Robot Learning

Meta & Multi-task Learning

USC University of Southern California

LiraLab

Today…

- Multi-task Learning

- Transfer Learning

- Meta Learning

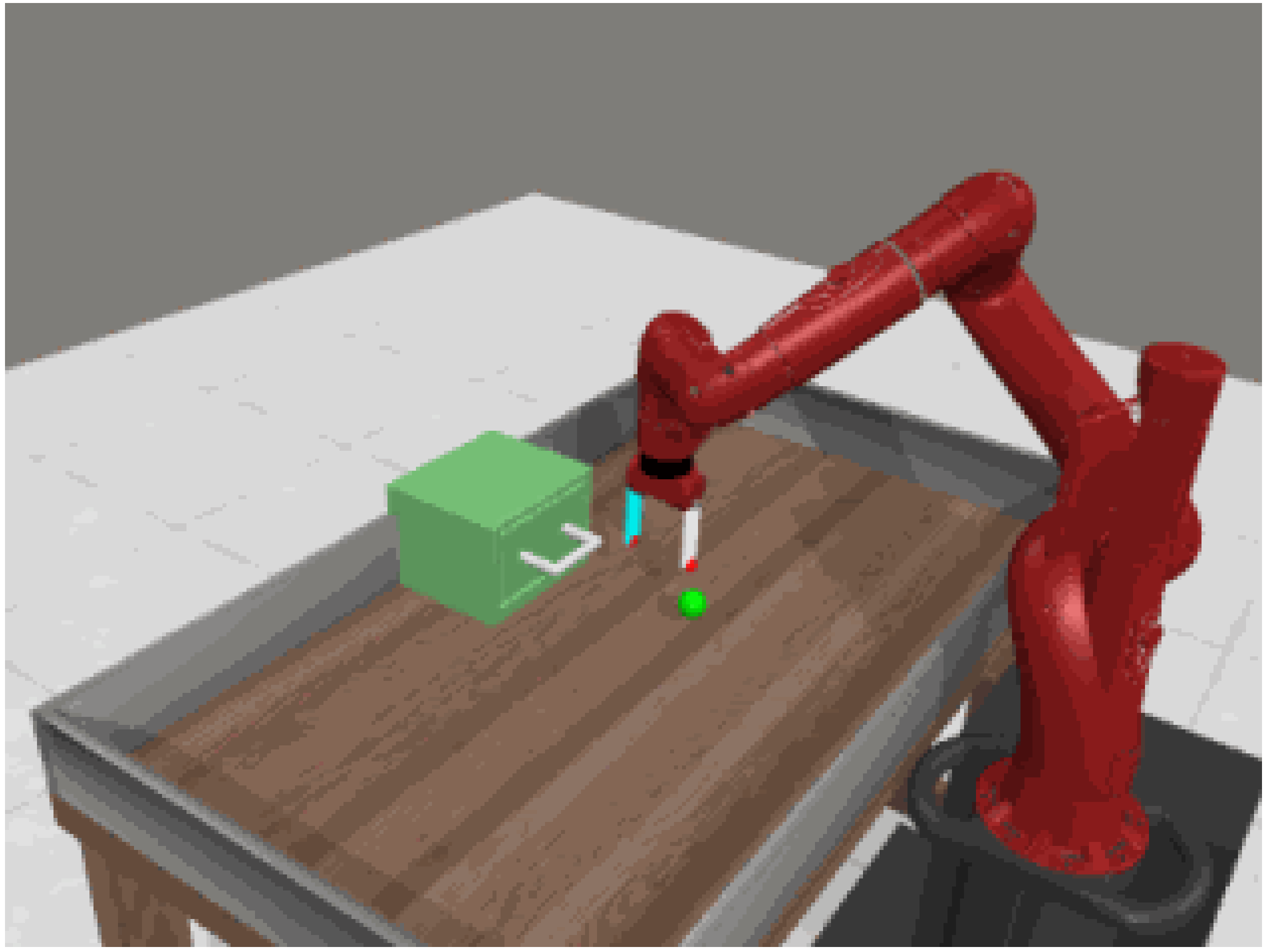
What is a task?

$$\underset{\pi}{\text{maximize}} \quad \mathbb{E}_{\boldsymbol{w}}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$$

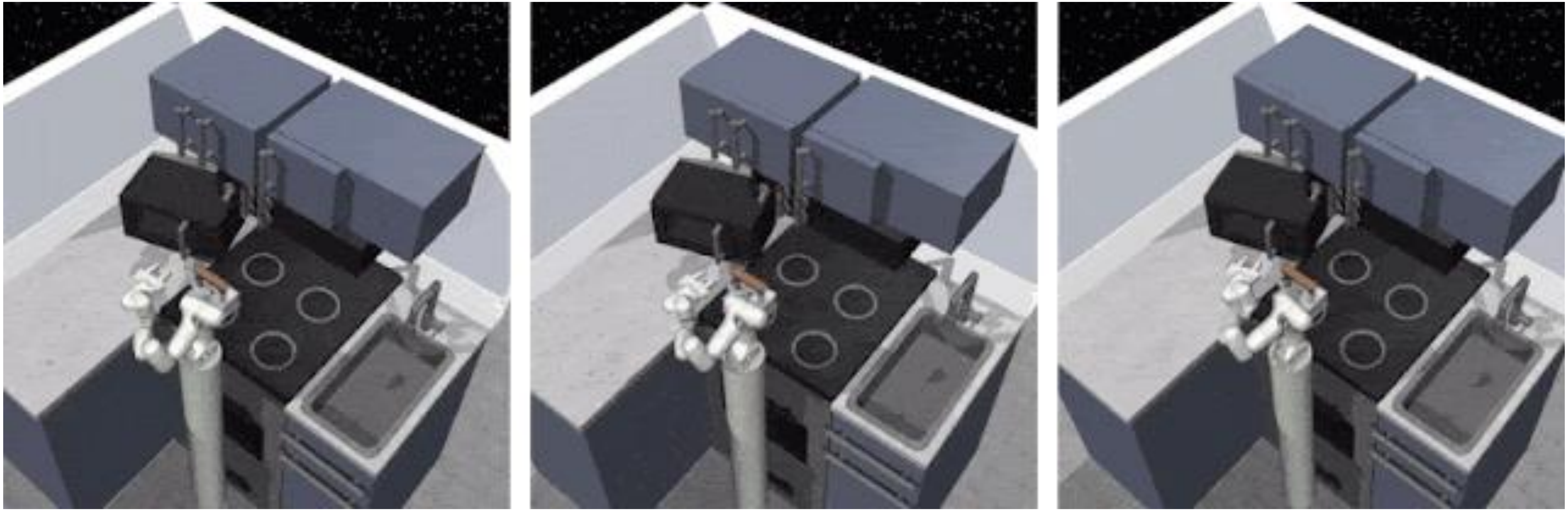
$$\text{subject to } s_{t+1} = f(s_t, a_t, w_t)$$
$$a_t = \pi(s_t)$$

More generally in machine learning, a dataset-loss function pair defines a task.

What's wrong with single-task learning?



What's wrong with single-task learning?



Still... Why multi-task learning?

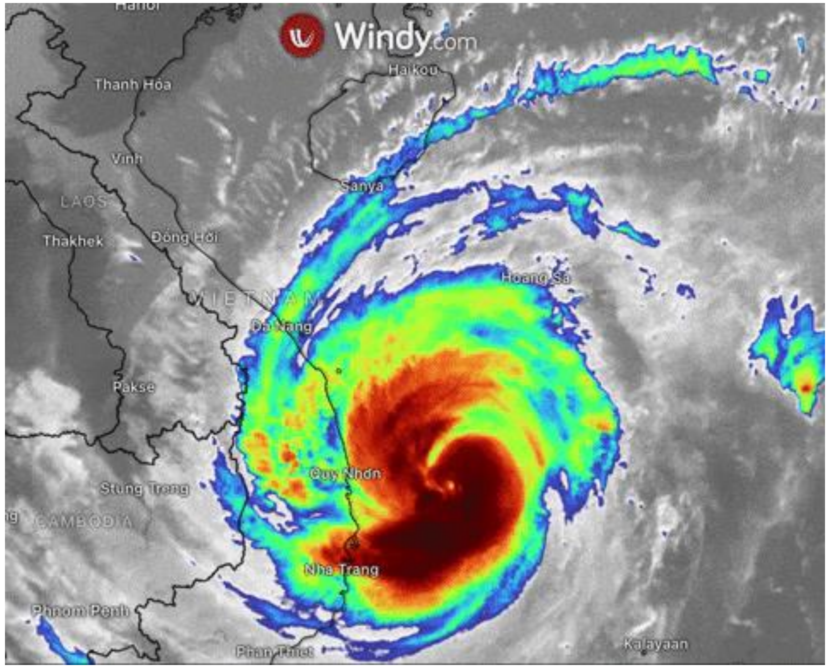
We could just learn each task independently!

- What if we have little data for some tasks?

- What if we have little time to learn some tasks?

Still... Why multi-task learning?

You should learn each task independently if there is no shared structure between the tasks.



Left: Playing atari with deep reinforcement learning
Mnih et al., NeurIPS Deep Learning Workshop 2013
Right: Windy.com Community

Still... Why multi-task learning?

In real life, many tasks share structure!

From: Stanford CS330

Today…

- Multi-task Learning

- Transfer Learning

- Meta Learning

Today…

- Multi-task Learning: Learn multiple tasks together
- Transfer Learning
- Meta Learning

Today…

- Multi-task Learning: Learn multiple tasks together
- Transfer Learning: Learn multiple tasks and transfer your knowledge to a new one
- Meta Learning

Today…

- Multi-task Learning: Learn multiple tasks together
- Transfer Learning: Learn multiple tasks and transfer your knowledge to a new one
- Meta Learning: Learn multiple tasks such that adapting to a new task will be easy

Today…

- Multi-task Learning: Learn multiple tasks together
- Transfer Learning: Learn multiple tasks and transfer your knowledge to a new one
- Meta Learning: Learn multiple tasks such that adapting to a new task will be easy

Multi-task learning

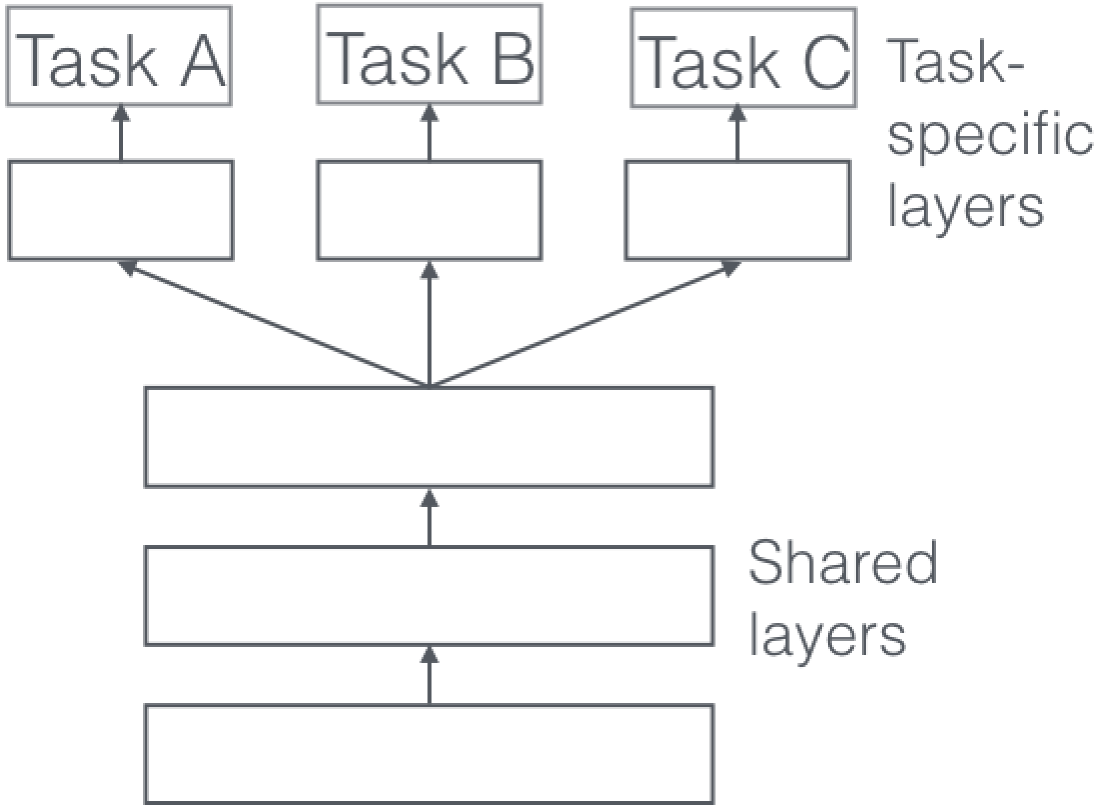
Common solution:

1. Sample tasks from the task distribution $P(\tau)$
2. Compute their losses
3. Sum the losses
4. Backpropagate
5. Go back to step 1

How will the model know which task to do?

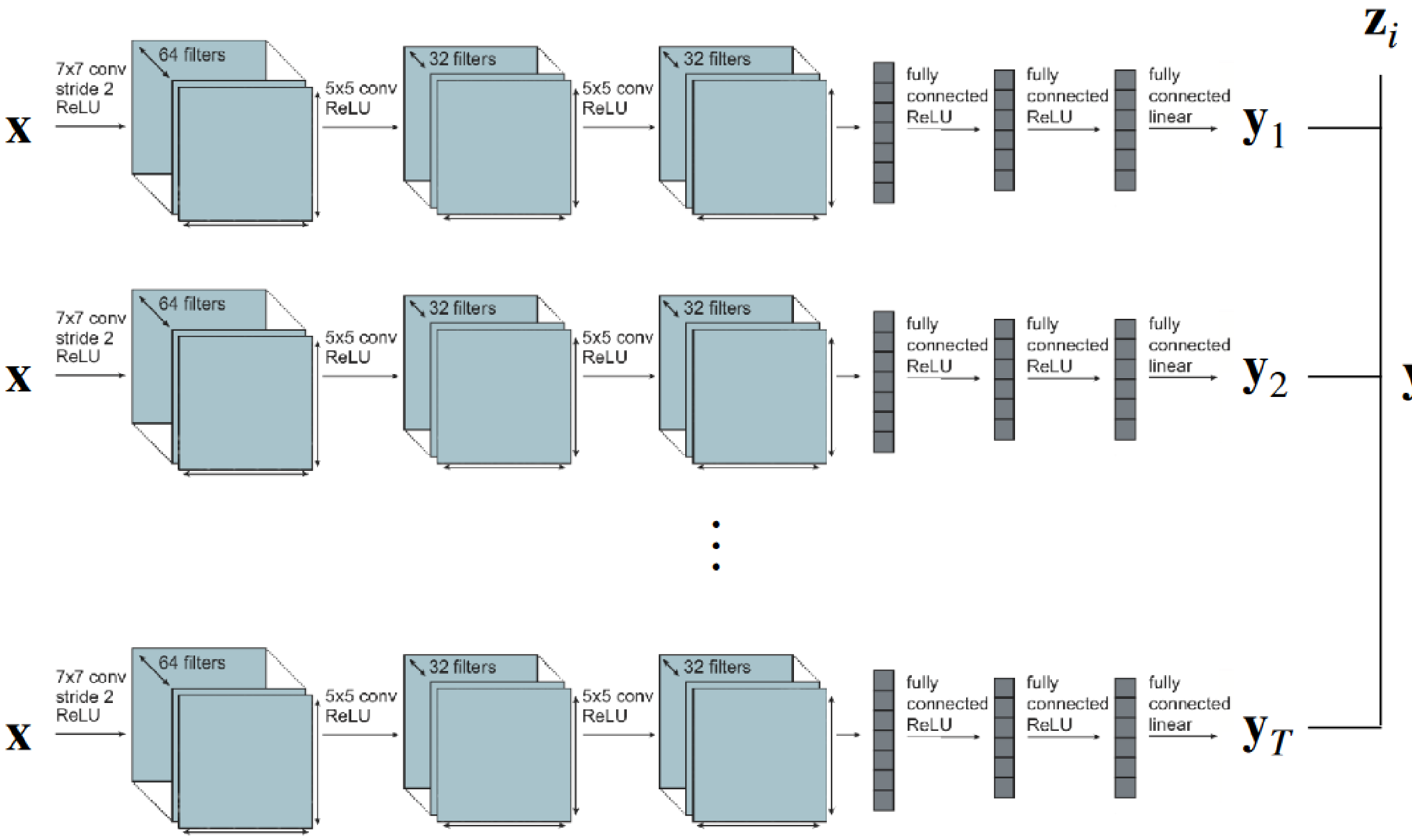
What if the sum of losses is not a good loss?

Parameter sharing



An Overview of Multi-Task Learning in Deep Neural Networks
Ruder.io

Extreme case: no parameter sharing

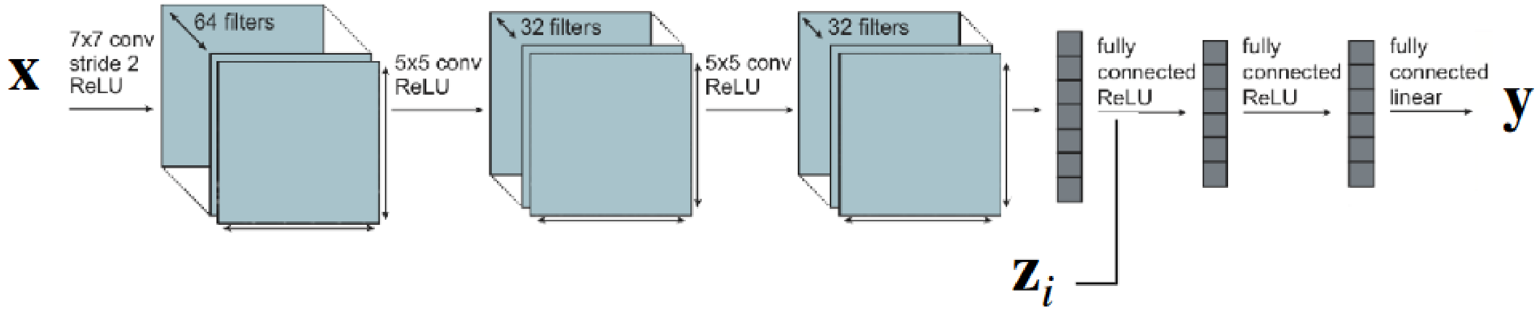


multiplicative gating

$$\mathbf{y} = \sum_j \mathbf{1}(\mathbf{z}_i = j)\mathbf{y}_j$$

From: Stanford CS330

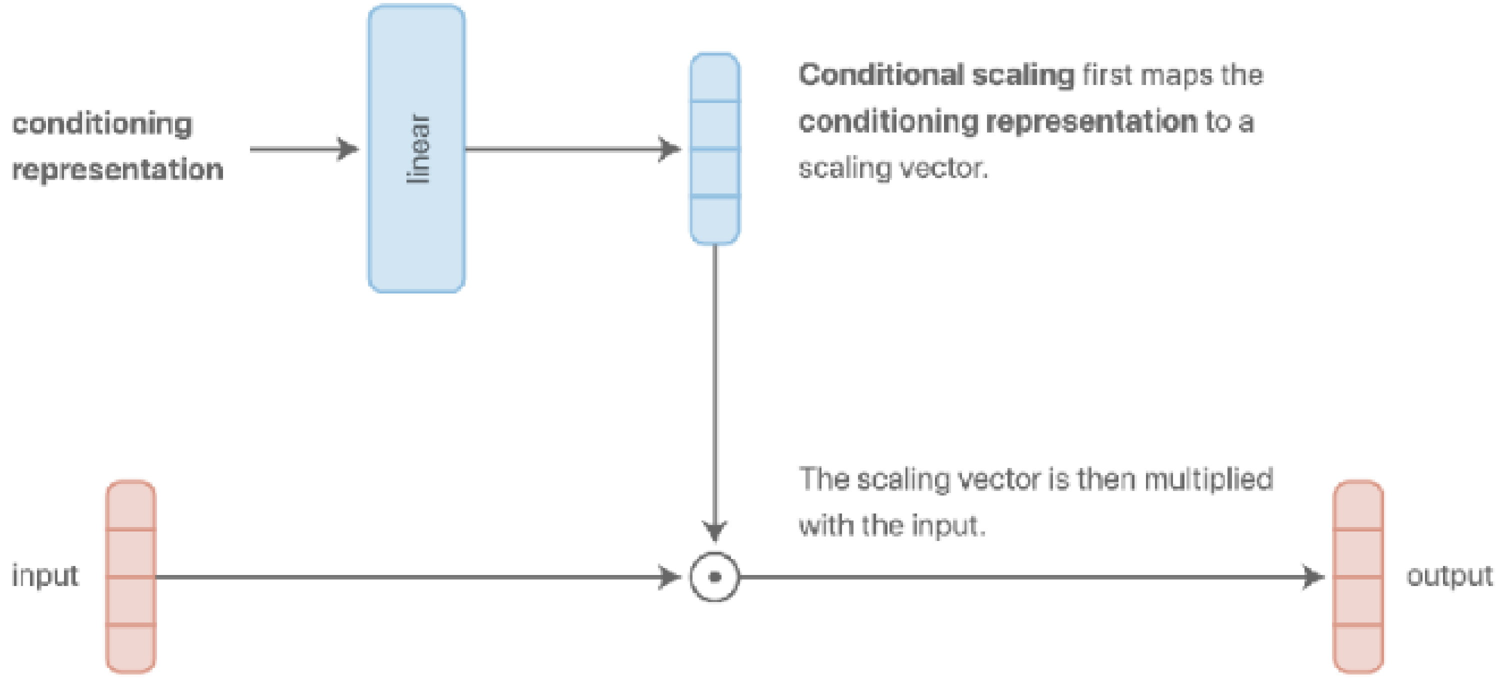
Extreme case: full parameter sharing



Concatenate $\mathbf{z}_i$ with input and/or activations

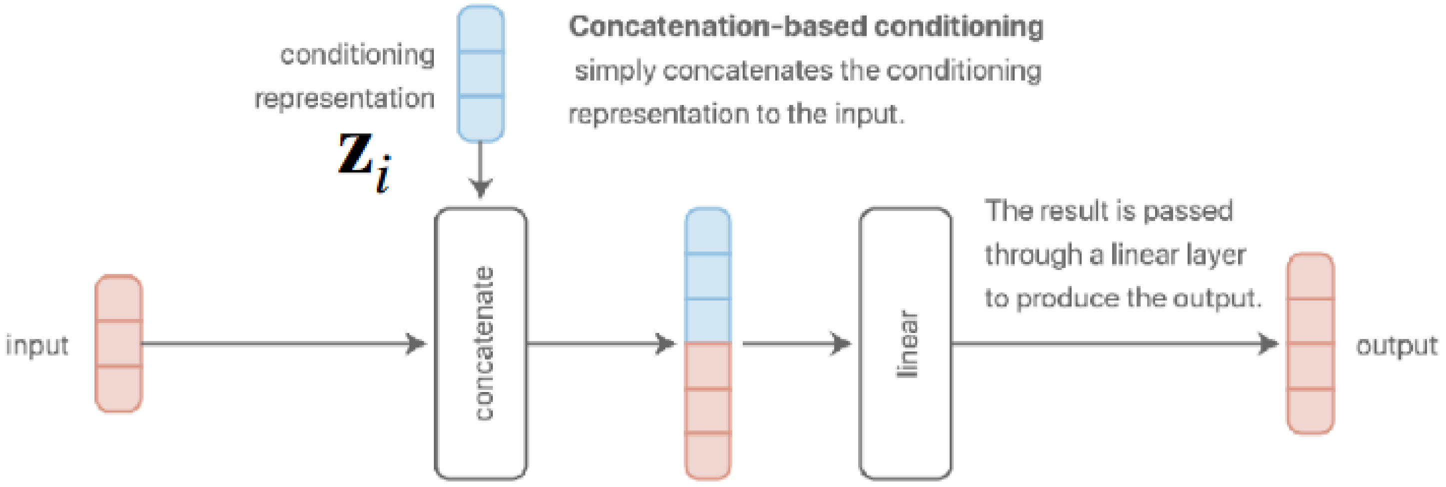
From: Stanford CS330

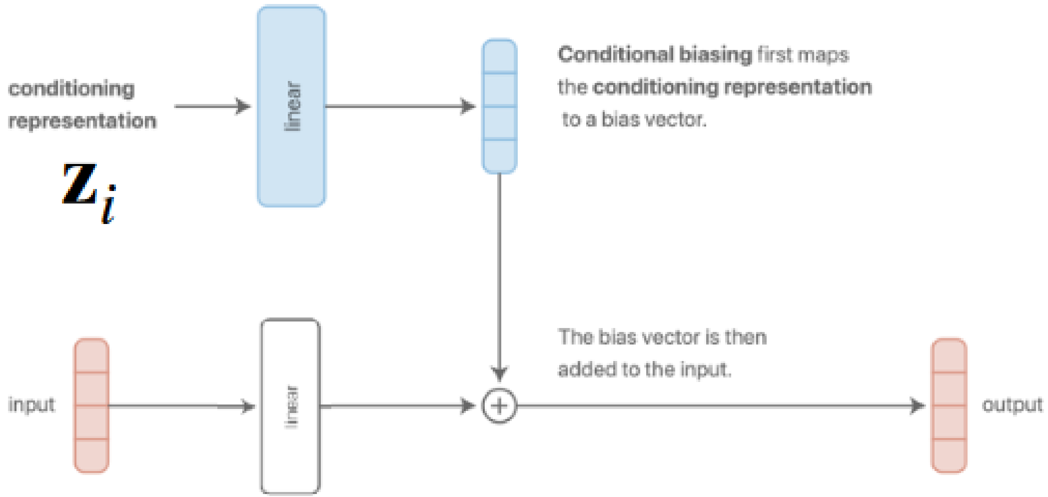
Multiplicative coding



Feature-wise transformations
Dumoulin et al., Distill 2018

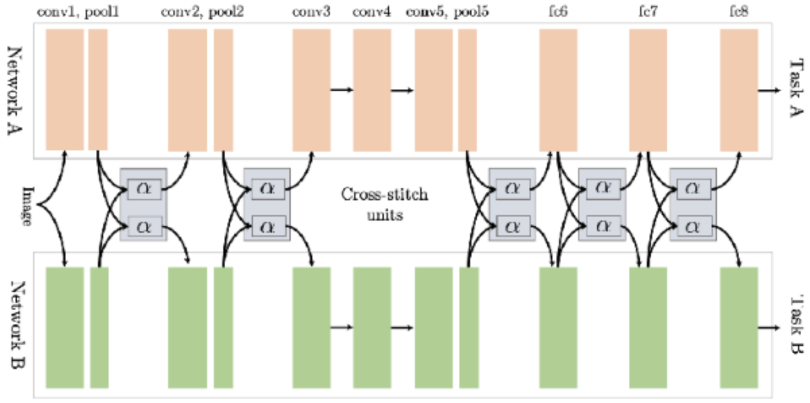
Concatenation-based coding



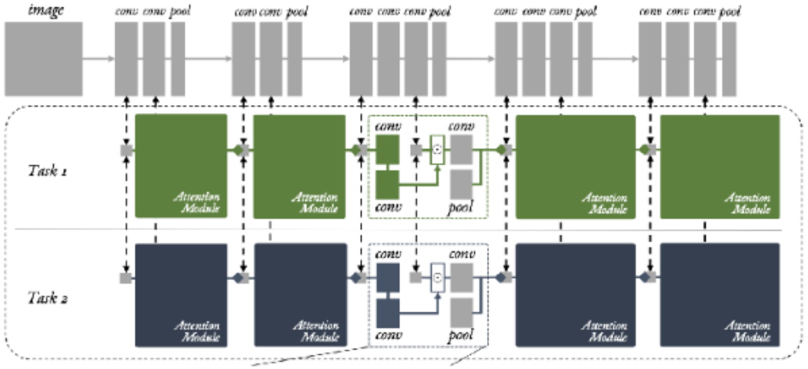


These are the same!

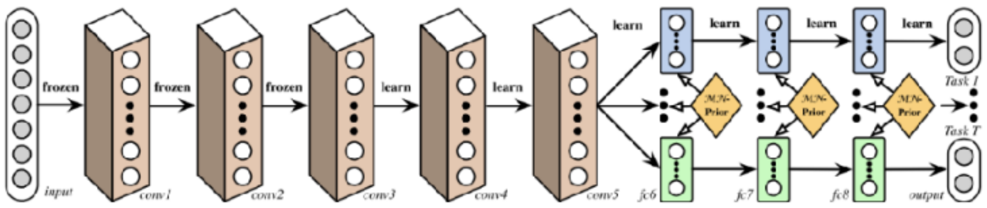
There is no right way



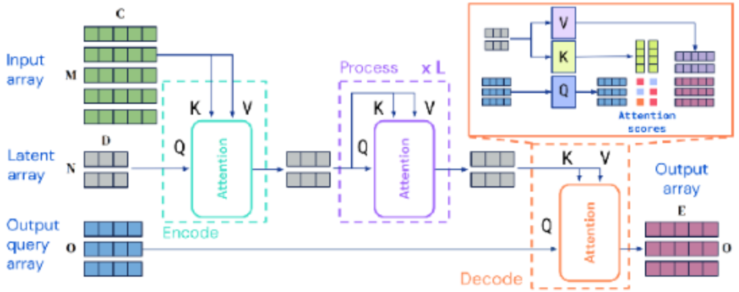
Cross-Stitch Networks. Misra, Shrivastava, Gupta, Hebert '16



Multi-Task Attention Network. Liu, Johns, Davison '18



Deep Relation Networks. Long, Wang '15



Perceiver IO. Jaegle et al. '21

From: Stanford CS330

Multi-task learning

Common solution:

1. Sample tasks from the task distribution $P(\tau)$
2. Compute their losses
3. Sum the losses
4. Backpropagate
5. Go back to step 1

How will the model know which task to do?

What if the sum of losses is not a good loss?

Multi-task learning

Common solution:

Popular heuristic: try to make gradients have similar magnitude

1. Sample tasks from the task distribution $P(\tau)$
2. Compute their losses
3. Sum the losses w/ some weights
4. Backpropagate
5. Go back to step 1

How will the model know which task to do?

What if the sum of losses is not a good loss?

Multi-task learning

Common solution:

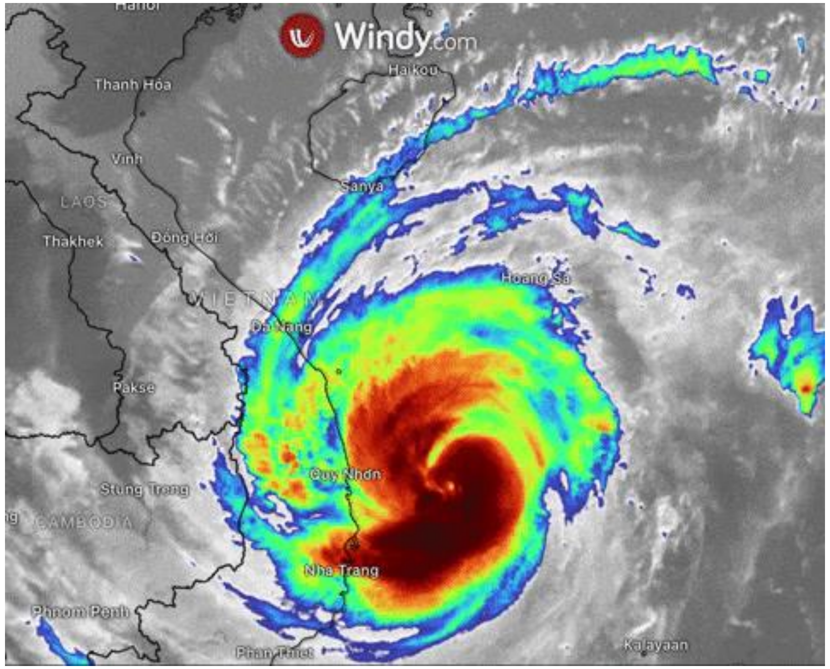
1. Sample tasks from the task distribution $P(\tau)$
2. Compute their losses
3. Take the maximum of the losses
4. Backpropagate
5. Go back to step 1

How will the model know which task to do?

What if the sum of losses is not a good loss?

Common problems

- Negative transfer



You should share less between the tasks.

How?

- Fewer parameters
- Soft-sharing

Left: Playing atari with deep reinforcement learning
Mnih et al., NeurIPS Deep Learning Workshop 2013
Right: Windy.com Community

Soft-sharing

Do not constrain the model to have the same parameters for different tasks.

Instead, penalize the model based on how different their parameters are.

Common problems

- Overfitting

Perhaps, you have little data for some of the tasks.

You should share more between the tasks.

Can we share based on task similarity!

Yes!

But what is task similarity?

Today…

- Multi-task Learning: Learn multiple tasks together
- Transfer Learning: Learn multiple tasks and transfer your knowledge to a new one
- Meta Learning: Learn multiple tasks such that adapting to a new task will be easy

Transfer learning

Training: Have access to tasks $\tau_1, \tau_2, \ldots, \tau_n$, but not τ_{n+1}.

Transfer: Have access to task τ_{n+1}, but not $\tau_1, \tau_2, \ldots, \tau_n$.

Transfer learning

Common solution:

Training:

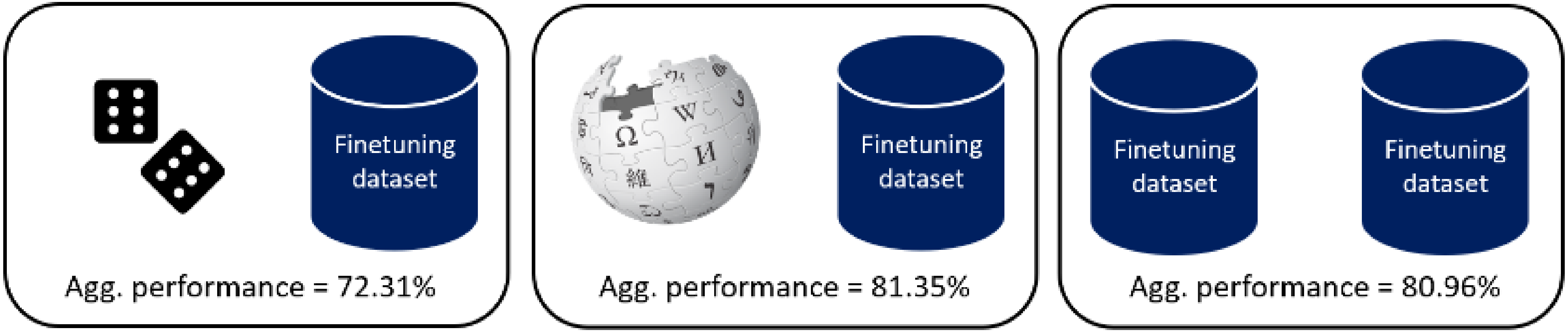
1. Run your favorite (multi-task) learning algorithm on $\tau_1, \tau_2, \ldots, \tau_n$

Transfer:

2. Fine-tune the model on τ_{n+1}

This is the idea behind using ImageNet features or BERT embeddings!

Transfer learning



Downstream datasets make surprisingly good pretraining corpora
Krishna et al., ACL 2023

Transfer learning

Common solution:

Training:

1. Run your favorite (multi-task) learning algorithm on $\tau_1, \tau_2, \ldots, \tau_n$

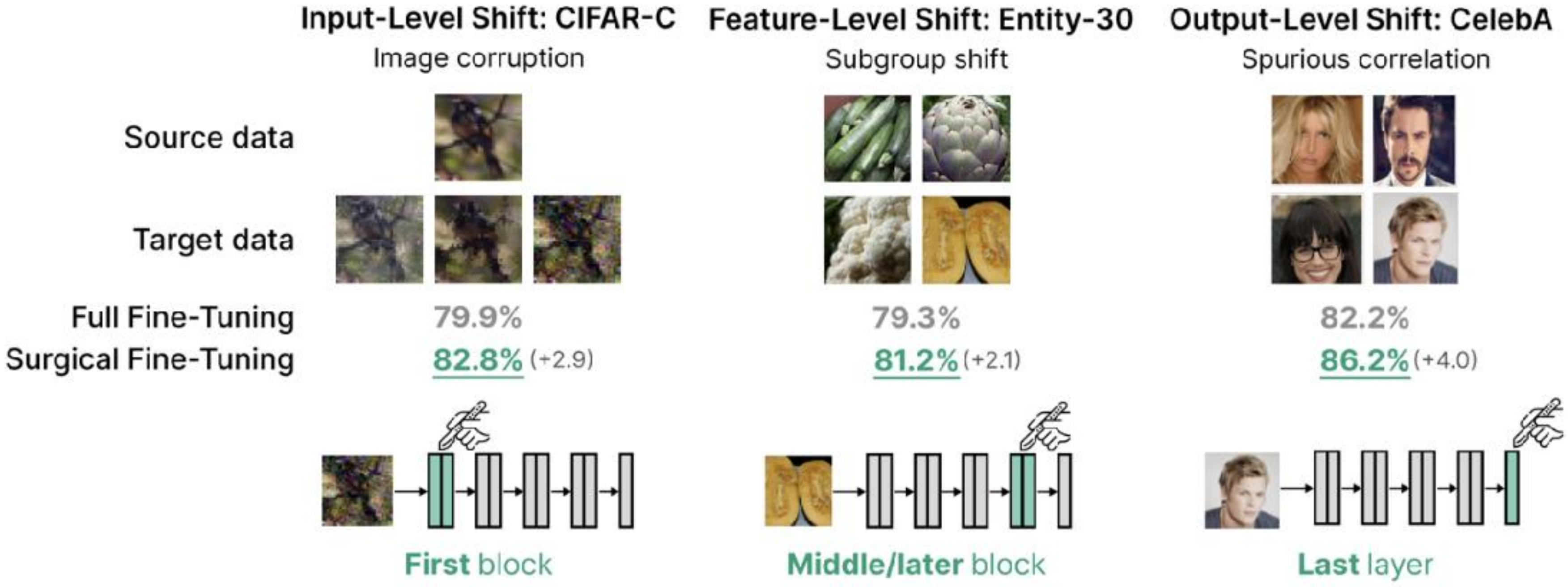
Transfer:

2. Fine-tune the model on τ_{n+1}

Fine-tune what?

Fine-tune what?

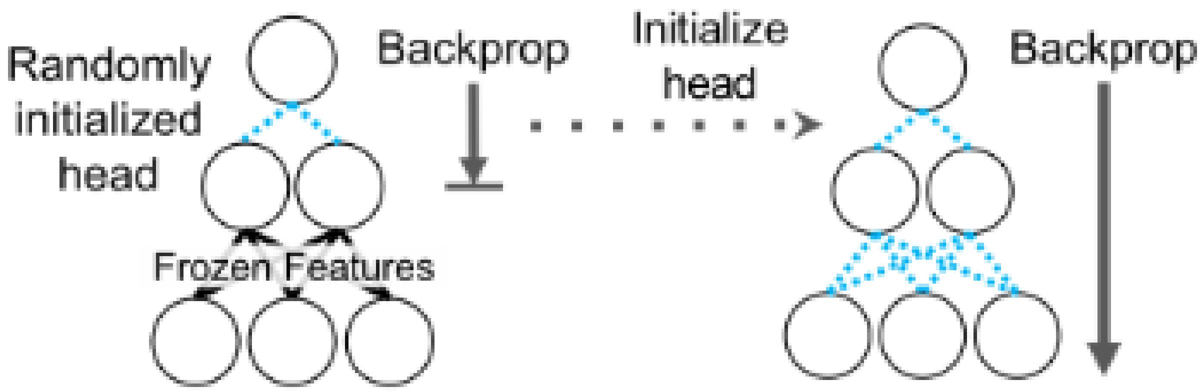
It depends.



Surgical fine-tuning improves adaptation to distribution shifts
Lee et al., ICLR 2023

Fine-tune what?

A good default:



Slide: Chelsea Finn (Stanford)
Image: Fine-Tuning can distort pretrained features
and underperform out-of-distribution
Kumar et al., ICLR 2022

Transfer learning

What if our dataset on the target set is so small that even transfer learning does not help?

Today…

- Multi-task Learning: Learn multiple tasks together
- Transfer Learning: Learn multiple tasks and transfer your knowledge to a new one
- Meta Learning: Learn multiple tasks such that adapting to a new task will be easy

Meta-learning

From: Stanford CS330

Meta-learning

Training: Have access to tasks $\tau_1, \tau_2, \ldots, \tau_n$, but not τ_{n+1}.

Transfer: Have access to task τ_{n+1}, but not $\tau_1, \tau_2, \ldots, \tau_n$.

Assumption:

τ_{n+1} comes from the same task distribution as $\tau_1, \tau_2, \ldots, \tau_n$.

Black-box adaptation

- Design a giant neural network that takes the datasets as the input and outputs the parameters of a smaller network.

Yes, I really said this.

But sometimes we can get away with lower dimensional vectors.

- The smaller network performs the task τ_{n+1}.

Optimization-based adaptation

Learn a model such that when we take one (or some) gradient step in task τ_{n+1}, it will perform good.

$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{n} L\left(\theta - \alpha \nabla_\theta L\left(\theta, \tau_i^{tr}\right), \tau_i^{ts}\right)$$

Optimization-based adaptation

1. Sample task τ_i
2. Compute $\phi \leftarrow \theta - \nabla_\theta L(\theta, \tau_i^{tr})$
3. Update θ using $\nabla_\theta L(\phi, \tau_i^{ts})$

Note we will need the second gradient!

From: Stanford CS330

Next time…

Week 11 Fri, Nov 3	*Presentation* Safe and robust learning *Lecture* Multi-agent learning	**Due** **Project Milestone Report** • Jeon et al., Shared Autonomy with Learned Latent Actions (2020). • Sui et al., Safe Exploration for Optimization with Gaussian Processes (2015). • Achiam et al., Constrained Policy Optimization (2017). • Robey et al., Learning Control Barrier Functions from Expert Demonstrations (2020). • Bansal and Tomlin, Deepreach: A Deep Learning Approach to High-dimensional Reachability (2021).