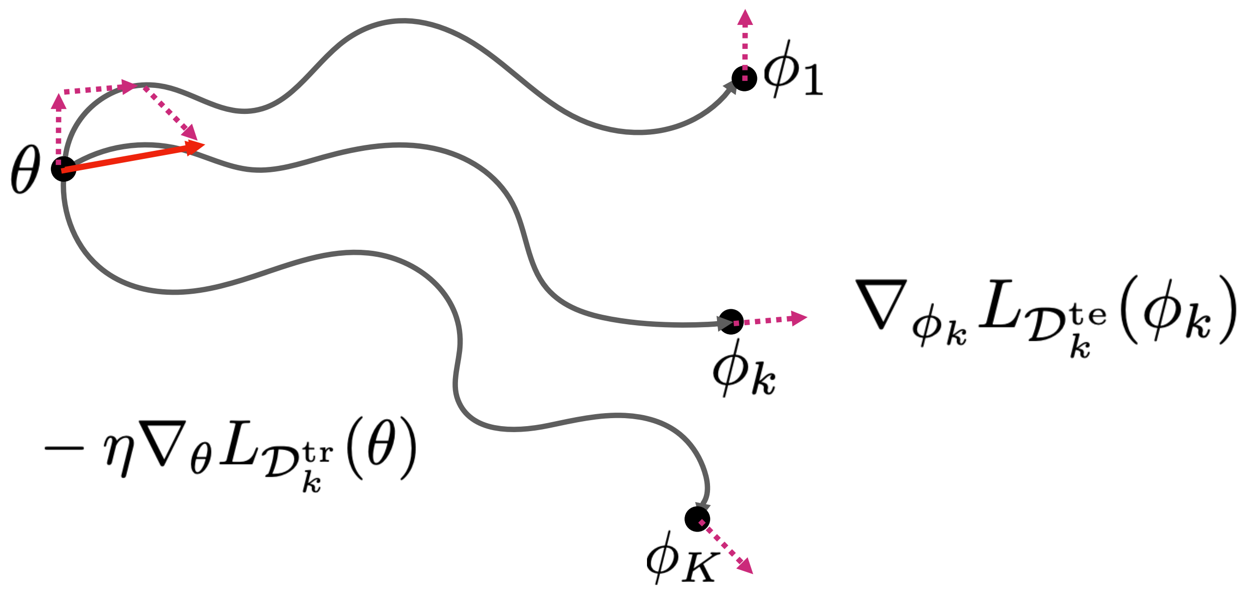
$$\min_{\theta} \sum_{k=1}^{K} L_{\mathcal{D}_k^{\text{te}}}(\phi_k)$$



$$\theta \leftarrow \theta - \kappa \sum_{k=1}^{K} (I - \eta \nabla_\theta^2 L_{\mathcal{D}_k^{\text{tr}}}(\theta)) \nabla_{\phi_k} L_{\mathcal{D}_k^{\text{te}}}(\phi_k)$$

FOMAML

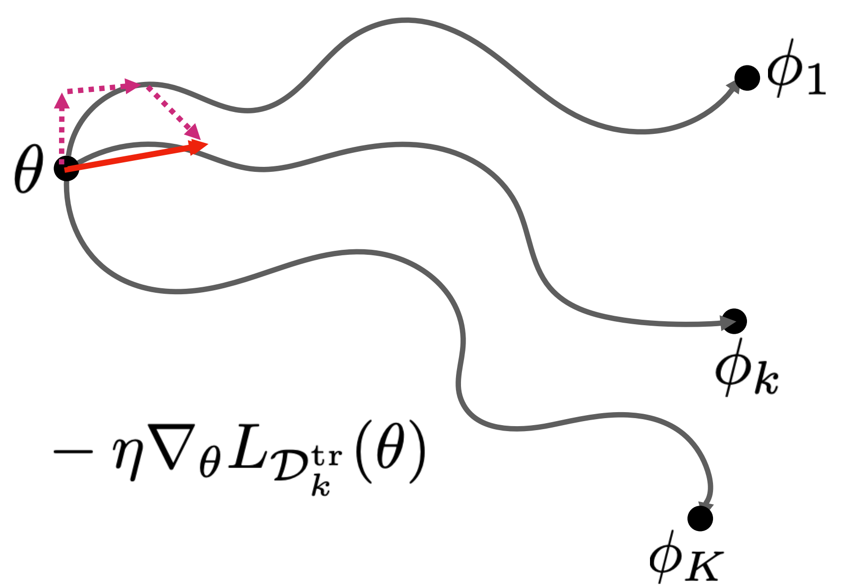
- First-Order Model-Agnostic Meta-Learning [Finn et al ’17]
  - Meta-update



$$\theta \leftarrow \theta - \kappa \nabla_{\phi_k} \sum_{k=1}^{K} L_{\mathcal{D}_k^{\text{te}}}(\phi_k)$$

REPTILE

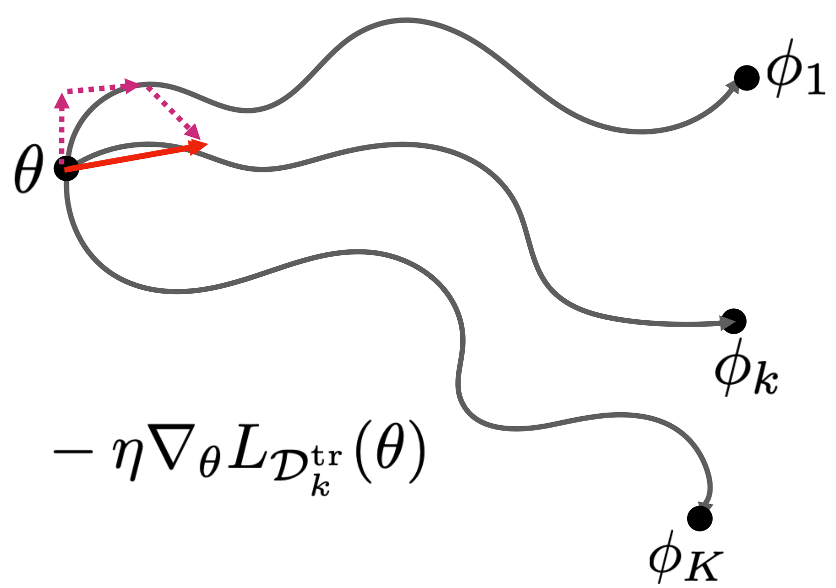
- REPTILE [Nichol et al ’18]
  - Meta-update



$$\theta \leftarrow (1-\kappa)\theta - \kappa\sum_{k=1}^{K}\phi_k$$

REPTILE

- REPTILE [Nichol et al ’18]
  - Meta-update

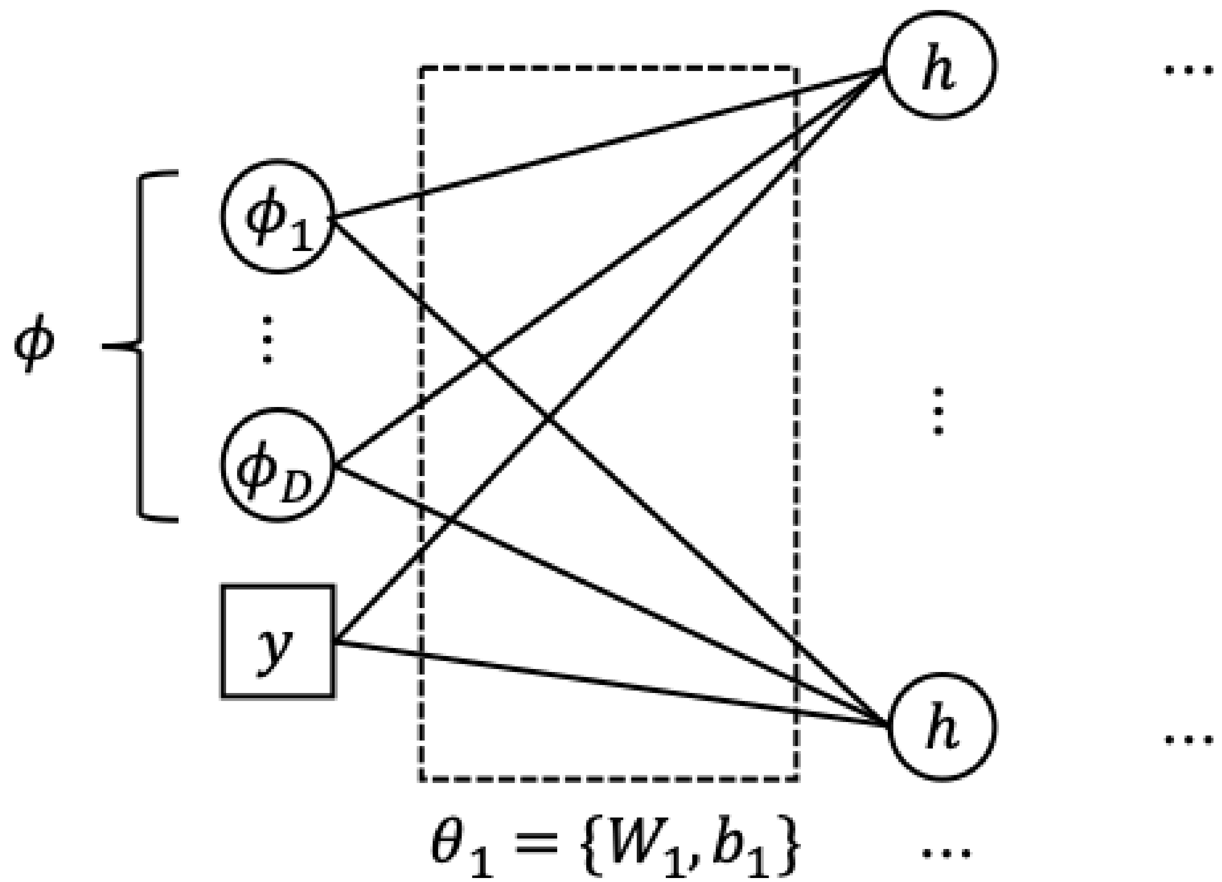


$$\theta \leftarrow (1-\kappa)\theta - \kappa\sum_{k=1}^{K}\phi_k$$

… equivalent to Federated Averaging [McMahan et al ’17]

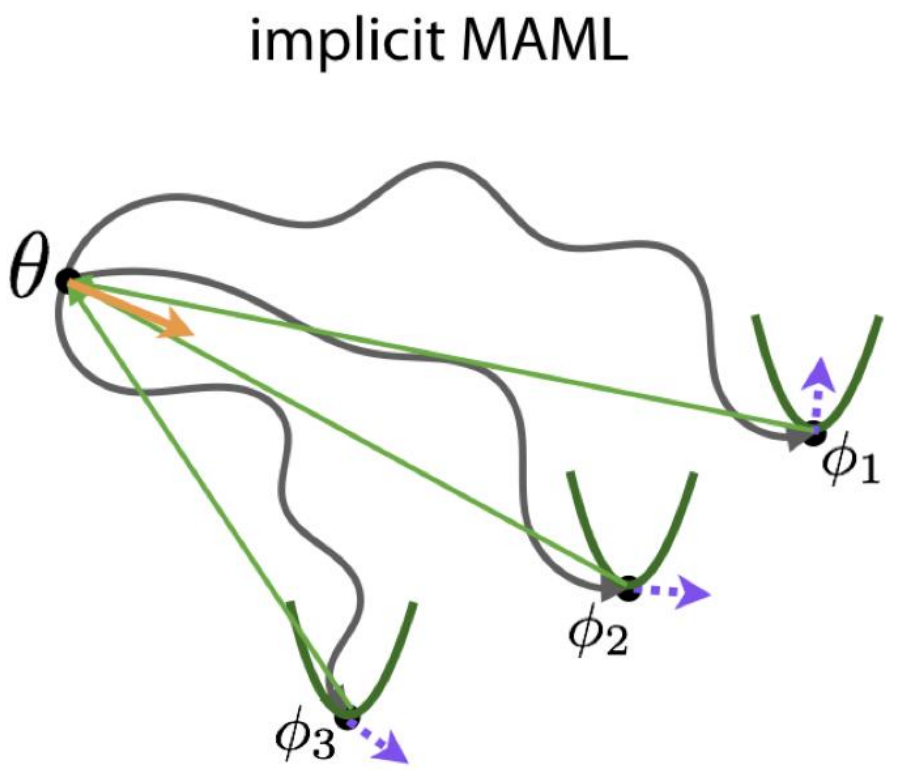
CAVIA

- fast Context Adaptation VIA meta-learning [Zintgraf et al '19]
  - Demodulator: $p(s|\tilde{y}, \theta)$, $\tilde{y} = [y, \phi]$, ϕ: additional input
  - Shared parameter: $\varphi = \theta$

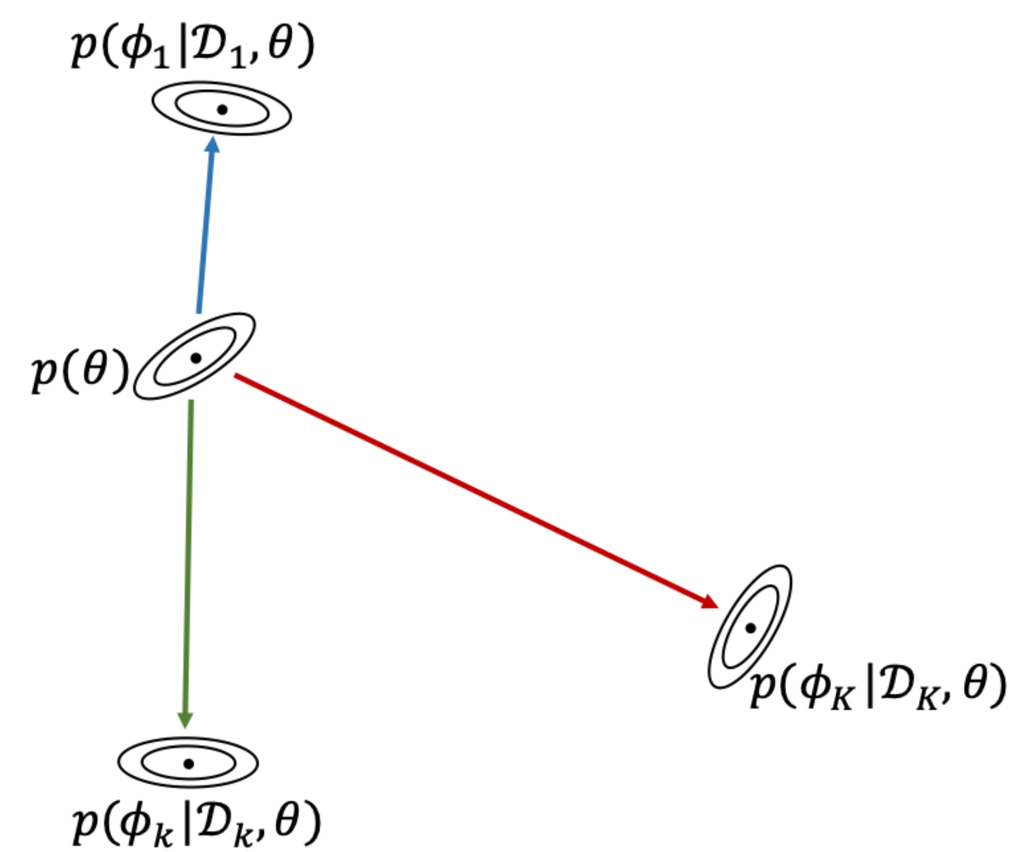


… And Many More

- Very active field with daily updates: T-MAML [Liu et al ’19], modular meta-learning [Chen et al ’19], implicit gradients [Rajeswaran et al ’19], zeroth-order MAML [Song et al ’19],…

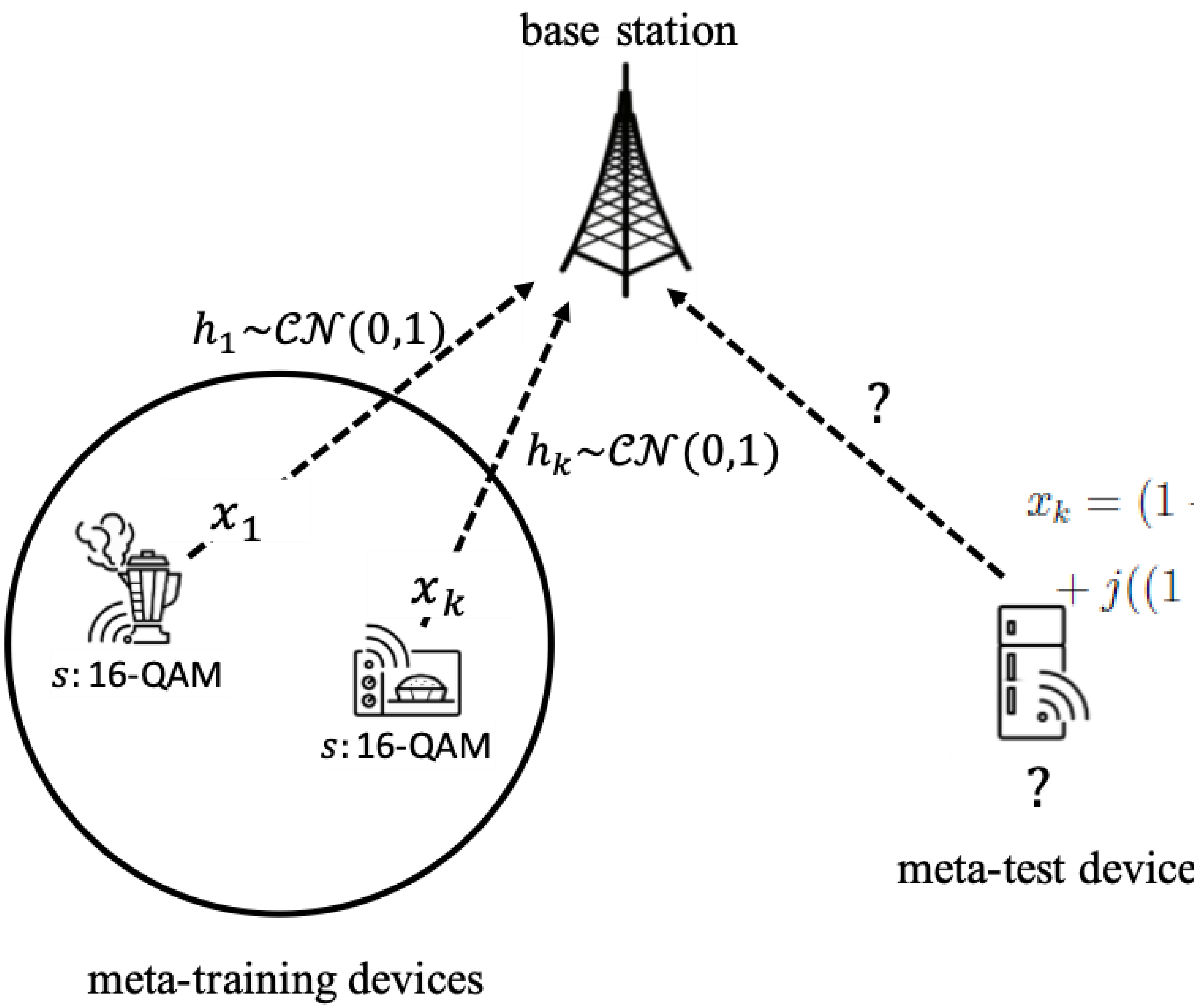


- Probabilistic approach [Finn et al ’18], [Ravi and Beatson ’19],[Gordon et al ’19], [Nguyen et al ’19]:
  - Instead of point estimate, approximate posterior distribution in E-step
  - Can add prior for shared parameter



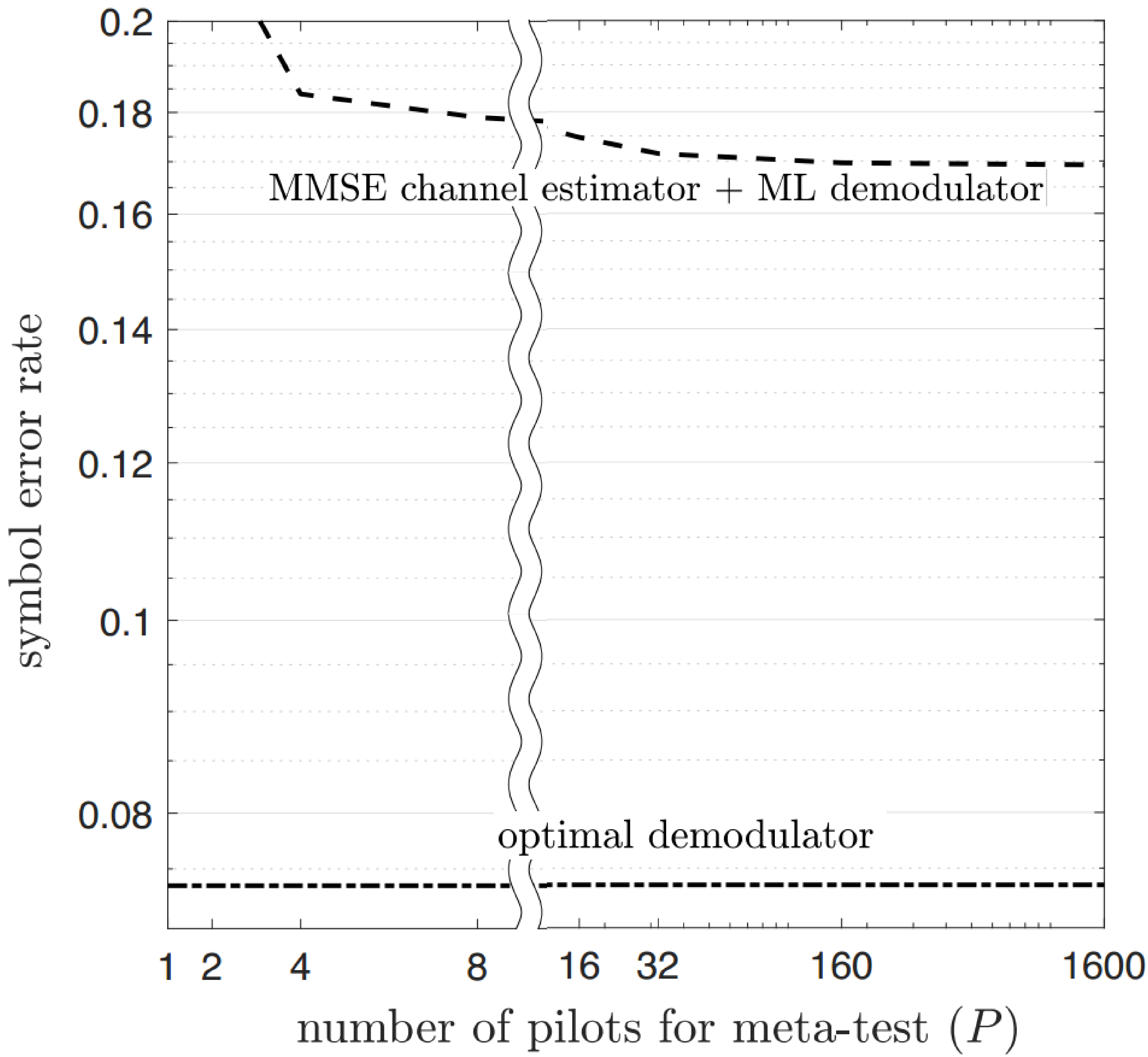
Experiments

- I/Q imbalance at the transmitters and Rayleigh fading channels with 16-QAM

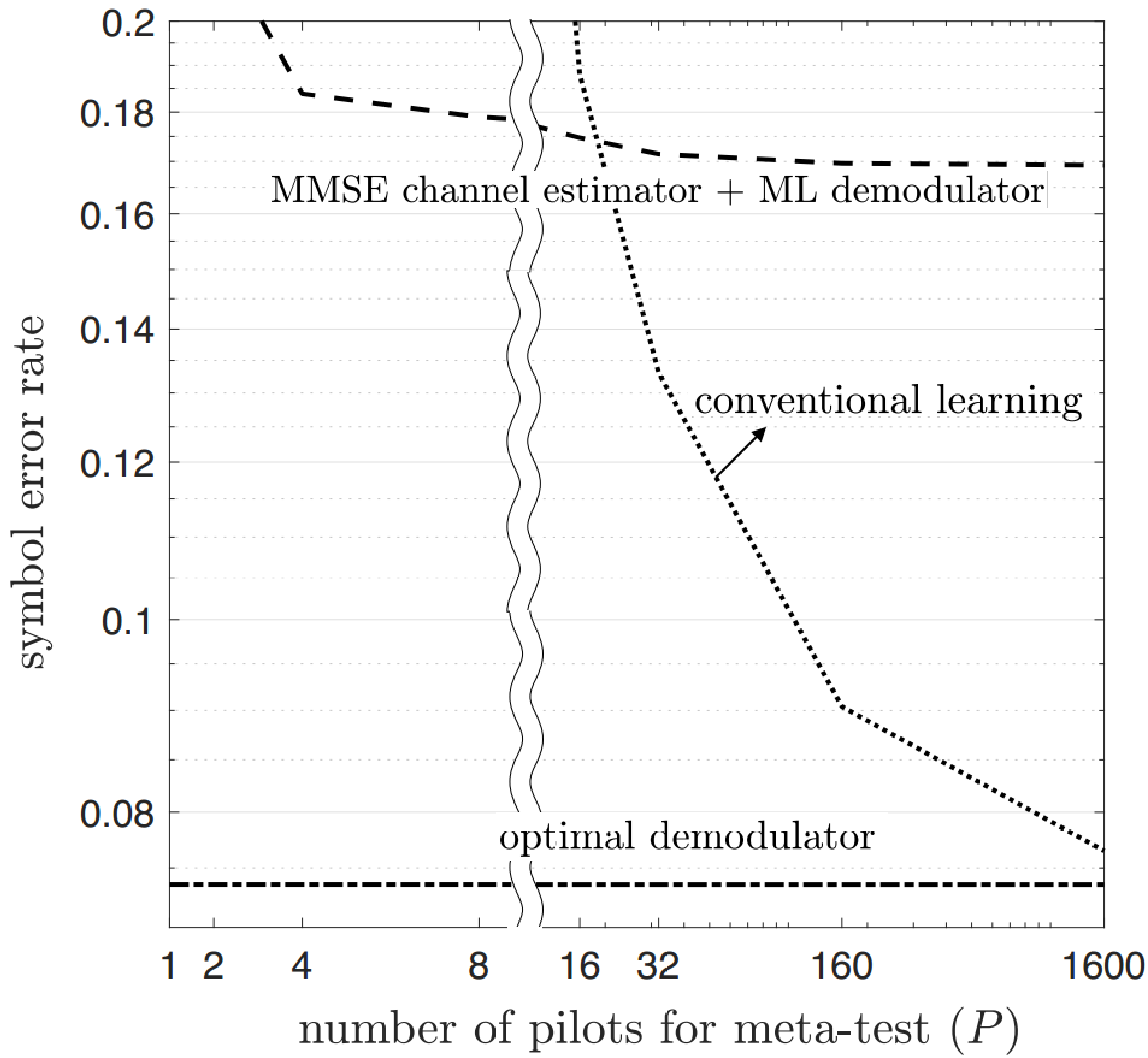


$$x_k = (1+\epsilon_k)\cos\delta_k \mathrm{Re}\{s_k\} - (1+\epsilon_k)\sin\delta_k \mathrm{Im}\{s_k\}$$
$$+ j((1-\epsilon_k)\cos\delta_k \mathrm{Im}\{s_k\} - (1-\epsilon_k)\sin\delta_k \mathrm{Re}\{s_k\})$$

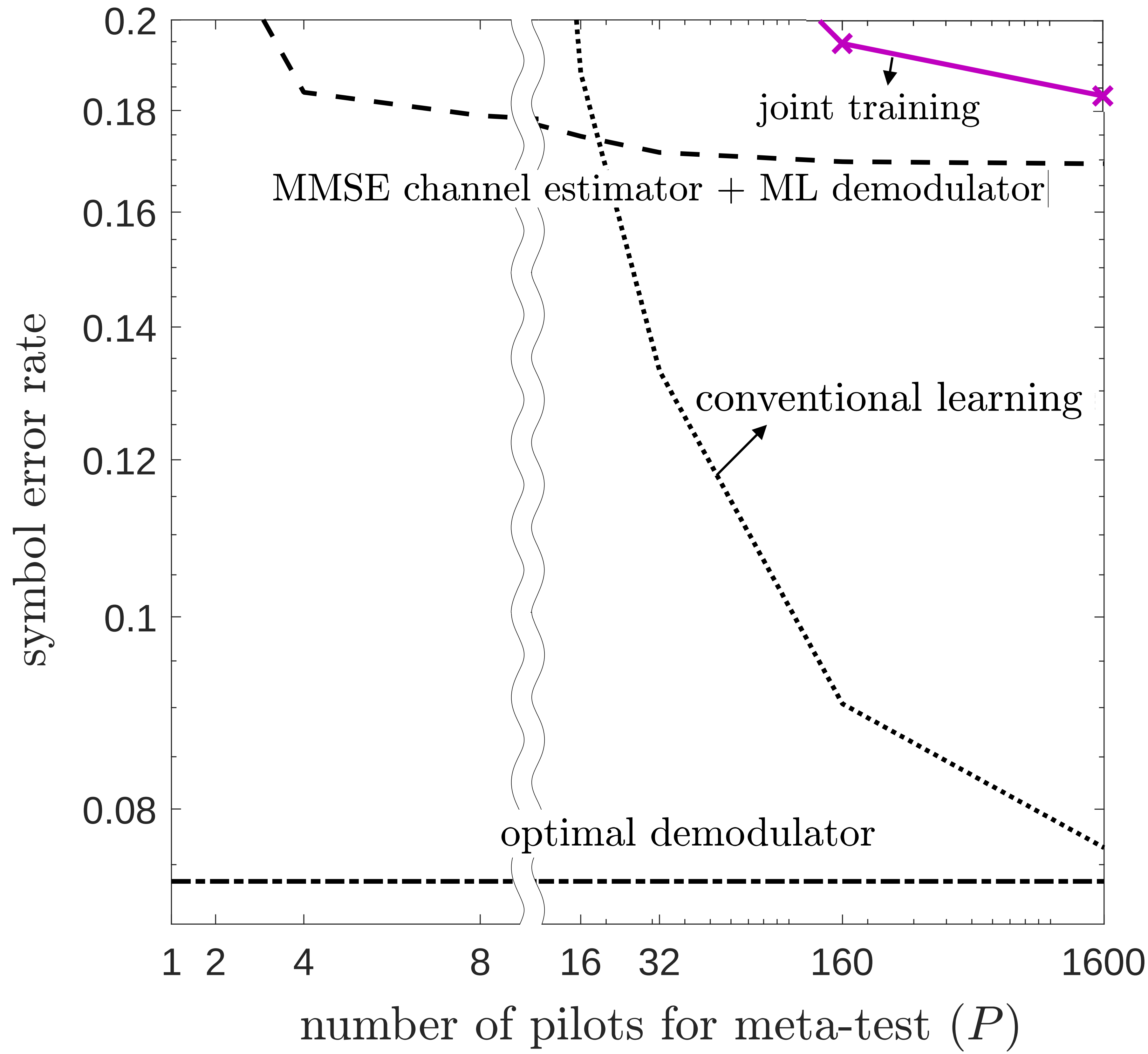
Experiments



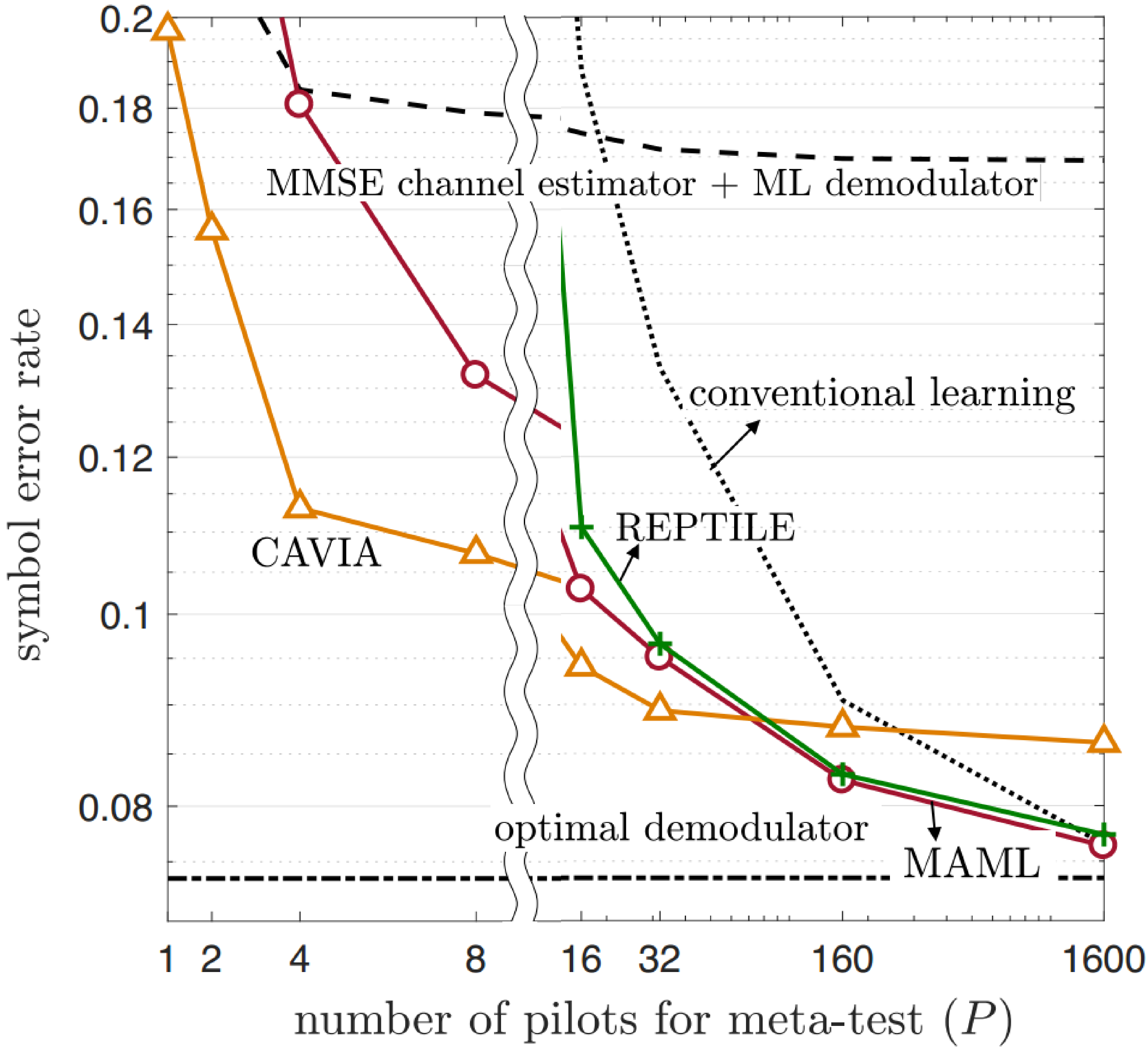
Experiments



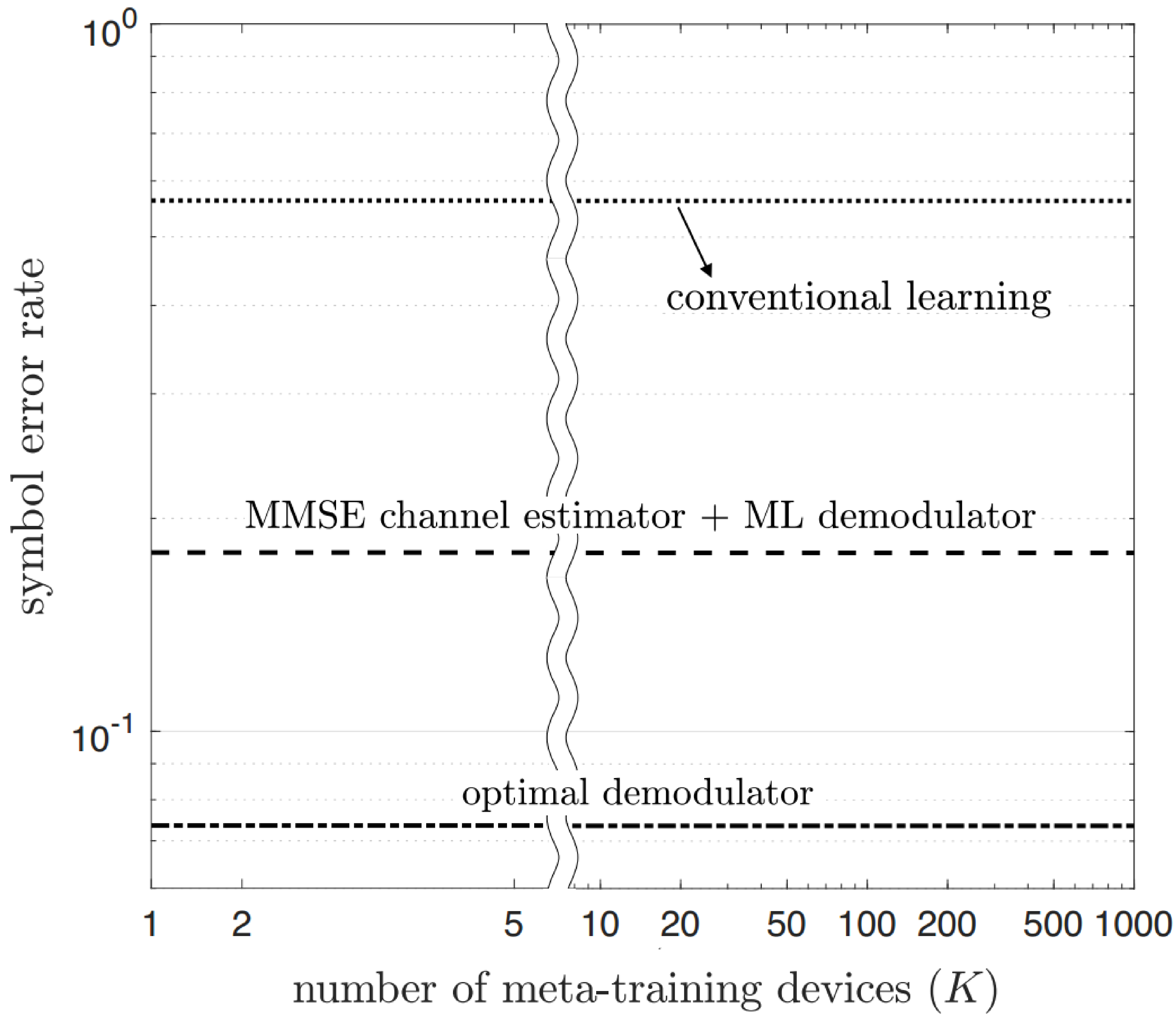
Experiments



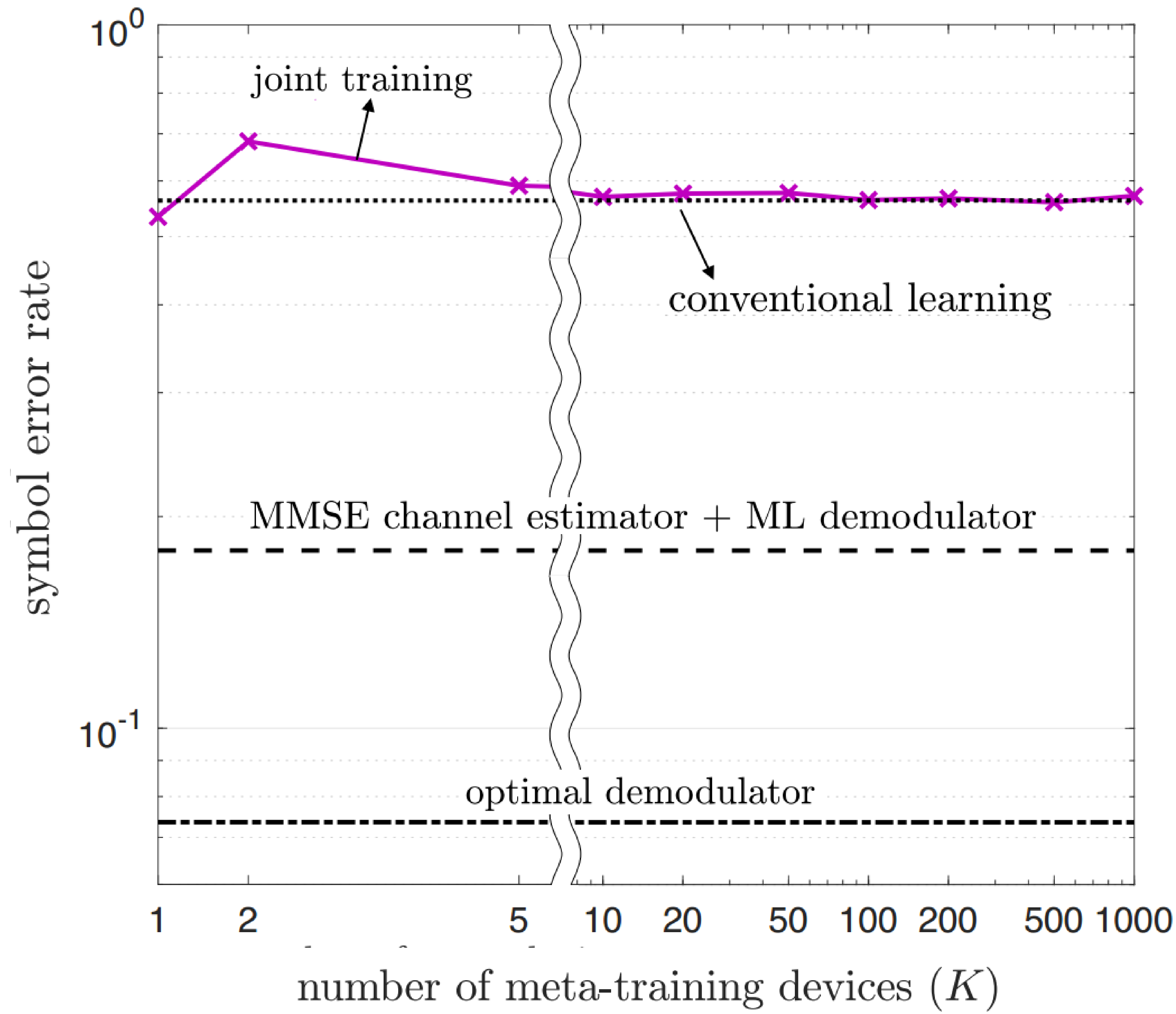
Experiments



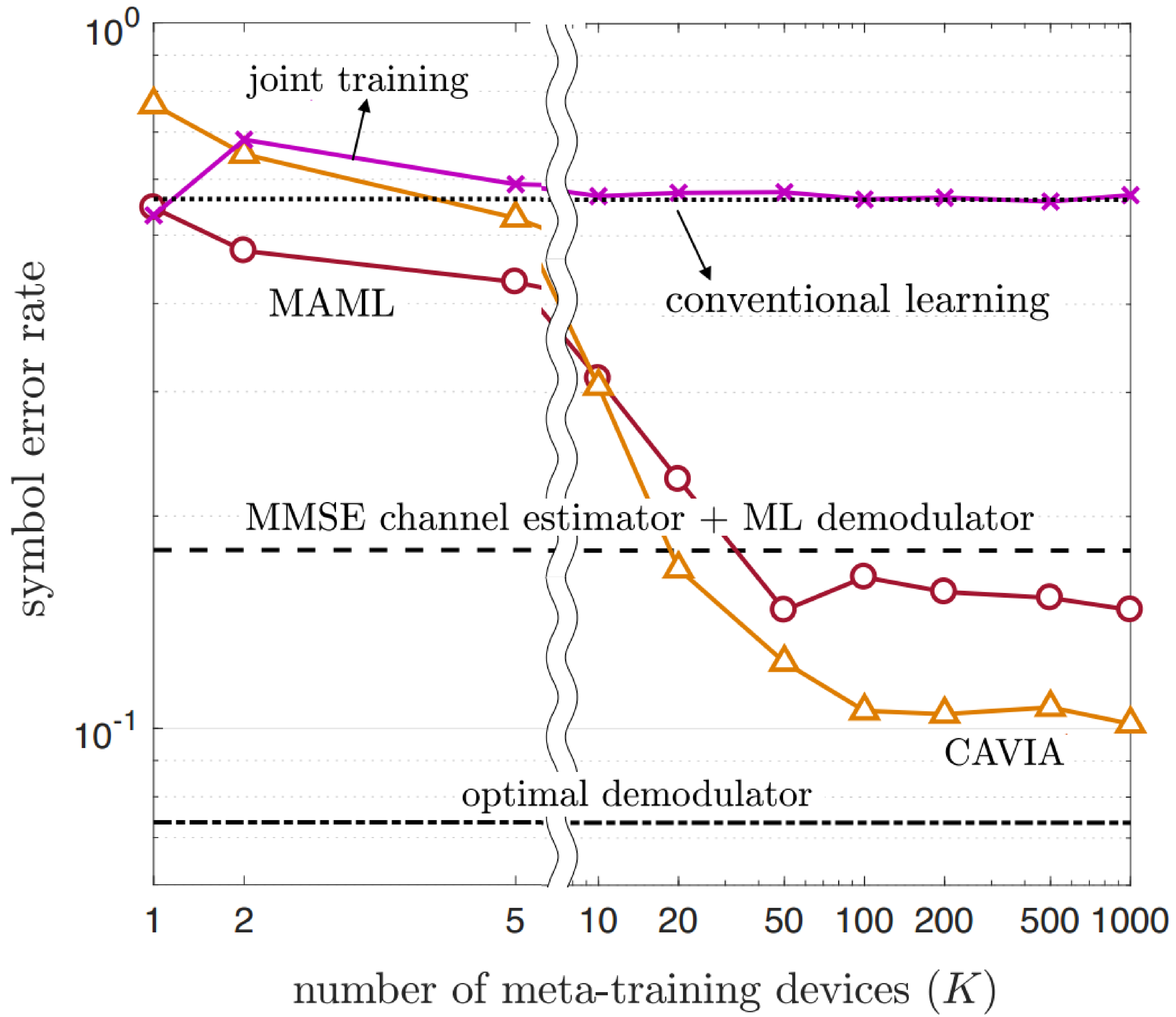
Experiments



Experiments



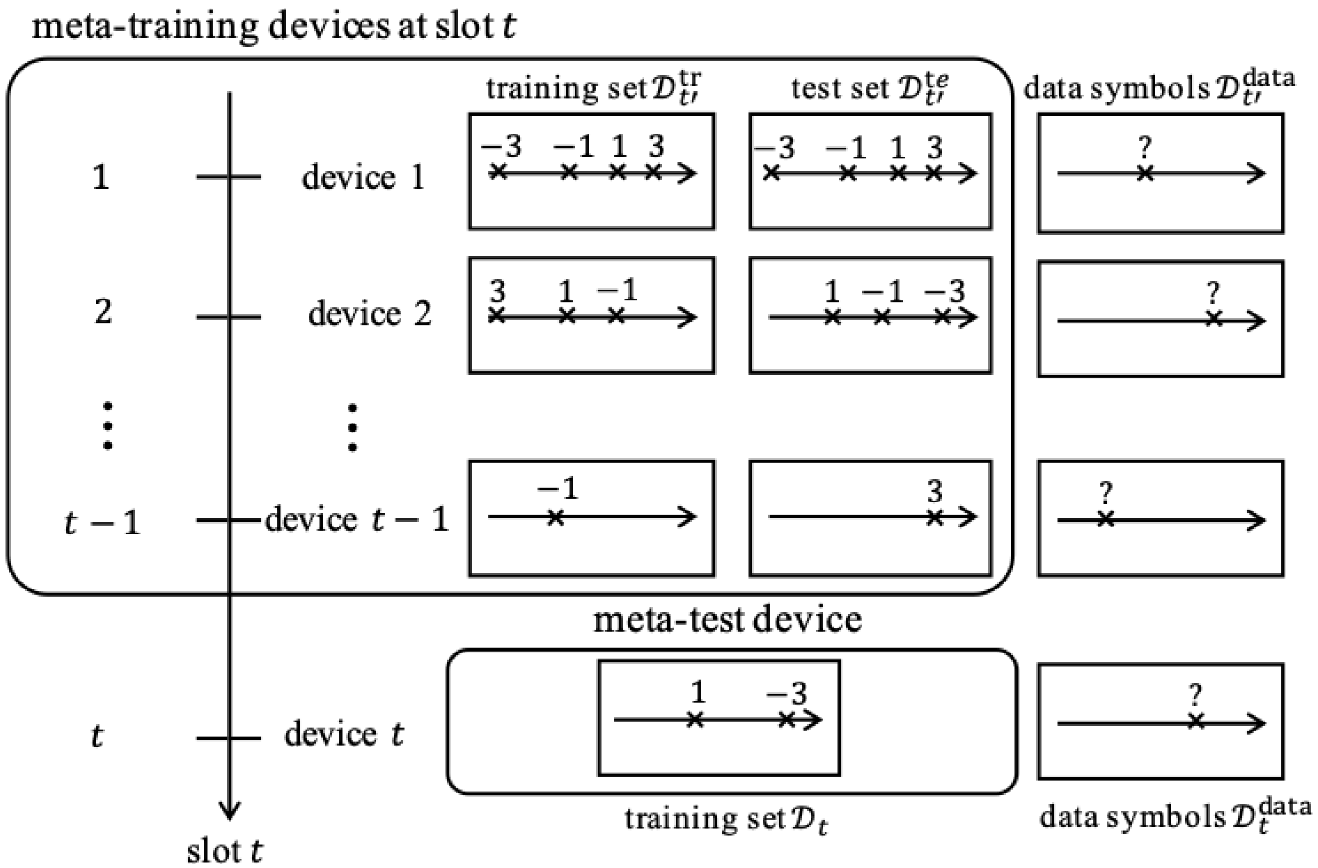
Experiments



Online Meta-Learning

key online meta-learning parameters

t: current time slot
$t-1$: # of meta-training devices
P_t: # of pilots for current time slot
P: maximum # of pilots
$\mathcal{D}^{t-1}$: meta-training dataset
$\mathcal{D}_t$: meta-test dataset



Meta-training dataset

$$\mathcal{D}^{t-1} = \{\mathcal{D}_{t'}\}_{t'=1}^{t-1}$$

$$\mathcal{D}_t = \{(s_t^{(n)}, y_t^{(n)}) : n = 1, \ldots, P_t\}$$

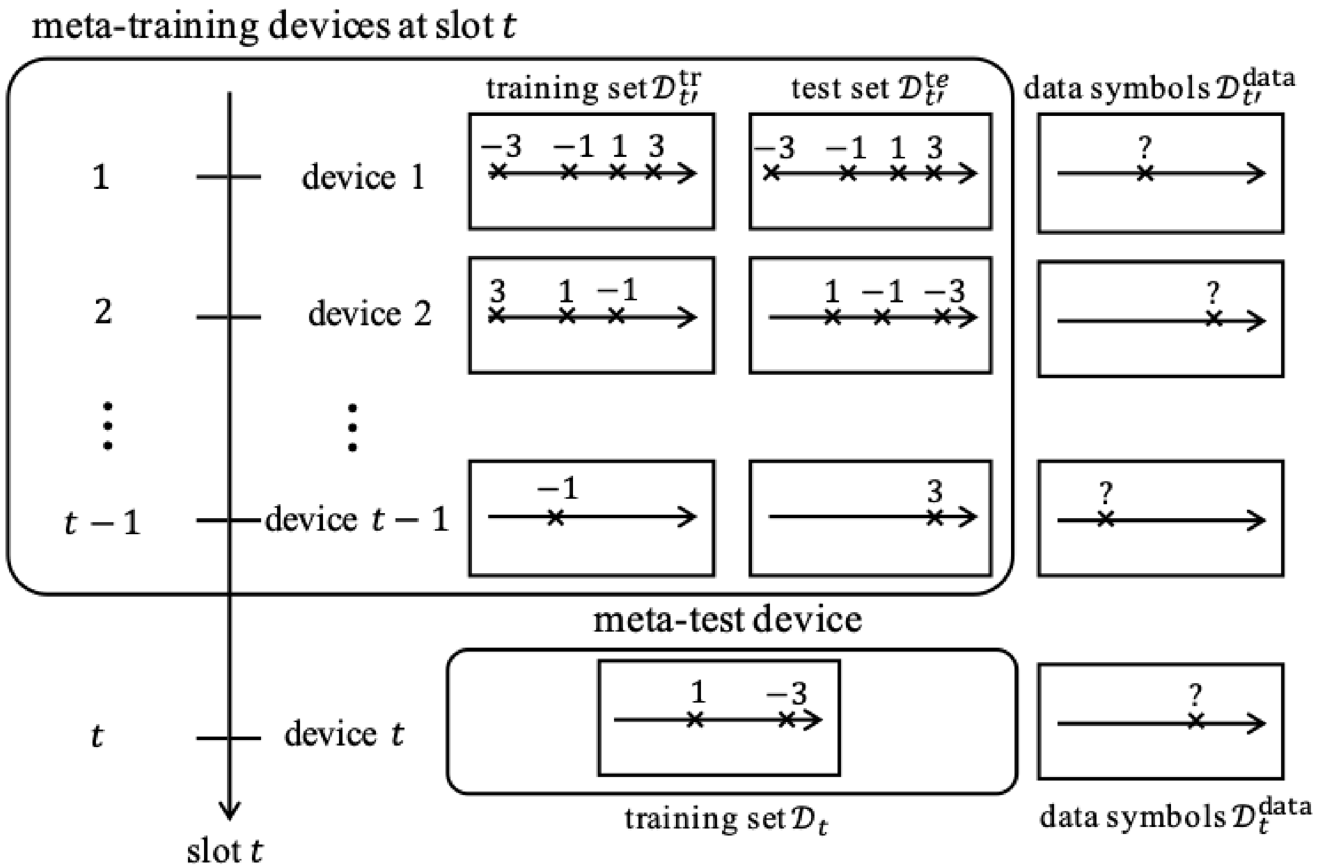
Meta-test dataset

$$\mathcal{D}_t = \{(s_t^{(n)}, y_t^{(n)}) : n = 1, \ldots, P_t\}$$

Online Meta-Learning

key online meta-learning parameters

t: current time slot
$t-1$: # of meta-training devices
P_t: # of pilots for current time slot
P: maximum # of pilots
$\mathcal{D}^{t-1}$: meta-training dataset
$\mathcal{D}_t$: meta-test dataset



Parameterized demodulator (neural network)
$$p(s_t|y_t,\varphi_t)$$

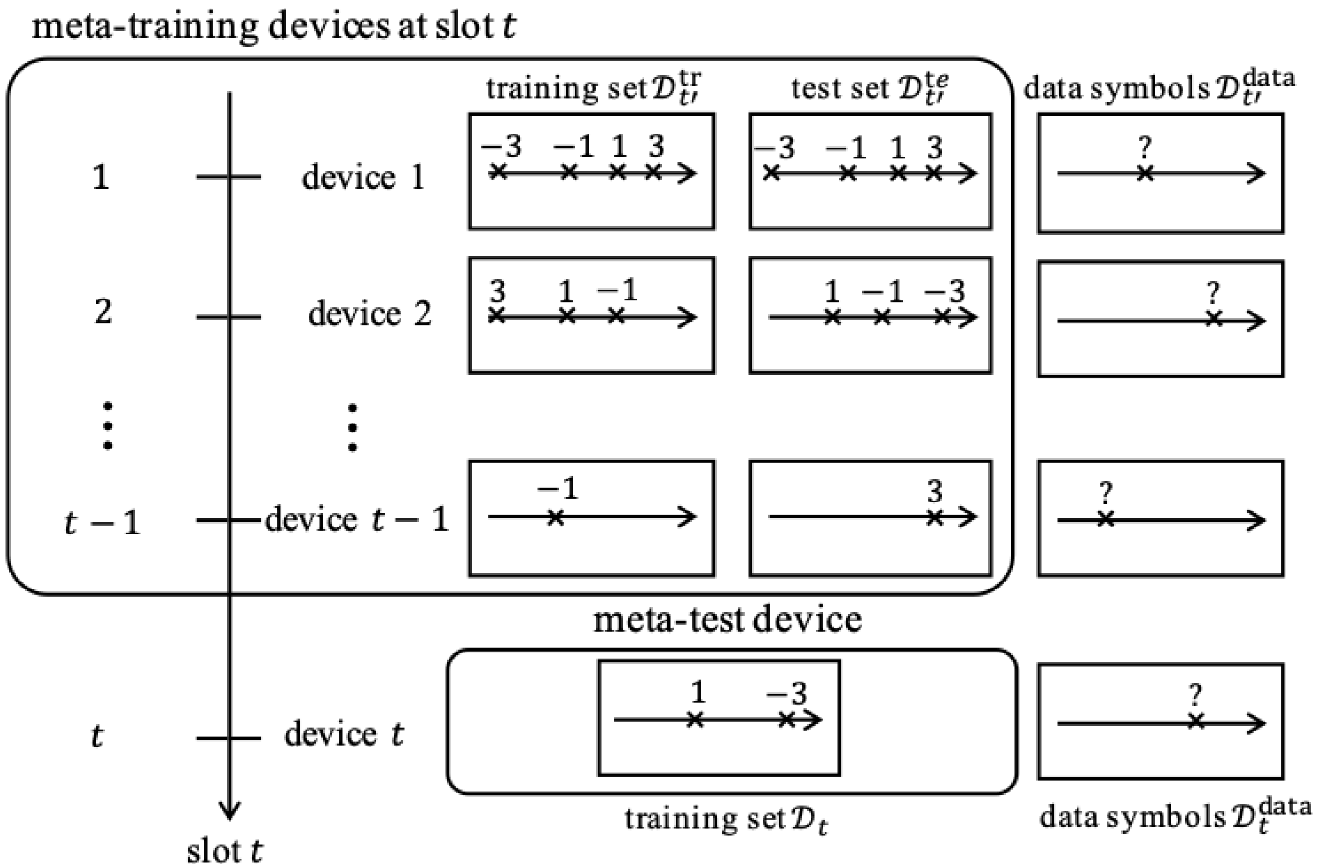
Meta-training dataset
$$\mathcal{D}^{t-1}=\{\mathcal{D}_{t'}\}_{t'=1}^{t-1}$$
$$\mathcal{D}_t=\{(s_t^{(n)},y_t^{(n)}):n=1,\dots,P_t\}$$

Meta-test dataset
$$\mathcal{D}_t=\{(s_t^{(n)},y_t^{(n)}):n=1,\dots,P_t\}$$

Online Meta-Learning

key online meta-learning parameters

t: current time slot
$t-1$: # of meta-training devices
P_t: # of pilots for current time slot
P: maximum # of pilots
$\mathcal{D}^{t-1}$: meta-training dataset
$\mathcal{D}_t$: meta-test dataset



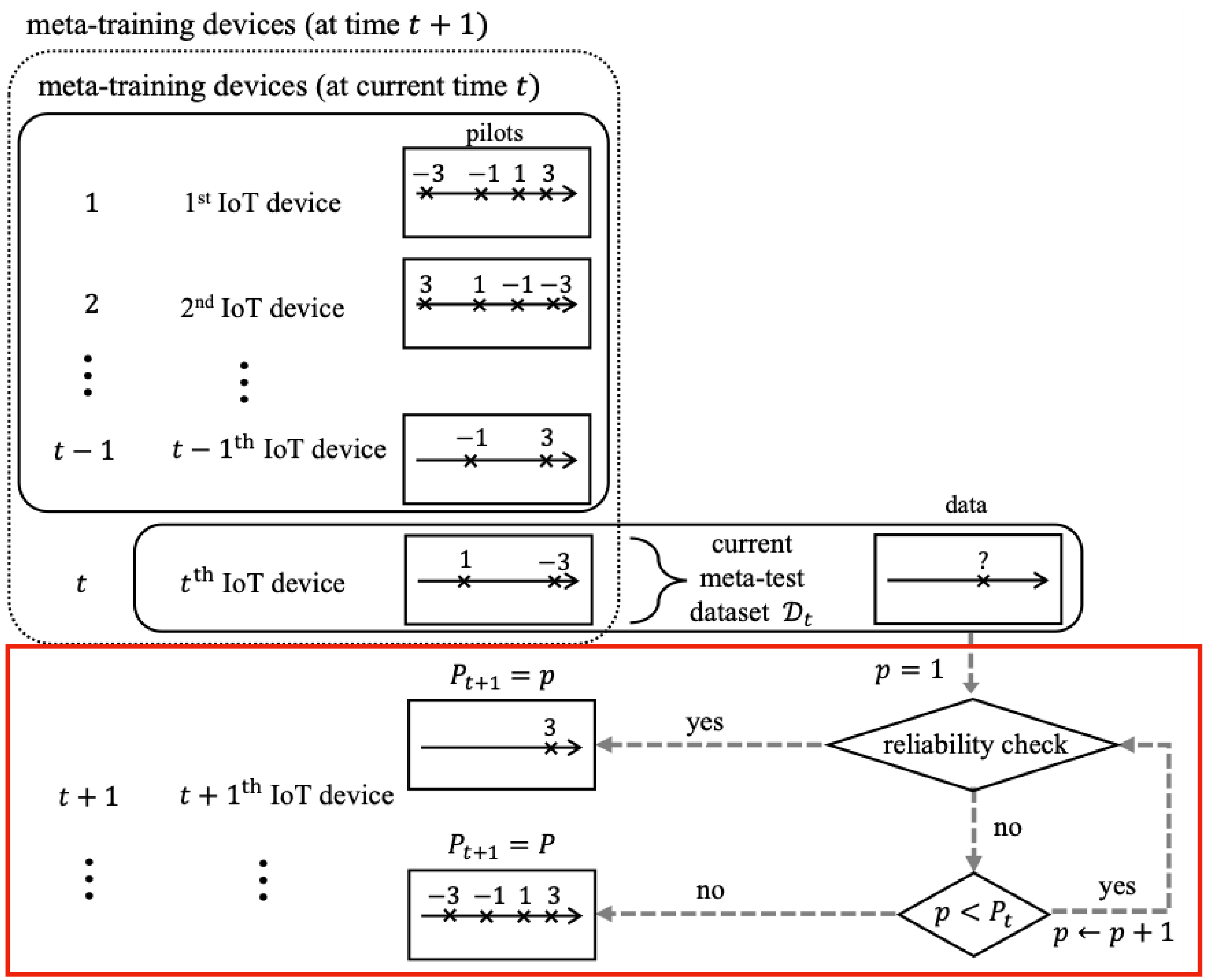
Parameterized demodulator (neural network)
$p(s_t|y_t, \varphi_t)$

Meta-training dataset
$$\mathcal{D}^{t-1} = \{\mathcal{D}_{t'}\}_{t'=1}^{t-1}$$
$$\mathcal{D}_t = \{(s_t^{(n)}, y_t^{(n)}) : n = 1, \ldots, P_t\}$$

Online meta-training is also known as lifelong learning

Meta-test dataset
$$\mathcal{D}_t = \{(s_t^{(n)}, y_t^{(n)}) : n = 1, \ldots, P_t\}$$

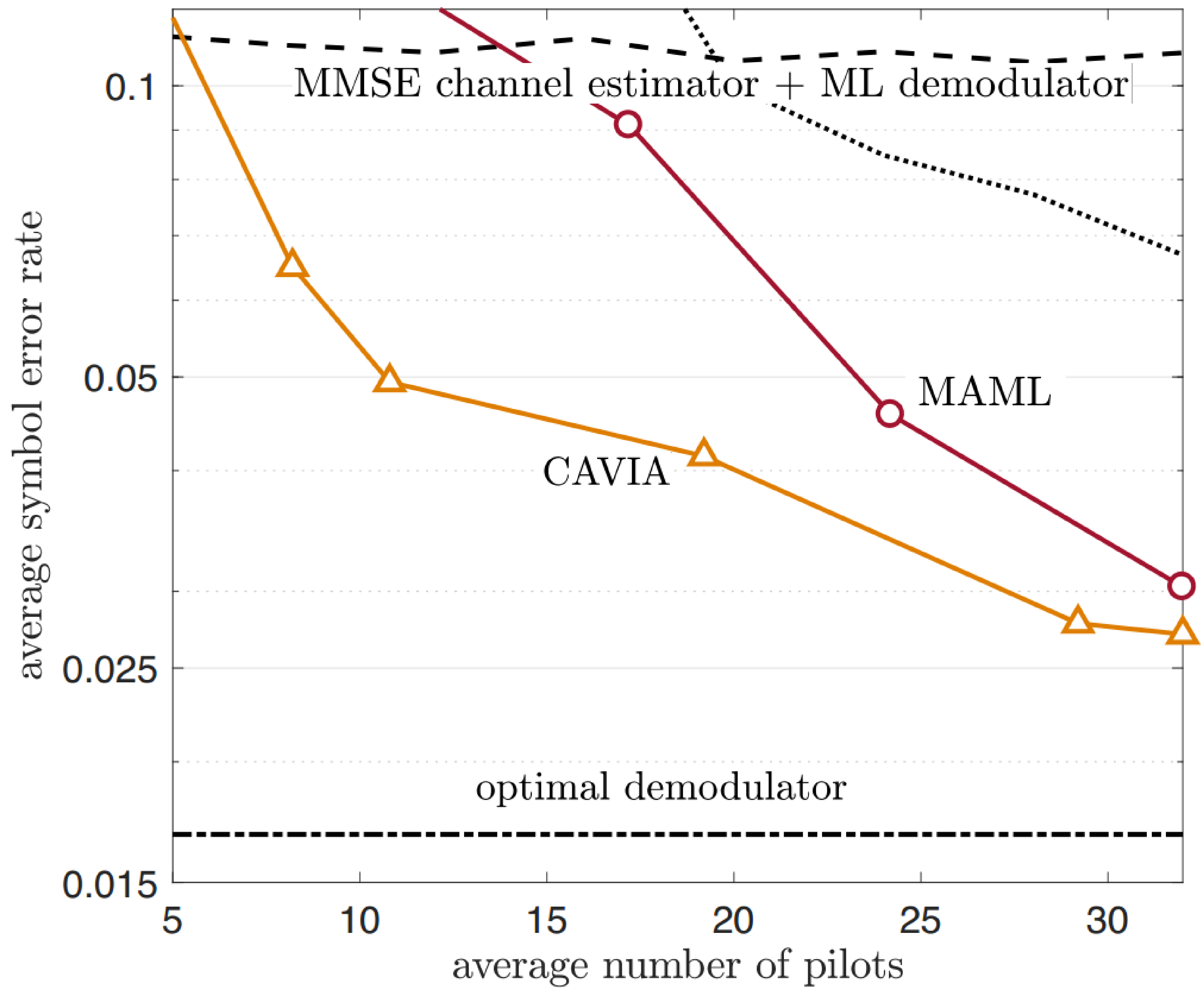
Adaptive Pilot Allocation



- Based on reliability check with different number of pilots in current time slot, determine number of pilots for the next time slot:

reliability check: $$-\sum_{y\in\mathcal{D}_t^{\text{data}}} \max_s[\log p(s|y,\phi_t^{(p)},\theta_t)]$$

Experiments



Concluding Remarks

- Meta-learning techniques can benefit communication systems with few pilots or when fast training is necessary.
- Reduction of sample or iteration complexity by transferring knowledge from related tasks.
- Online meta-learning may yield novel adaptive resource allocation.
- How many tasks should we observe? Information theoretic analysis [Jose and Simeone '20]

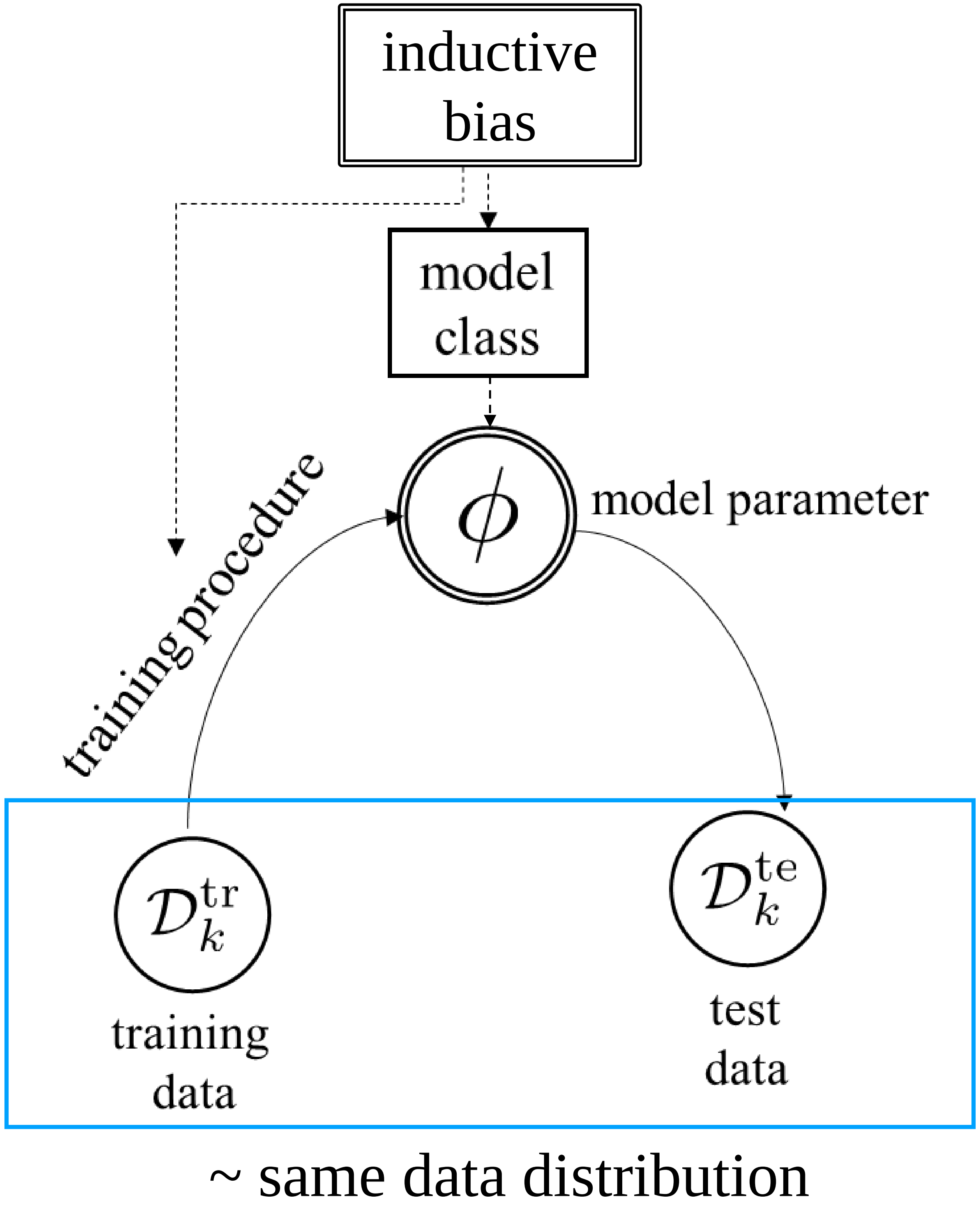
- Other potential applications of meta-learning:
  - Channel estimation and prediction
  - Precoding in multi-antenna systems
  - …

Extra Slides

- Some theory

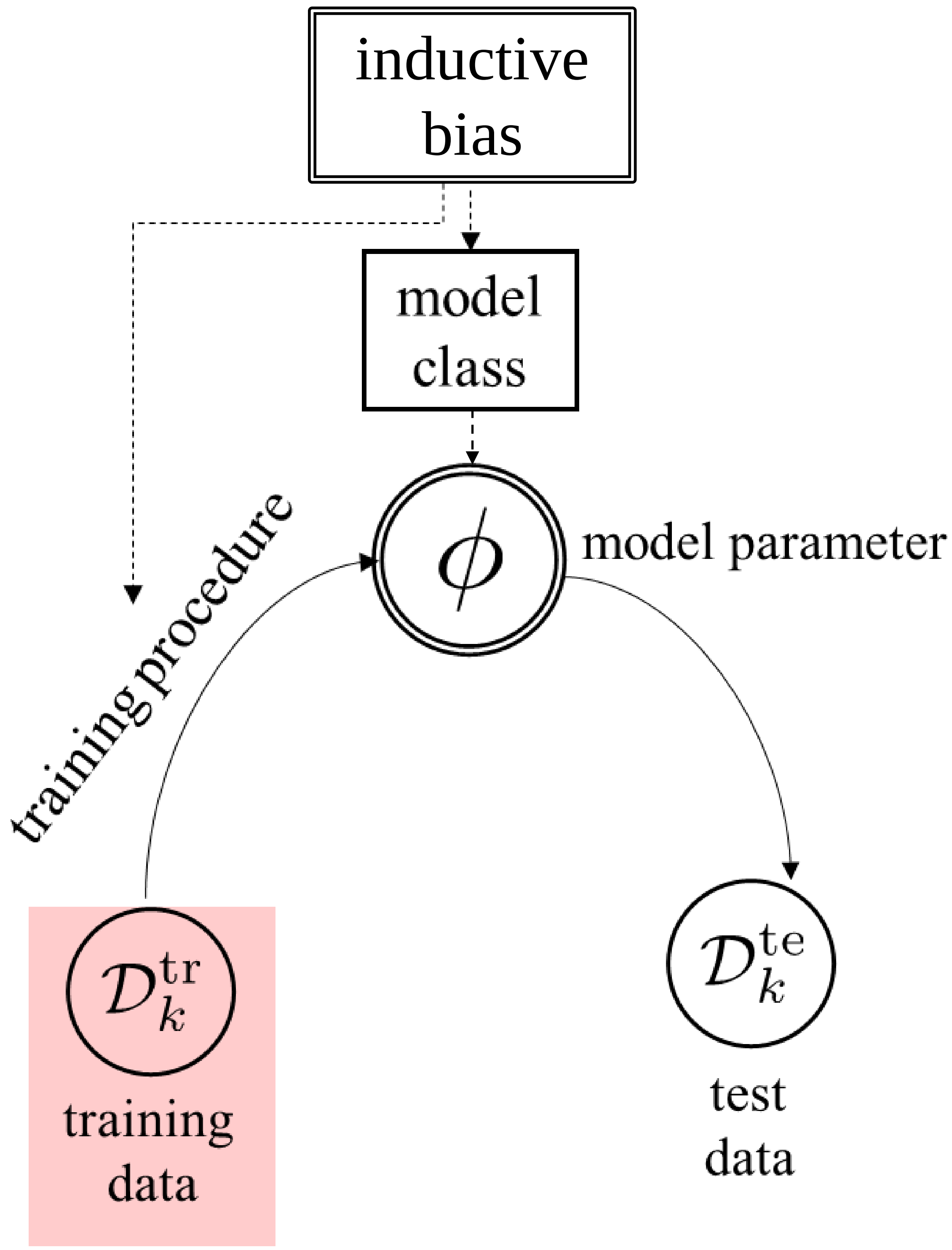
Sharu Theresa Jose and Osvaldo Simeone, “Information-Theoretic Generalization Bounds for Meta-Learning and Applications,” arXiv:2005.04372

Statistical Learning Theory



Statistical Learning Theory

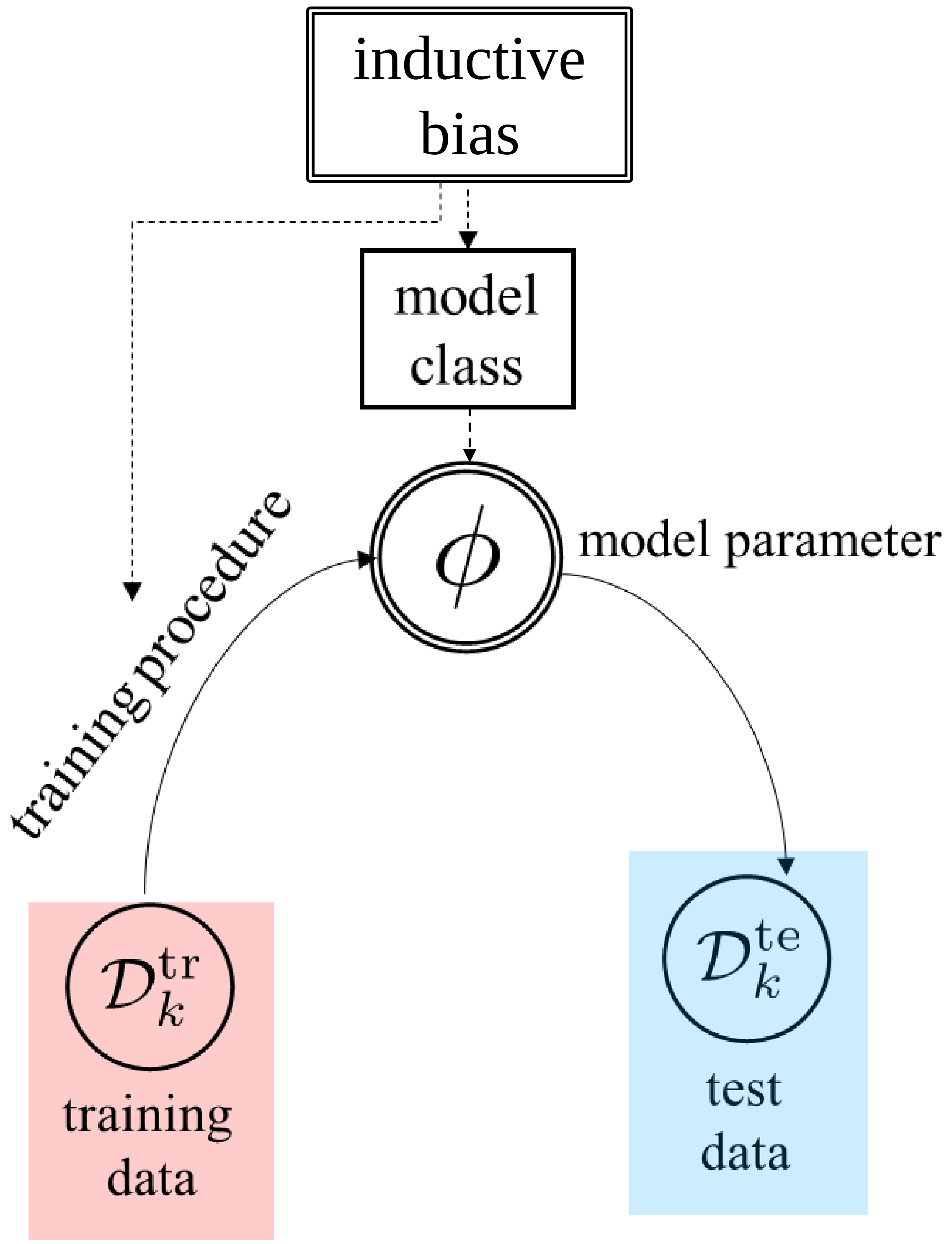
- For a given task k, the learner can compute the training loss $L_{D_k}(\phi_k)$.



Statistical Learning Theory

- For a given task k, the learner can compute the training loss $L_{D_k}(\phi_k)$.

- Test performance is measured is measured by the (unknown) test loss $L_k(\phi_k)$.

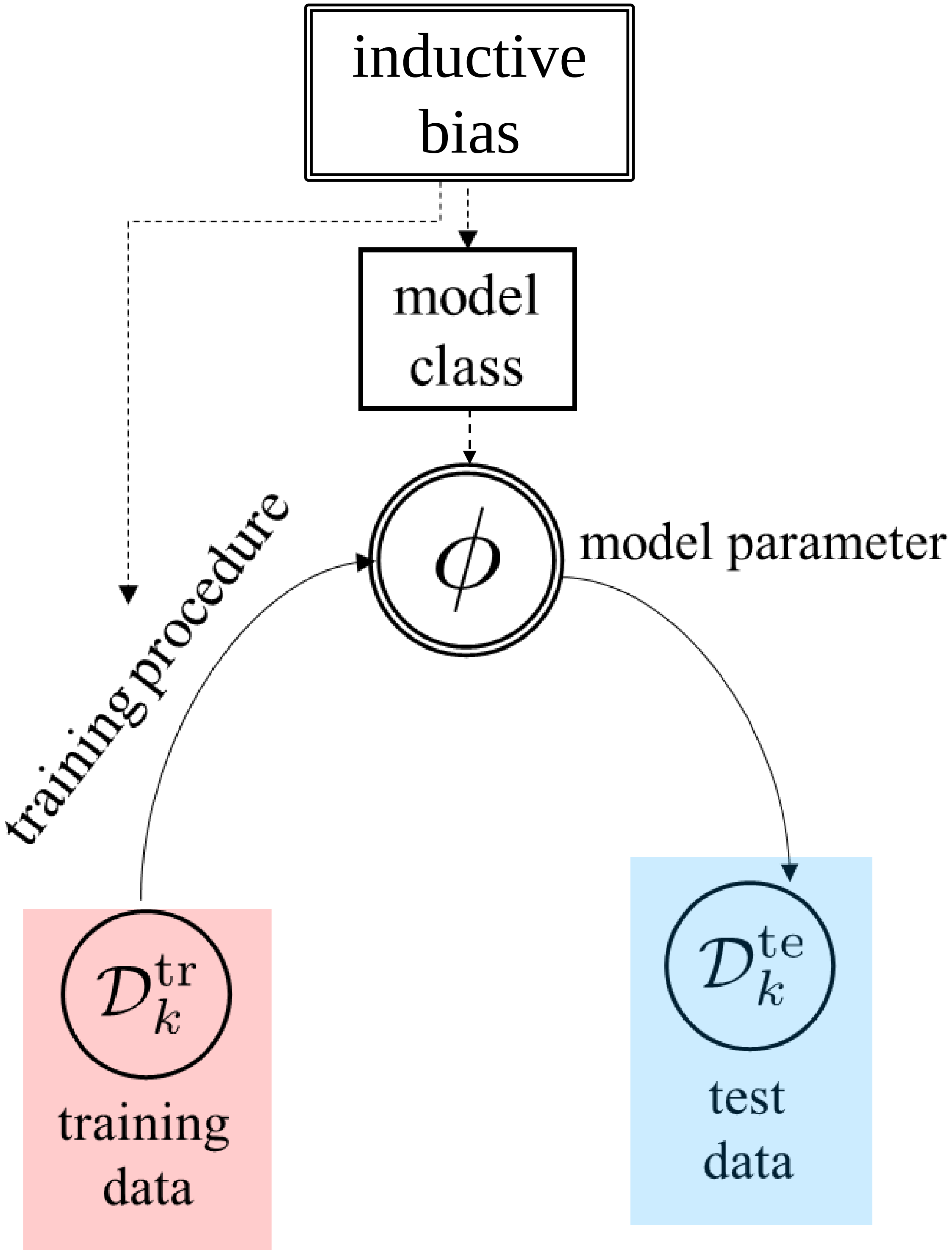


Statistical Learning Theory

- For a given task k, the learner can compute the training loss $L_{D_k}(\phi_k)$.

- Test performance is measured is measured by the (unknown) test loss $L_k(\phi_k)$.

- Test performance can be guaranteed if the generalization gap $L_k(\phi_k) - L_{D_k}(\phi_k)$ is small.



Statistical Learning Theory

- Under suitable assumptions [Xu-Raginsky '17],

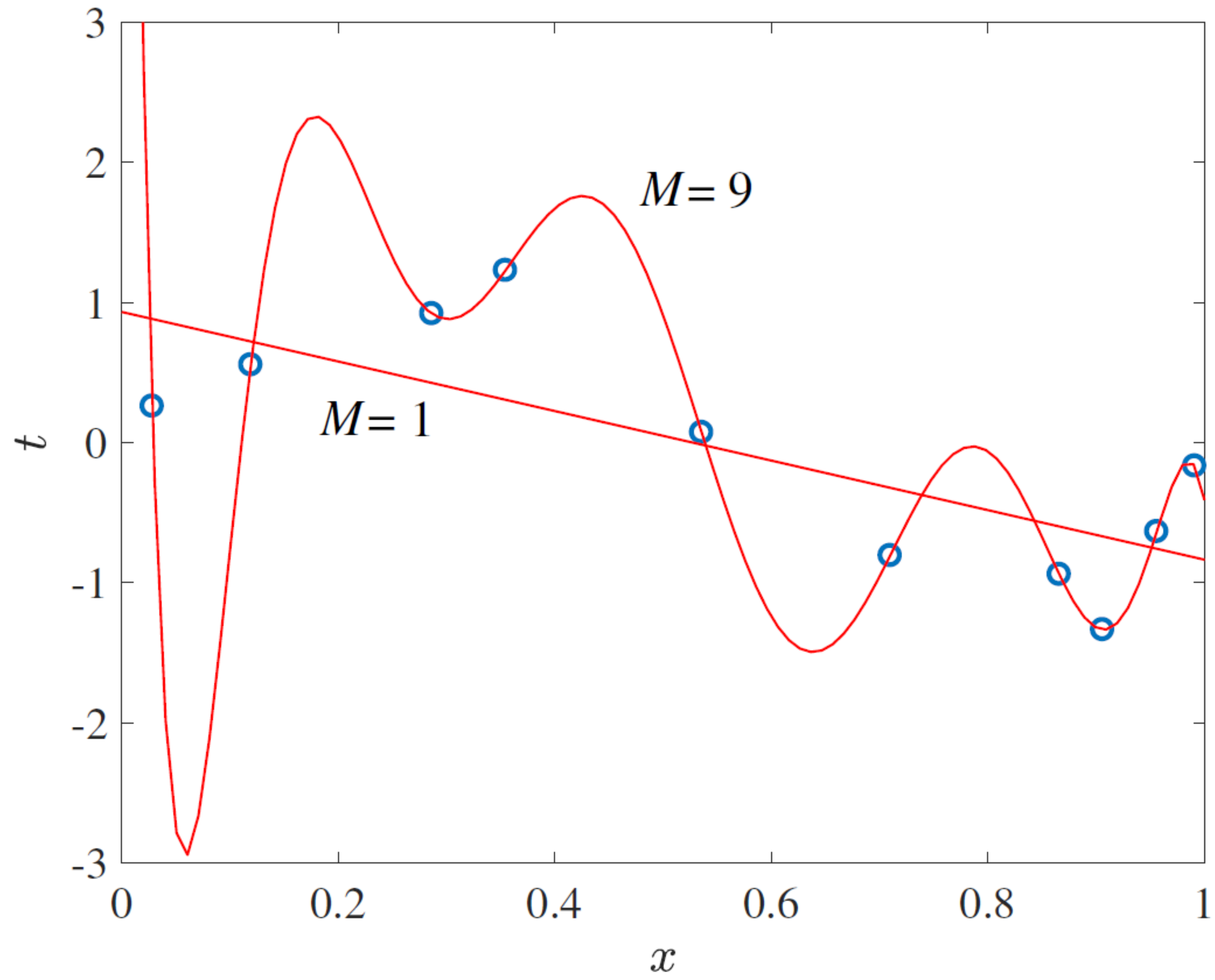
$$\mathrm{E}_{\text{train algo}}\left[L_k(\phi_k) - L_{\mathcal{D}_k}(\phi_k)\right] \leq \sqrt{\frac{2\sigma^2}{\#\text{train samples}} I(\phi_k; \mathcal{D}_k)}$$

Statistical Learning Theory

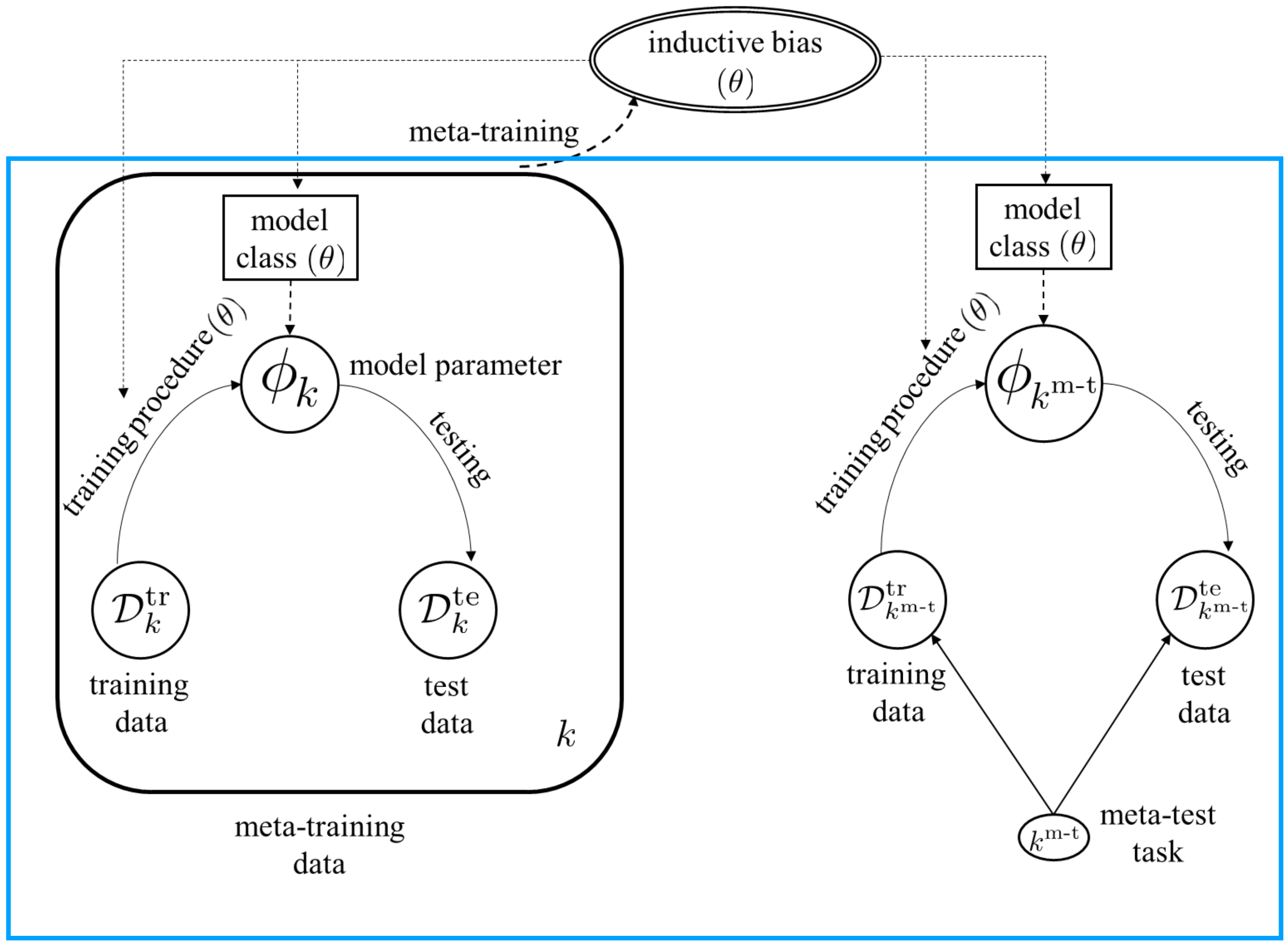
- Under suitable assumptions [Xu-Raginsky '17],

$$\mathrm{E}_{\text{train algo}}\left[L_k(\phi_k) - L_{\mathcal{D}_k}(\phi_k)\right] \leq \sqrt{\frac{2\sigma^2}{\#\text{train samples}} \boxed{I(\phi_k;\mathcal{D}_k)}}$$

"sensitivity" of training procedure to training data



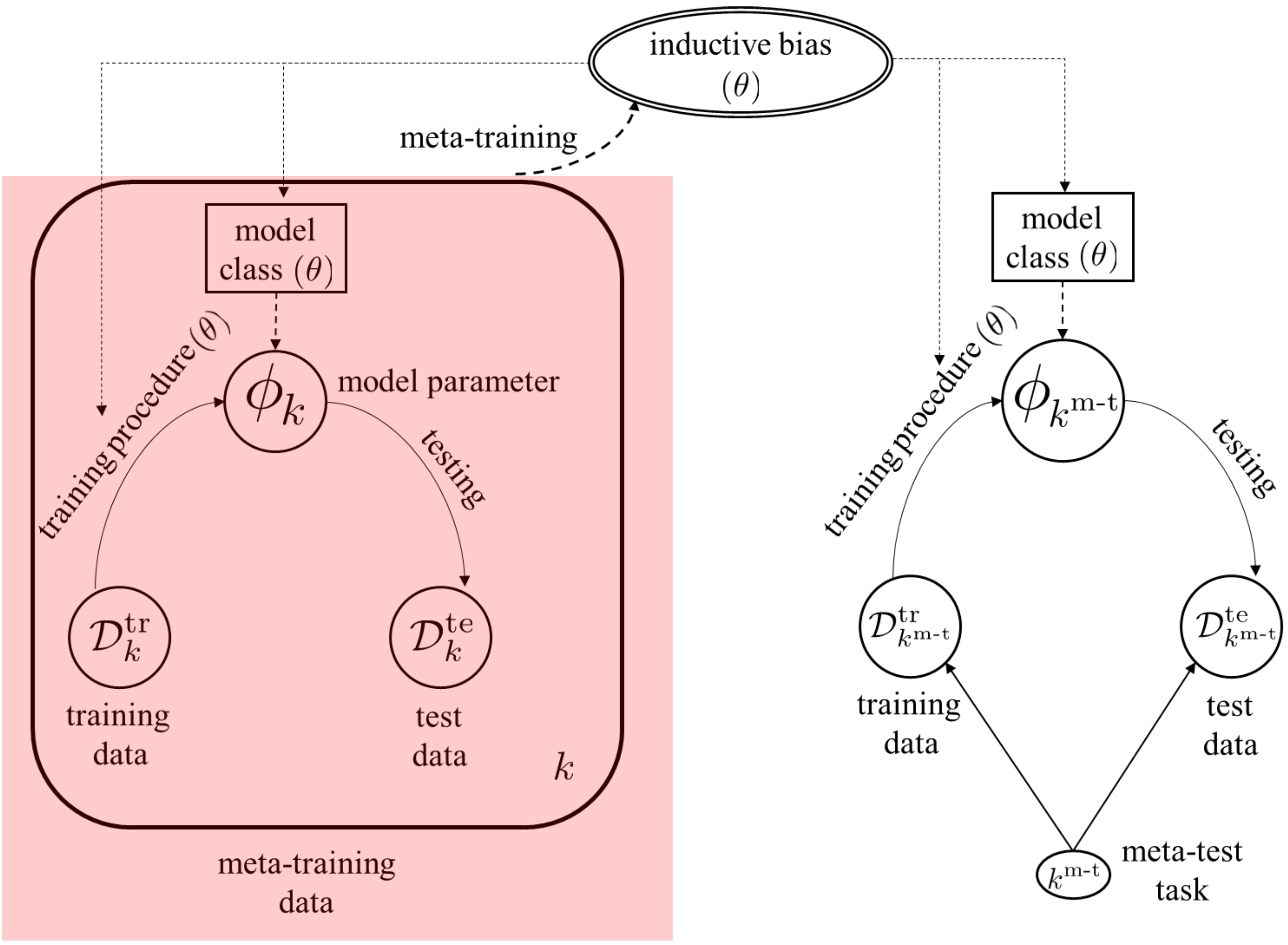
Statistical Meta-Learning Theory



~ same environment (task) distribution

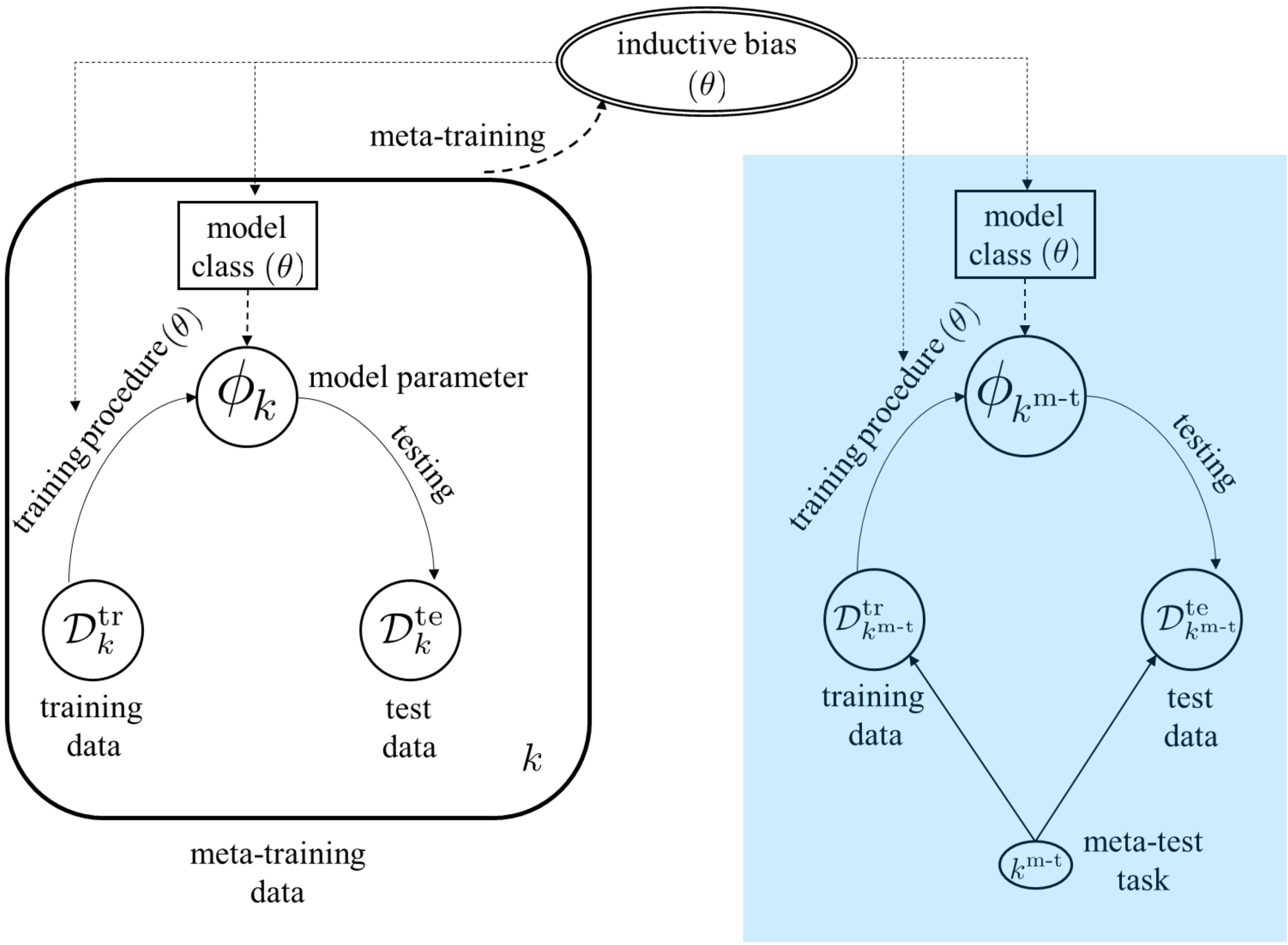
Statistical Meta-Learning Theory

- The meta-learner can compute the meta-training loss $L_D(\theta)$.



Statistical Meta-Learning Theory

- The meta-learner can compute the meta-training loss $L_{\mathcal{D}}(\theta)$.
- Test performance is measured is measured by the (unknown) meta-test loss $L(\theta)$.

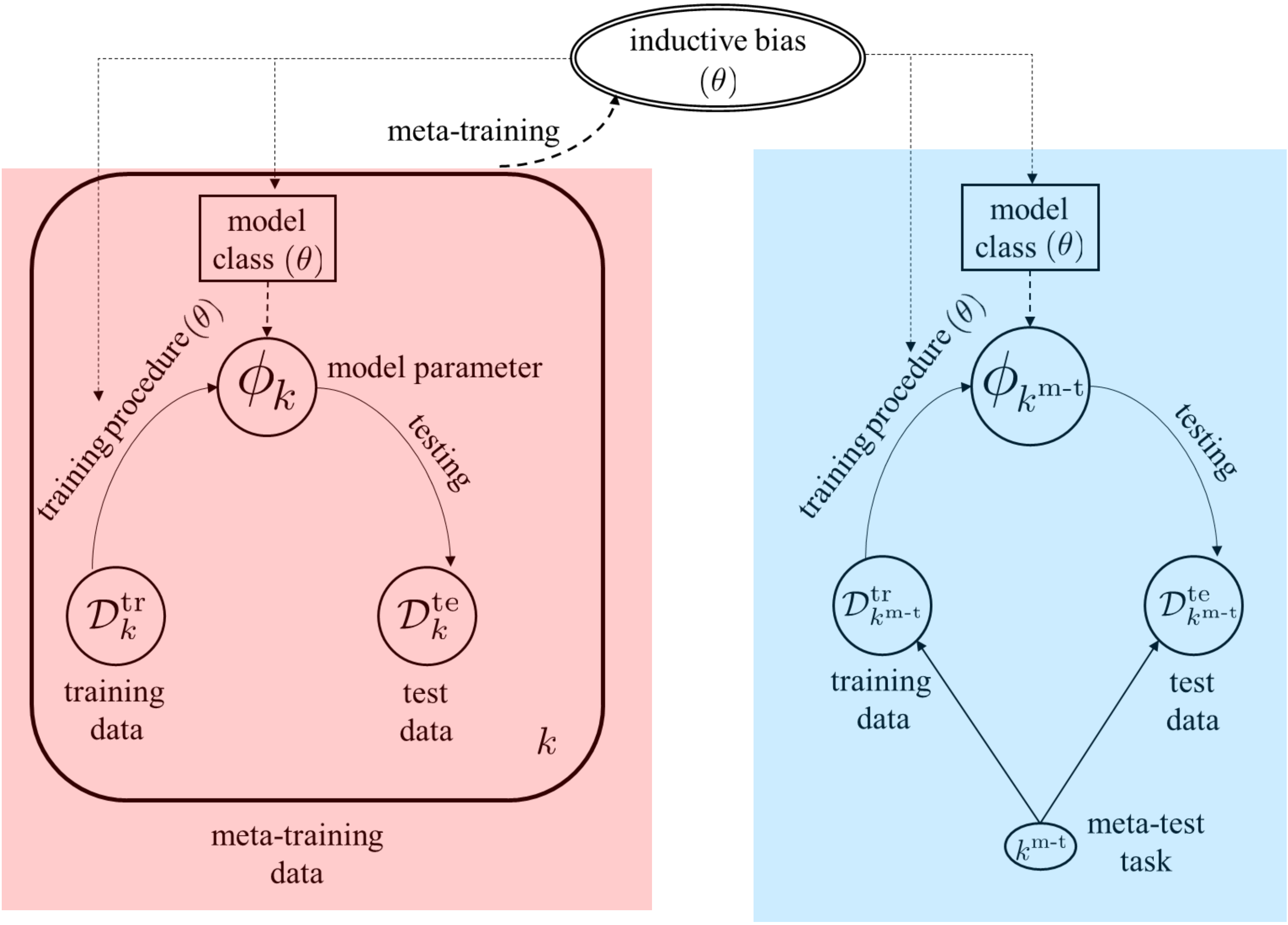


Statistical Meta-Learning Theory

- The meta-learner can compute the meta-training loss $L_D(\theta)$.

- Test performance is measured is measured by the (unknown) meta-test loss $L(\theta)$.

- Test performance can be guaranteed if the meta-generalization gap $L(\theta) - L_D(\theta)$ is small.



Statistical Meta-Learning Theory

- Under suitable assumptions [Jose and Simeone '20],

$$\mathrm{E}_{\text{meta-train algo}}\left[L(\theta) - L_{\mathcal{D}}(\theta)\right] \leq \sqrt{\frac{2\sigma^2}{\#\text{meta-train tasks}} \boxed{I(\theta;\mathcal{D})}}$$

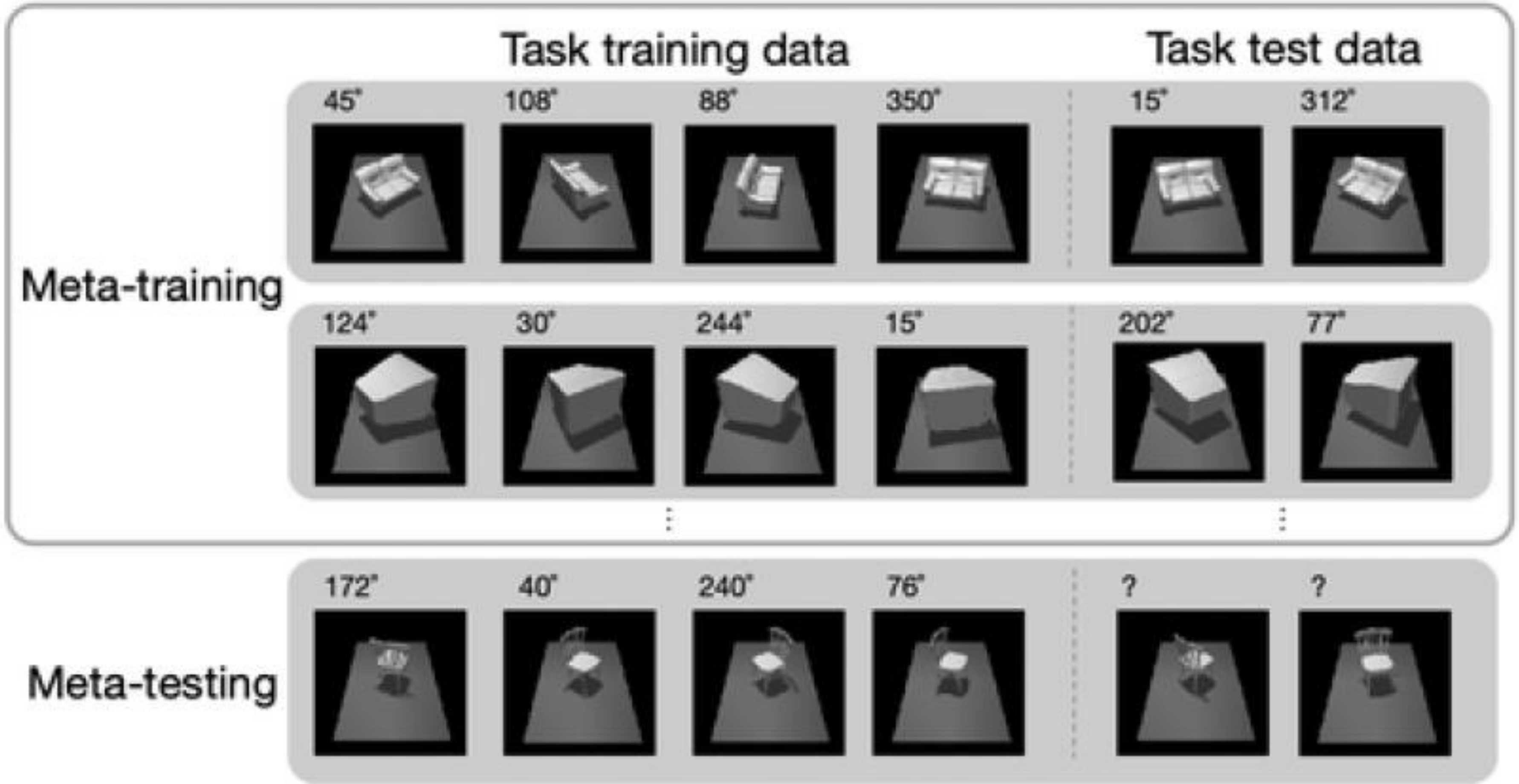
"sensitivity" of meta-training procedure to meta-training data

Statistical Meta-Learning Theory

- Under suitable assumptions [Jose and Simeone '20],

$$\mathrm{E}_{\text{meta-train algo}}\left[L(\theta) - L_{\mathcal{D}}(\theta)\right] \leq \sqrt{\frac{2\sigma^2}{\#\text{meta-train tasks}} \boxed{I(\theta;\mathcal{D})}}$$

"sensitivity" of meta-training procedure to meta-training data



[Yin et al '19]