Transfer Learning

Multi-Task Learning

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

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

Transfer Learning

Solve target task $\mathcal{T}_b$ after solving source task(s) $\mathcal{T}_a$

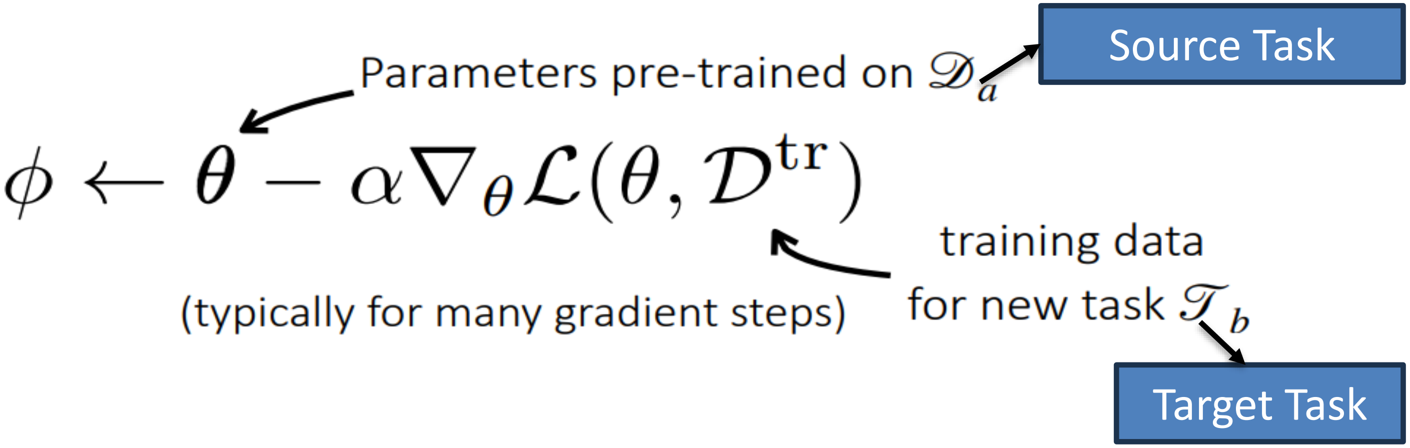
by *transferring* knowledge learned from $\mathcal{T}_a$

Common assumption: Cannot access data $\mathcal{D}_a$ during transfer.

Application Areas of Transfer Learning

- Resource constraint environment
- Source task and target task need not to be solved simultaneously.
- Privacy concerns associated with the source training data-set.

Transfer Learning via Fine Tuning



Parameters pre-trained on $\mathscr{D}_a$

Source Task

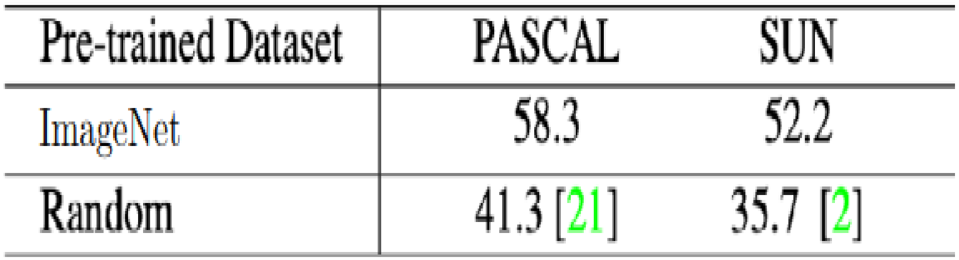
$$\phi \leftarrow \theta - \alpha \nabla_{\theta} \mathcal{L}(\theta, \mathcal{D}^{\mathrm{tr}})$$

training data for new task $\mathscr{T}_b$

(typically for many gradient steps)

Target Task

Transfer Learning via Fine Tuning

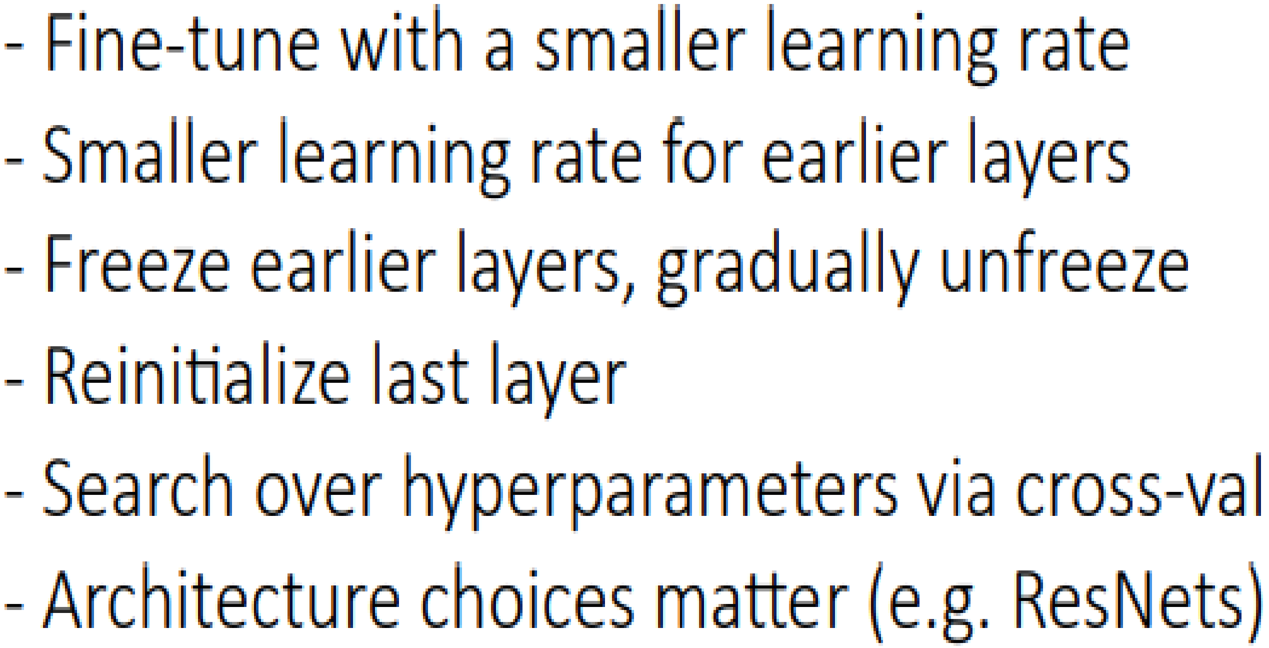


Pre-trained Dataset	PASCAL	SUN
ImageNet	58.3	52.2
Random	41.3 [21]	35.7 [2]

What makes ImageNet good for transfer learning? Huh, Agrawal, Efros. '16

Feature embeddings obtained by training on coarse classes be able to distinguish fine classes they were never trained on

Fine tuning Design Choices



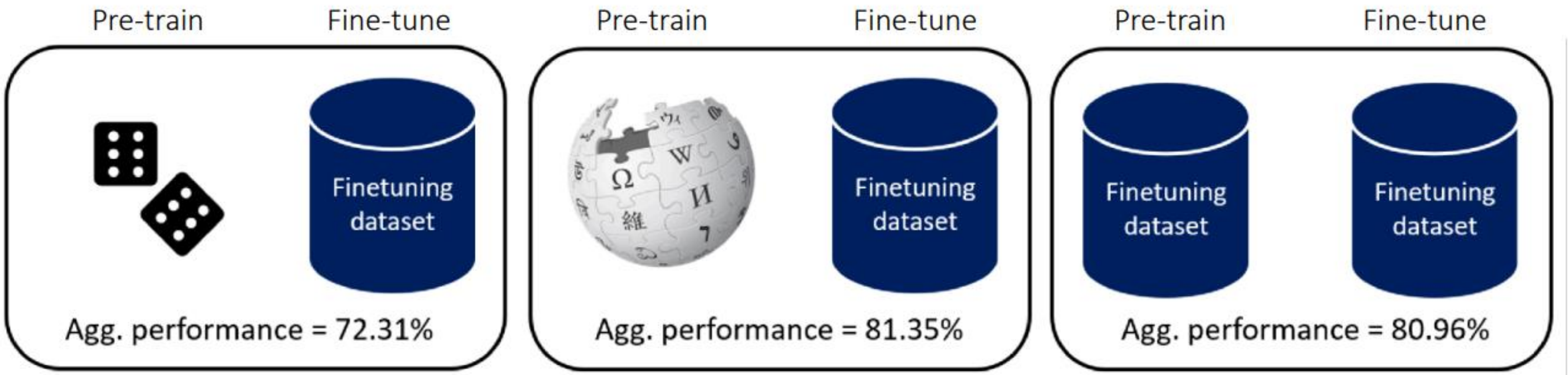
- Fine-tune with a smaller learning rate
- Smaller learning rate for earlier layers
- Freeze earlier layers, gradually unfreeze
- Reinitialize last layer
- Search over hyperparameters via cross-val
- Architecture choices matter (e.g. ResNets)

Transfer Learning

- Large-scale pretrained models typically provide significant performance boosts when compared to models trained directly on the downstream task (with random initializations) (Peters et al., 2018; Devlin et al., 2019; Chiang and Lee, 2020; Krishna et al., 2021).
- Upstream corpora tend to be significantly larger than the downstream corpora and the success of this approach is often attributed to its ability to leverage these massive upstream corpora (Liu et al., 2019; Yang et al., 2019).

Transfer Learning

Unsupervised pre-training objectives may not require diverse data for pre-training.



Krishna, Garg, Bingham, Lipton. Downstream Datasets Make Surprisingly Good Pretraining Corpora. ACL 2023.

Figure 1: Aggregate performance of an ELECTRA model across 10 finetuning datasets when it is (i) randomly initialized (ii) pretrained on upstream corpus (BookWiki) (iii) pretrained on the finetuning dataset itself

Transfer Learning

- Pretraining models only on text from the downstream dataset performs comparably to pretraining on a huge upstream corpus for a wide variety of datasets.
- The errors made by such self-pretrained models on the downstream tasks are significantly different from the ones made by the off-the-shelf models pretrained on upstream corpora.
- Downstream datasets which are tiny in comparison to typical upstream corpora, still function as useful pretraining corpora for getting performance gains across a wide range of datasets.

Transfer Learning

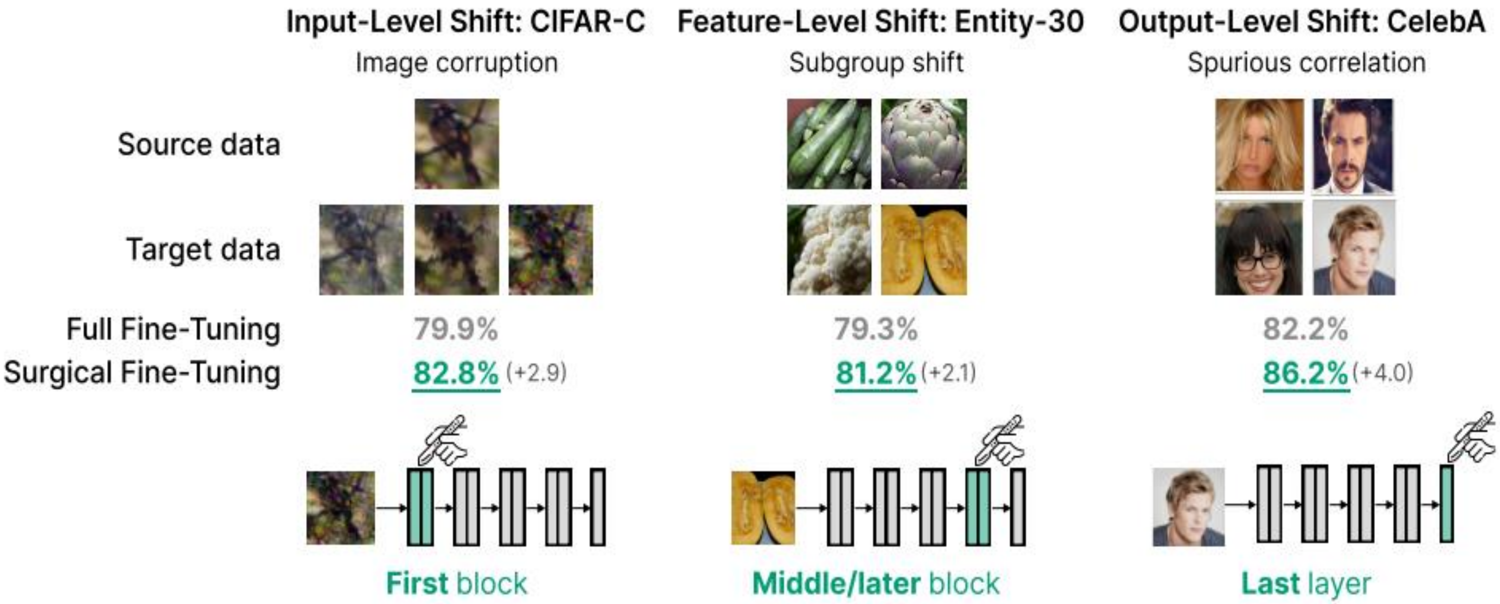


Figure 1: Surgical fine-tuning, where we tune only one block of parameters and freeze the remaining parameters, outperforms full fine-tuning on a range of distribution shifts. Moreover, we find that tuning different blocks performs best for different types of distribution shifts. Fine-tuning the first block works best for input-level shifts such as CIFAR-C (image corruption), later blocks work best for feature-level shifts such as Entity-30 (shift in entity subgroup), and tuning the last layer works best for output-level shifts such as CelebA (spurious correlation between gender and hair color).

Lee*, Chen*, Tajwar, Kumar, Yao, Liang, Finn. Surgical Fine-Tuning Improves Adaptation to Distribution Shifts. ICLR 2023.

Transfer Learning to Meta Learning

Transfer learning: Initialize model. Hope that it helps the target task.

Meta-learning: Can we explicitly ***optimize*** for transferability?

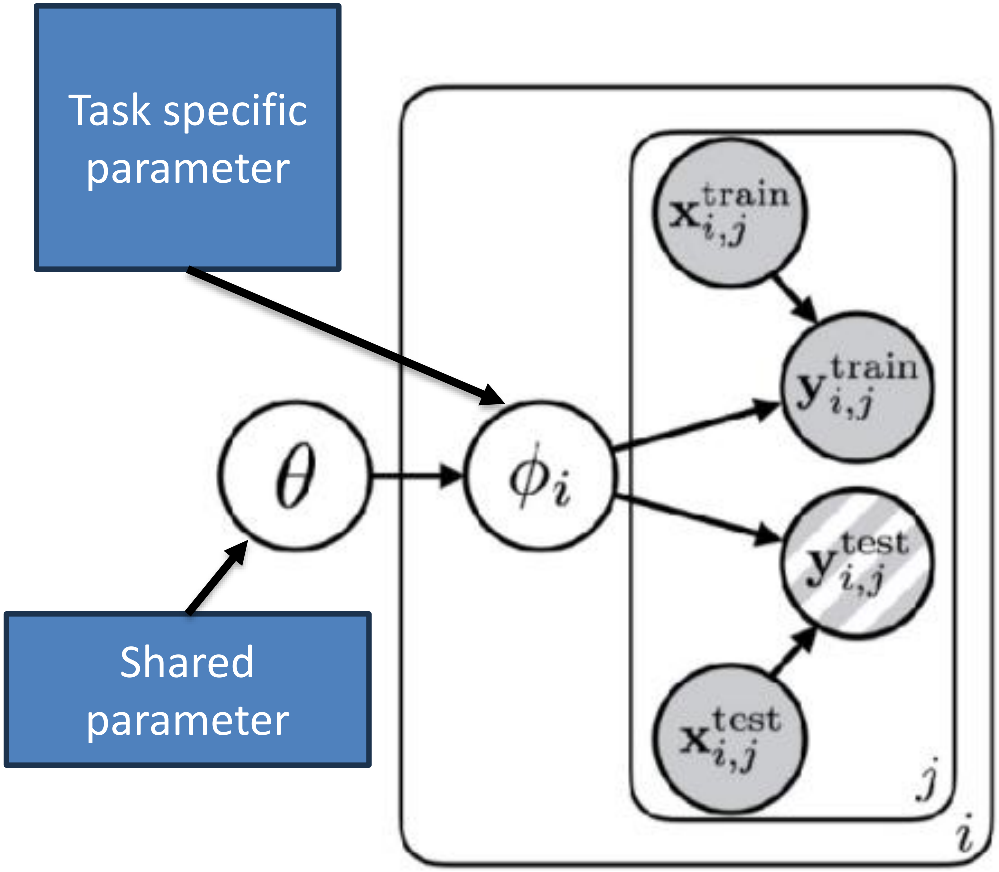
Given a set of training tasks, can we optimize for the ability to learn these tasks quickly?
so that we can learn ***new*** tasks quickly too

Learning a task: $\mathscr{D}_i^{tr} \longrightarrow \theta$

Can we optimize this function?

(for small $\mathscr{D}_i^{tr}$)

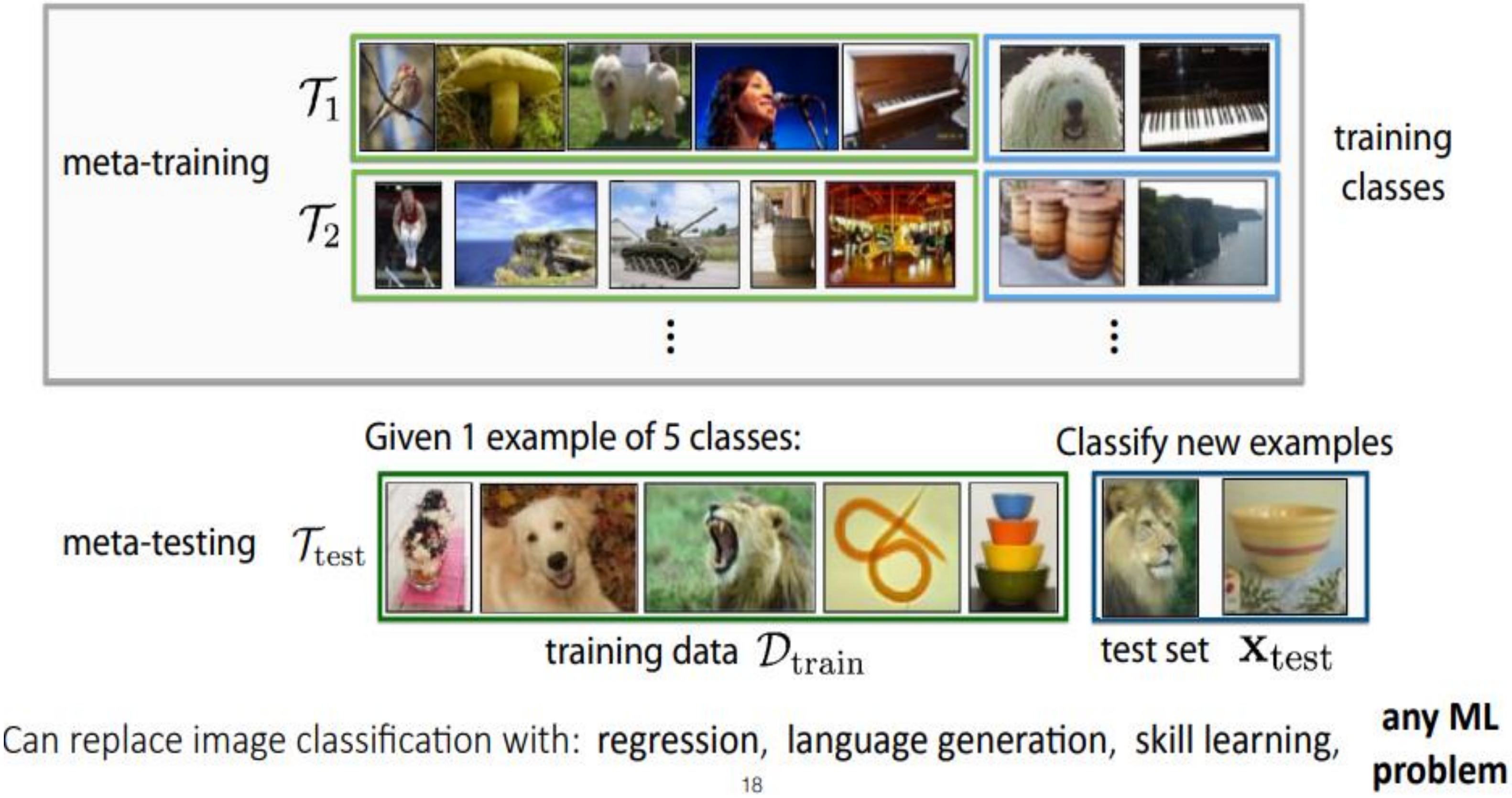
Bayes view of Meta Learning


θ
ϕ_i
$\mathbf{x}_{i,j}^{\text{train}}$
$\mathbf{y}_{i,j}^{\text{train}}$
$\mathbf{y}_{i,j}^{\text{test}}$
$\mathbf{x}_{i,j}^{\text{test}}$
j
i

Task specific parameter

Shared parameter

Meta Learning



meta-training $\mathcal{T}_1$ $\mathcal{T}_2$

training classes

Given 1 example of 5 classes:

Classify new examples

meta-testing $\mathcal{T}_{\text{test}}$

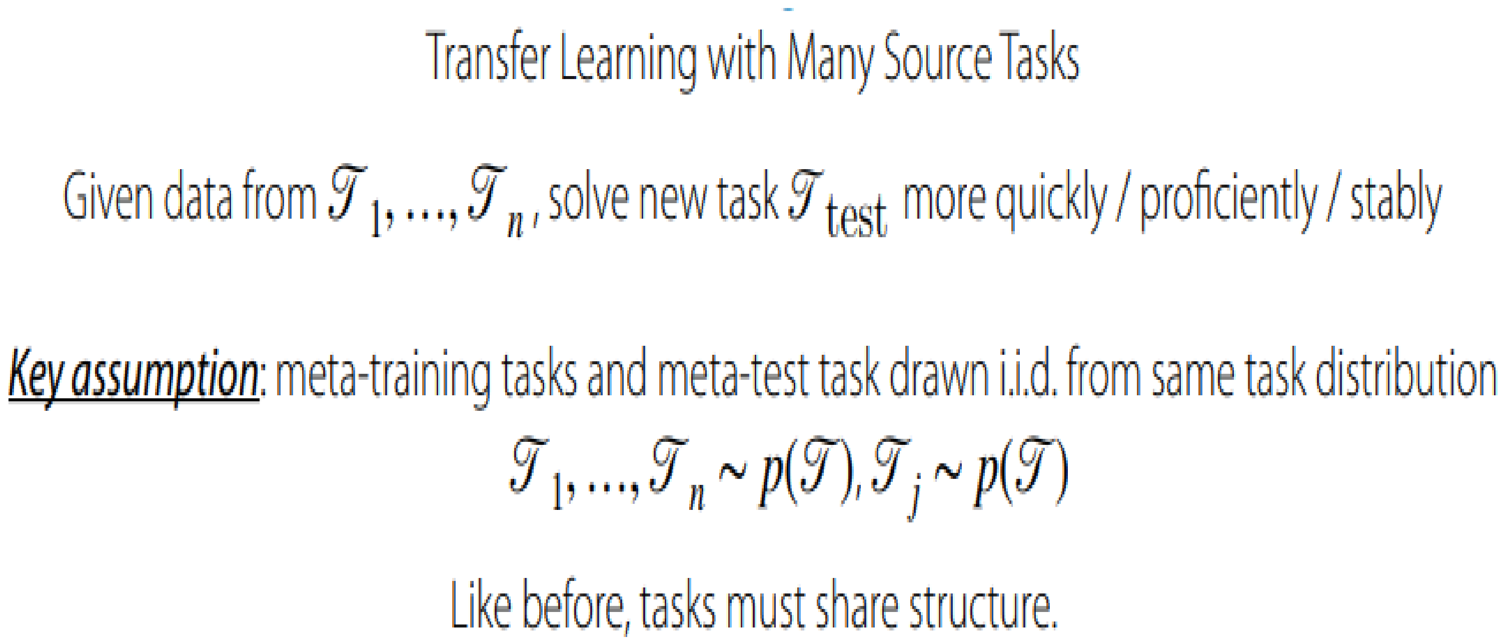
training data $\mathcal{D}_{\text{train}}$

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

Can replace image classification with: **regression, language generation, skill learning, any ML problem**

18

Meta Learning



Transfer Learning with Many Source Tasks

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

Key assumption: meta-training tasks and meta-test task drawn i.i.d. from same task distribution

$$\mathcal{T}_1, \ldots, \mathcal{T}_n \sim p(\mathcal{T}), \mathcal{T}_j \sim p(\mathcal{T})$$

Like before, tasks must share structure.

Multi-Task Learning

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

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

Transfer Learning

Solve target task $\mathcal{T}_b$ after solving source task(s) $\mathcal{T}_a$

by *transferring* knowledge learned from $\mathcal{T}_a$

Meta-Learning Problem

Transfer Learning with Many Source Tasks

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

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

In all settings: tasks must share structure.