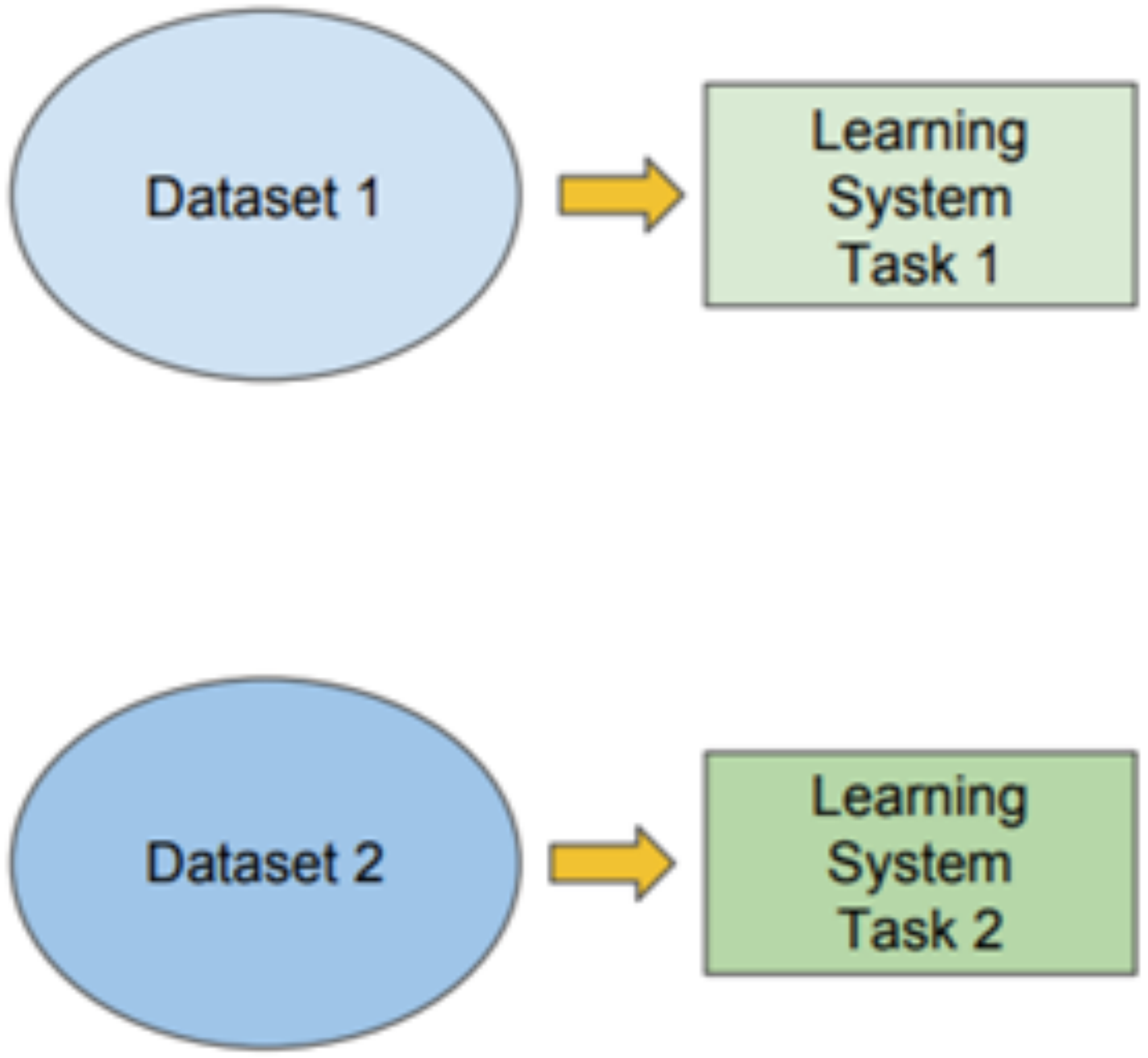
# Aprendizado Profundo (Deep Learning)

## Transfer Learning

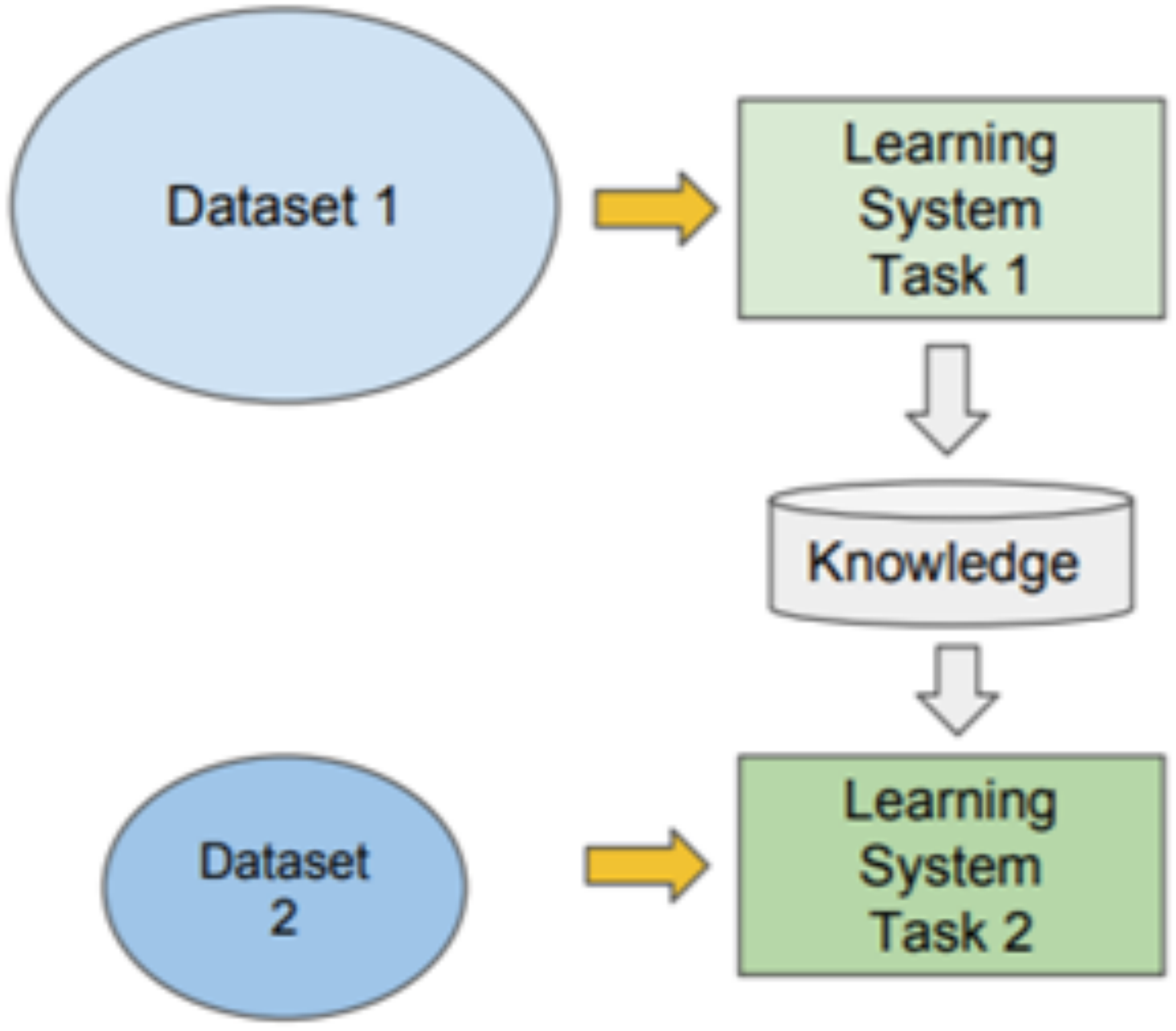
**Dario Oliveira (dario.oliveira@fgv.br)**

# Traditional ML vs Transfer Learning

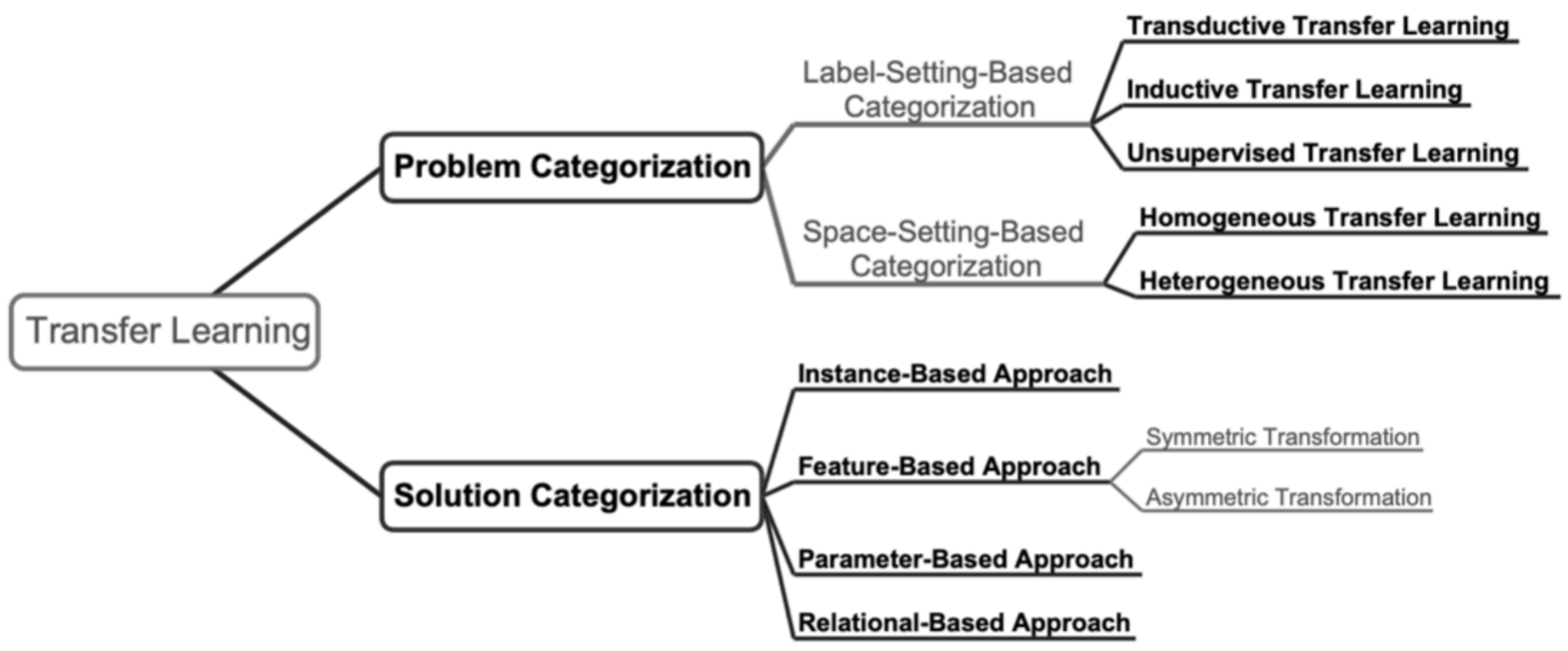
- Isolated, single task learning:
    - Knowledge is not retained or accumulated. Learning is performed w.o. considering past learned knowledge in other tasks



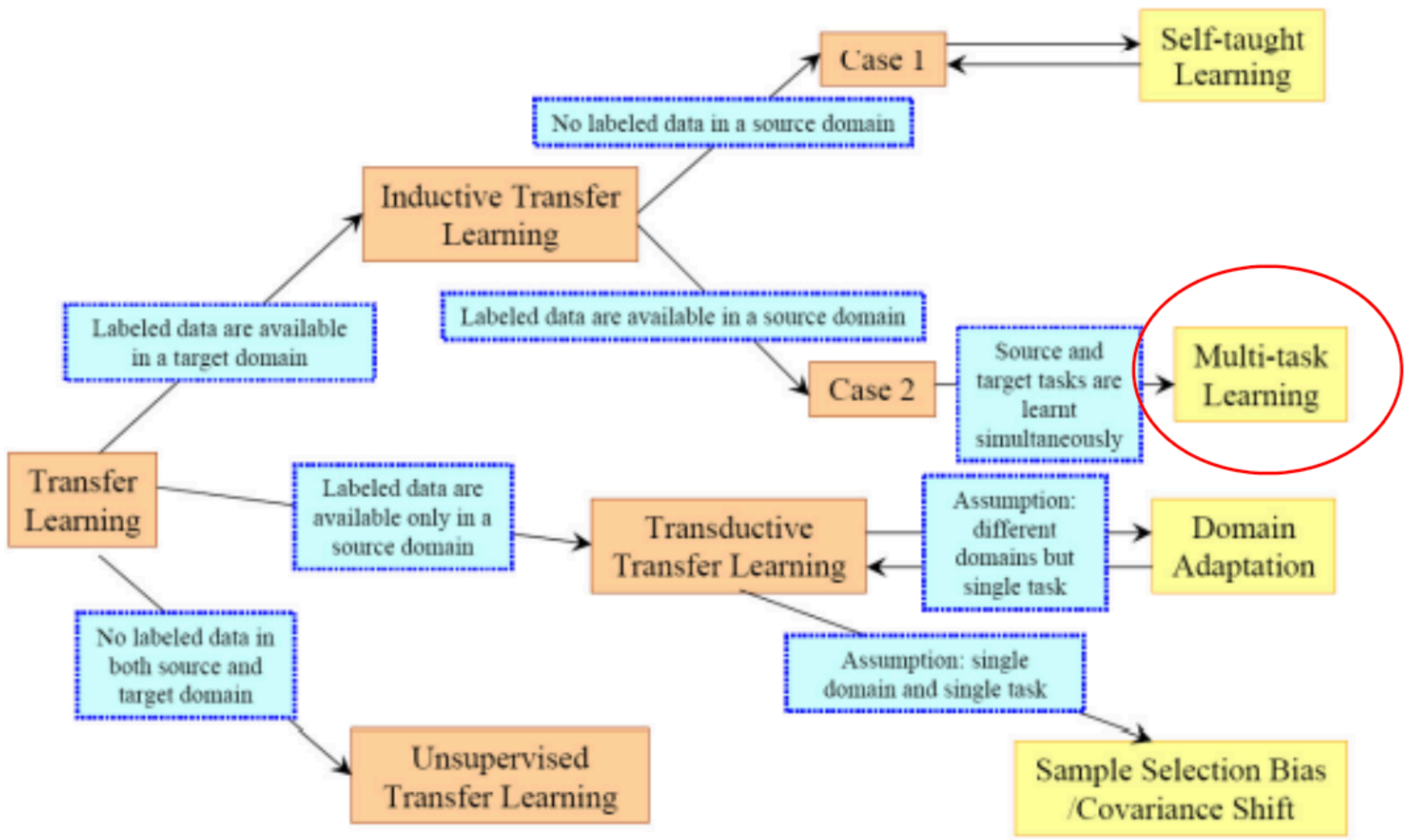

- Learning of a new tasks relies on the previous learned tasks:
    - Learning process can be faster, more accurate and/or need less training data



# Strategies: complex taxonomy in dispute




"A Comprehensive Survey on Transfer Learning"' Zhuang et al., ArXiv, 2020.




"A Survey on Transfer Learning", Jialin and Yang , IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, 2009

# Definitions

A **Domain** consists of two components: $D = \{\mathcal{X}, P(X)\}$

- Feature space: $\mathcal{X}$
- Marginal distribution: $P(X)$, $X = \{x_1, \cdots, x_n\}, x_i \in \mathcal{X}$

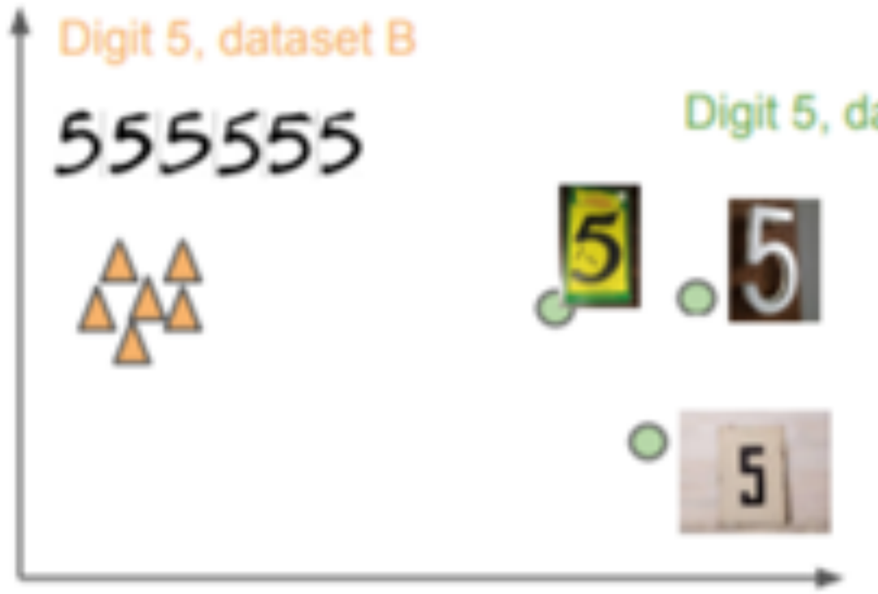
For a given domain **D**, a **Task** is defined by two components:

$$T = \{\mathcal{Y}, P(Y|X)\} = \{\mathcal{Y}, \eta\} \qquad Y = \{y_1, \cdots, y_n\}, y_i \in \mathcal{Y}$$

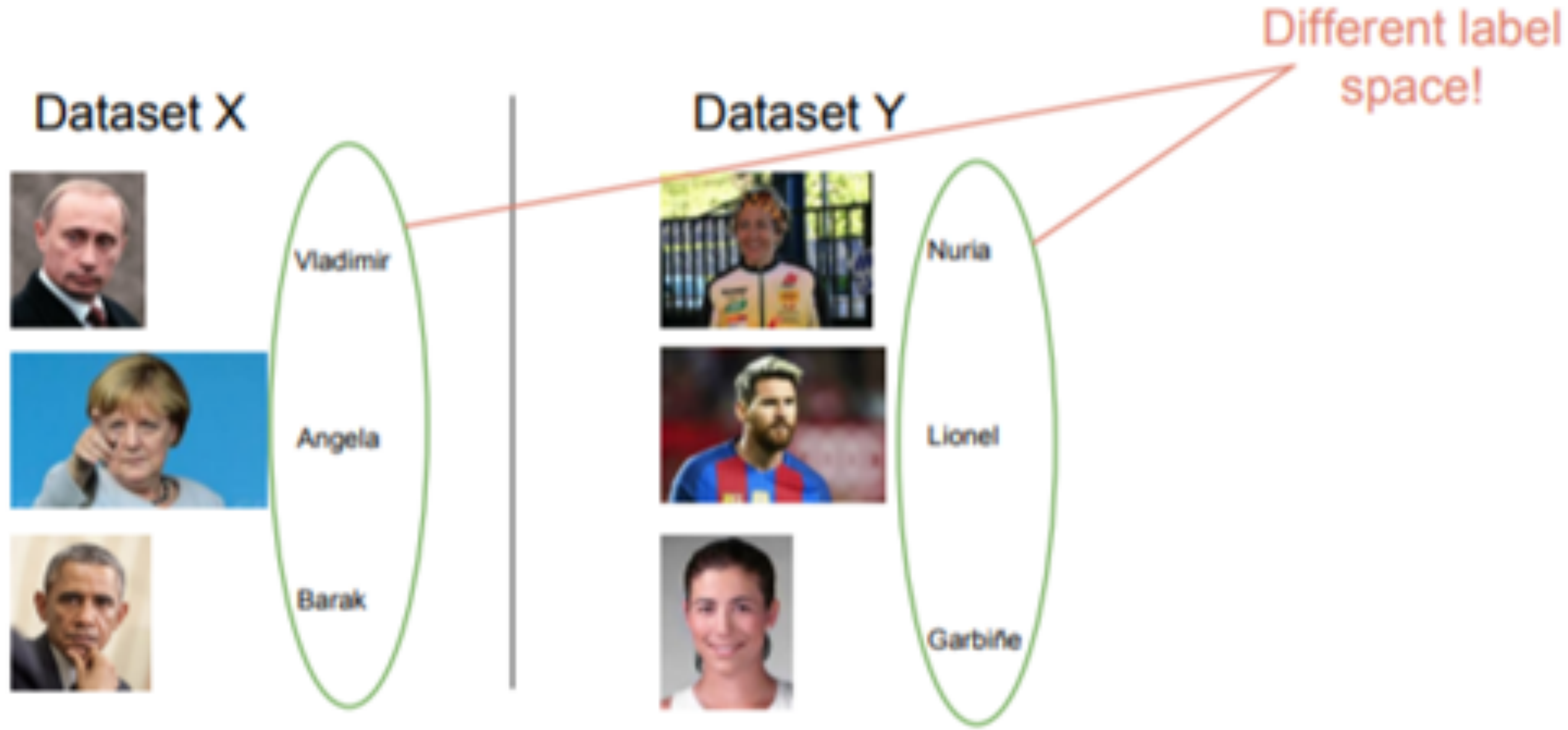
- A label space: $\mathcal{Y}$
- A predictive function $\eta$, learned from *feature vector/label* pairs, $(x_i, y_i)$, $x_i \in \mathcal{X}, y_i \in \mathcal{Y}$
- For each feature vector in the domain, $\eta$ predicts its corresponding label: $\eta(x_i) = y_i$

# Definitions

If two domains are different, they may have different feature spaces or different marginal distributions




If two tasks are different, they may have different label spaces or different conditional distributions



# Transfer Learning in DL

**Myth**: you can't do deep learning unless you have a million labeled examples for your problem.

**Reality**

- You can learn useful representations from **unlabeled data**
- You can train on a nearby **surrogate objective** for which it is easy to generate labels
- You can **transfer** learned representations from a related task

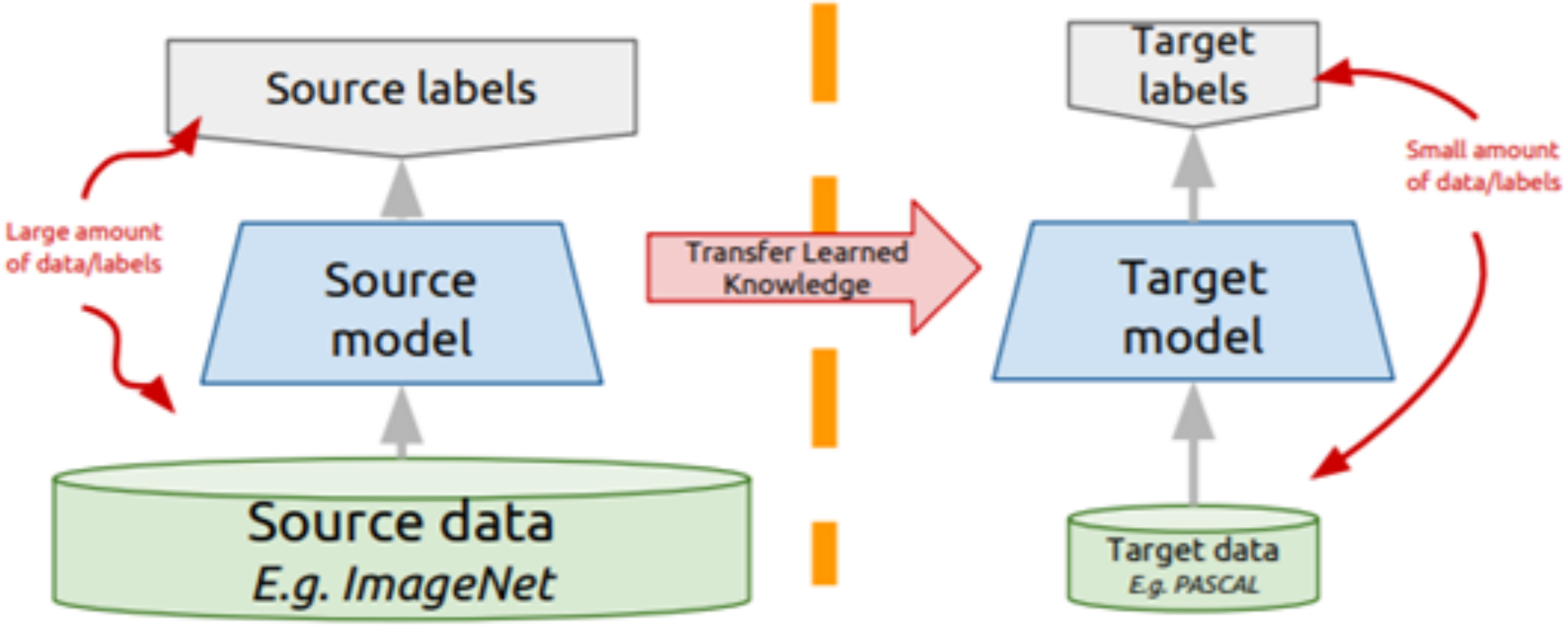
# Transfer Learning Basic Idea

Instead of training a deep network from scratch for your task:

- Take a network trained on a different domain for a different **source task**
- Adapt it for your domain and your **target task**

Variations:

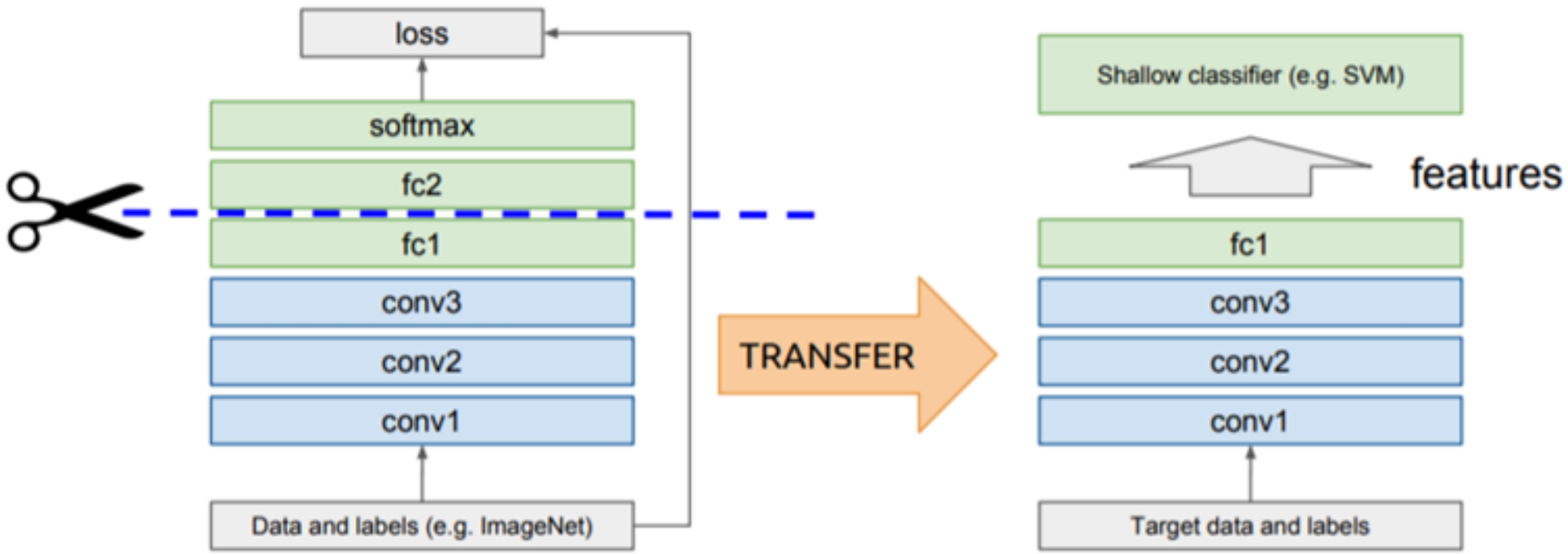
- Same domain, different task
- Different domain, same task



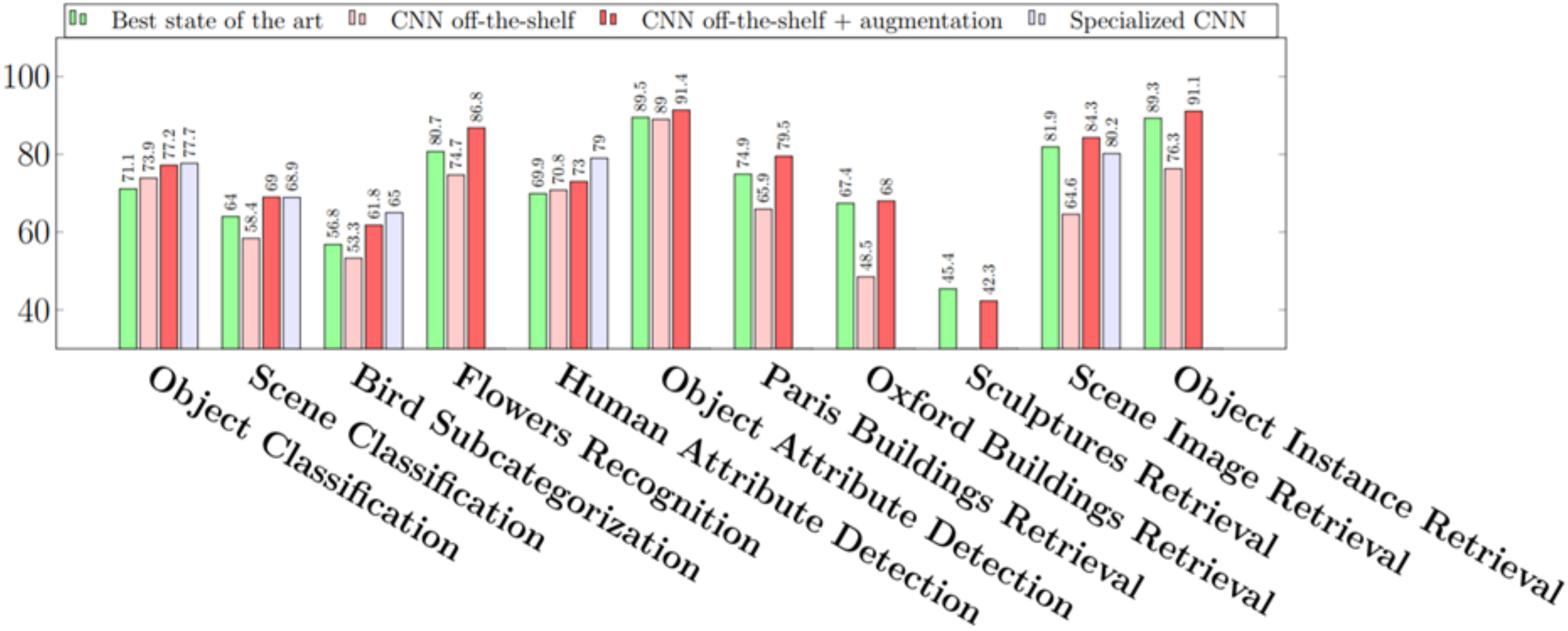
# Transfer Learning using pre-trained models as feature extractors

Idea: use outputs of one or more layers of a network trained on a different task as generic feature detectors. Train a new shallow model on these features.

Assumes that $D_S = D_T$

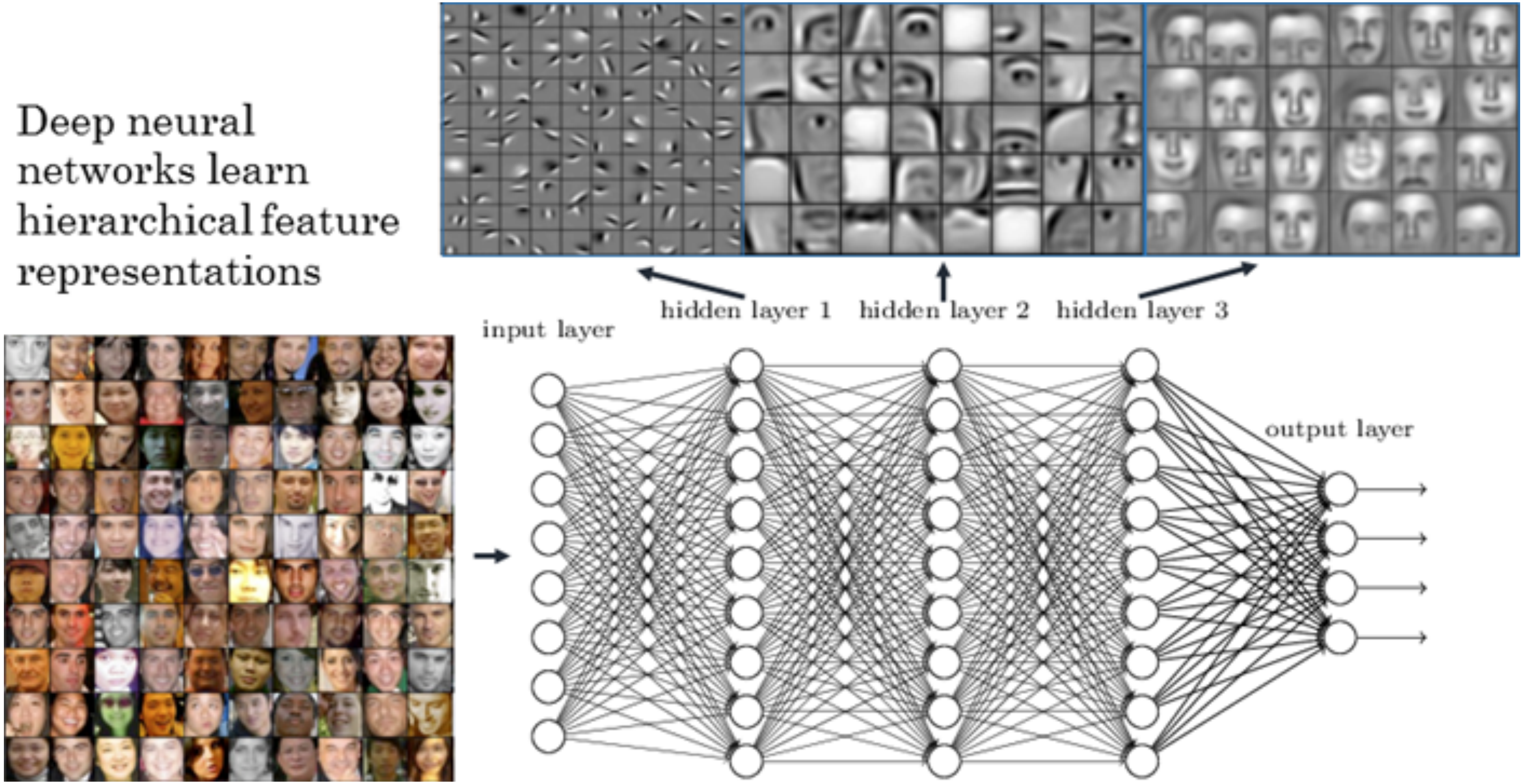


# Transfer Learning works!!



# Transfer Learning fine-tuning pre-trained models

Deep neural networks learn hierarchical feature representations



# Transfer Learning fine-tuning pre-trained models

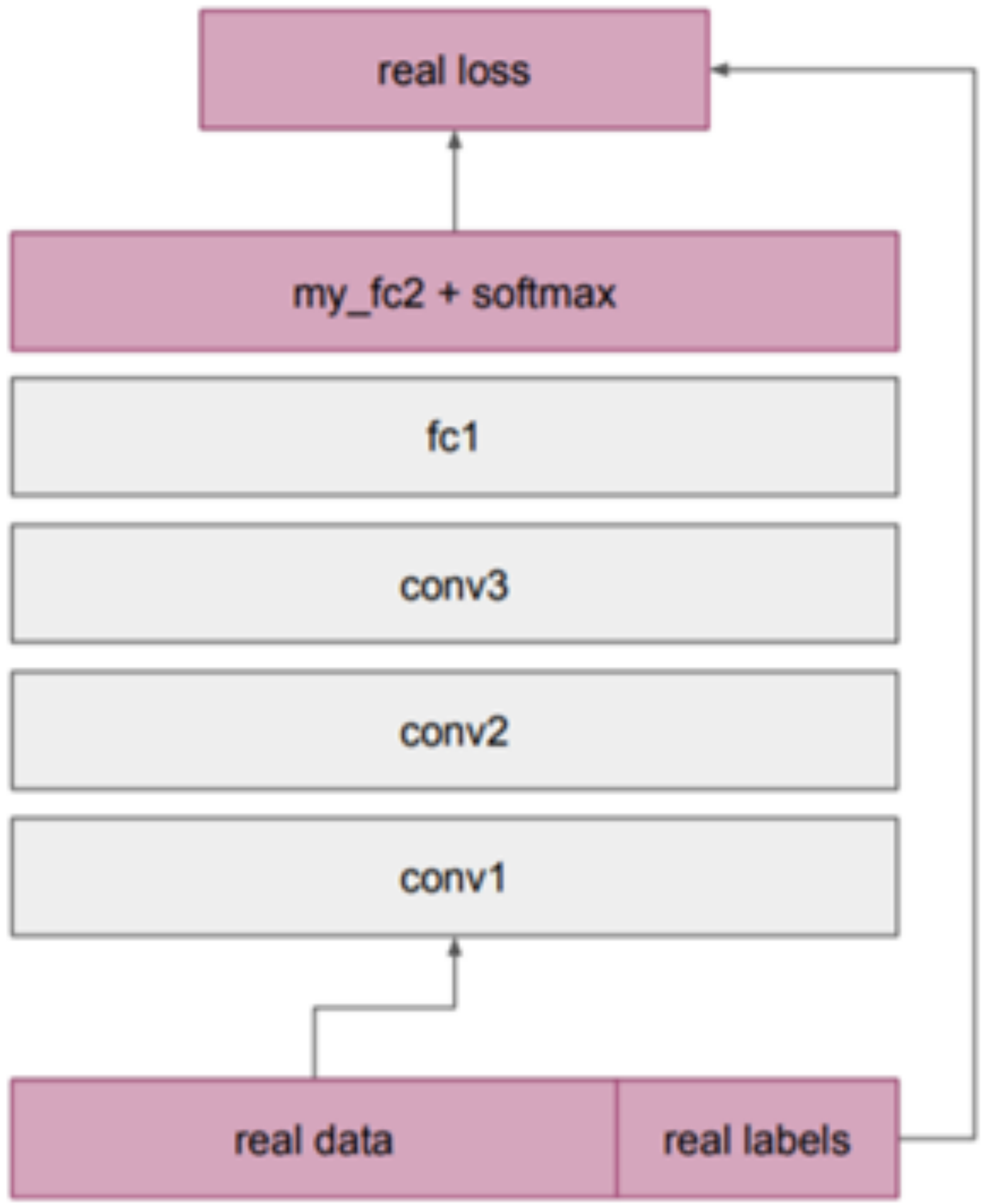
Train deep net on “nearby” task for which it is easy to get labels using standard backprop

- E.g. ImageNet classification
- Pseudo classes from augmented data
- Slow feature learning, ego-motion

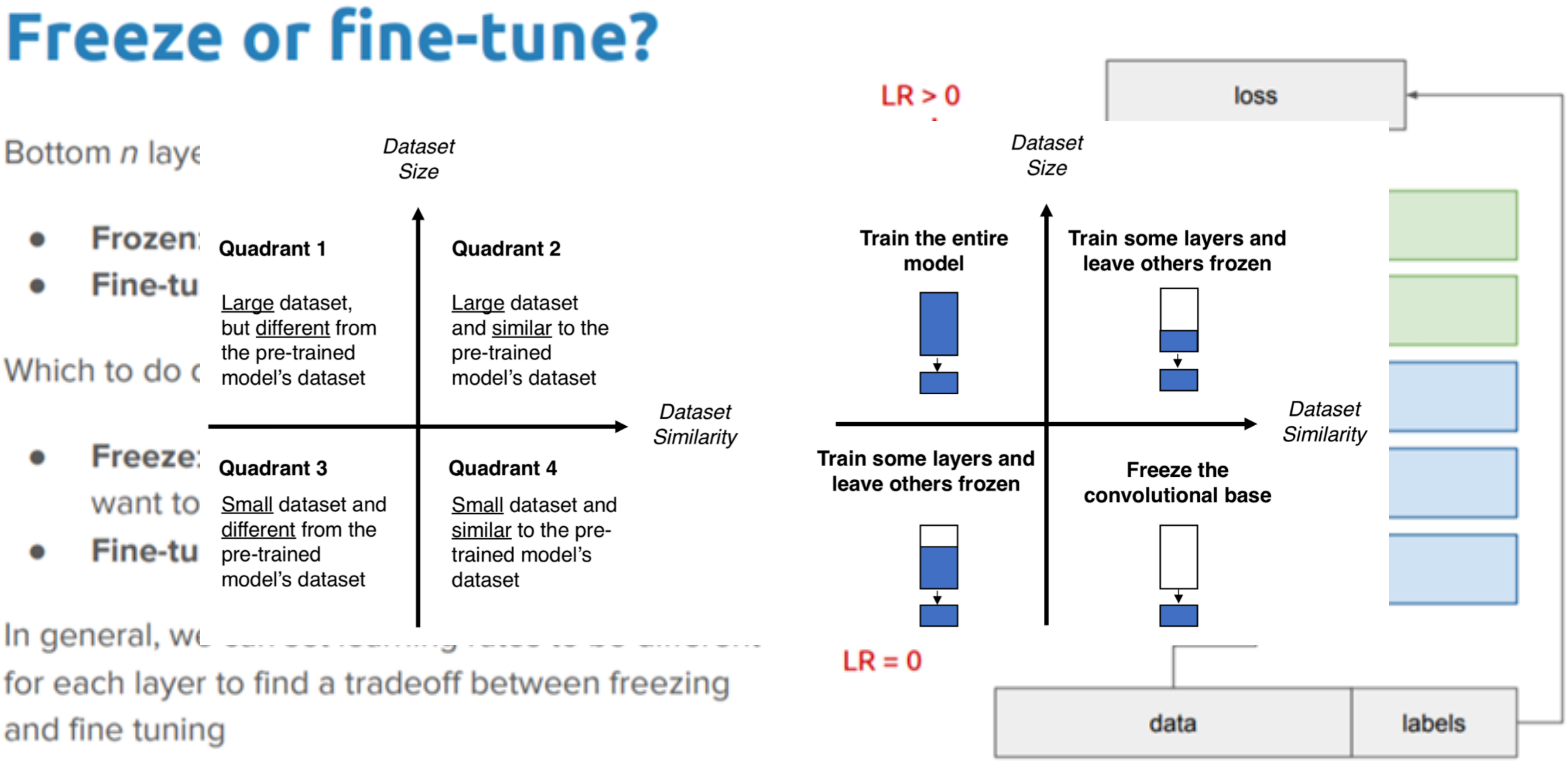
Cut off top layer(s) of network and replace with supervised objective for target domain

**Fine-tune** network using backprop with labels for target domain until validation loss starts to increase

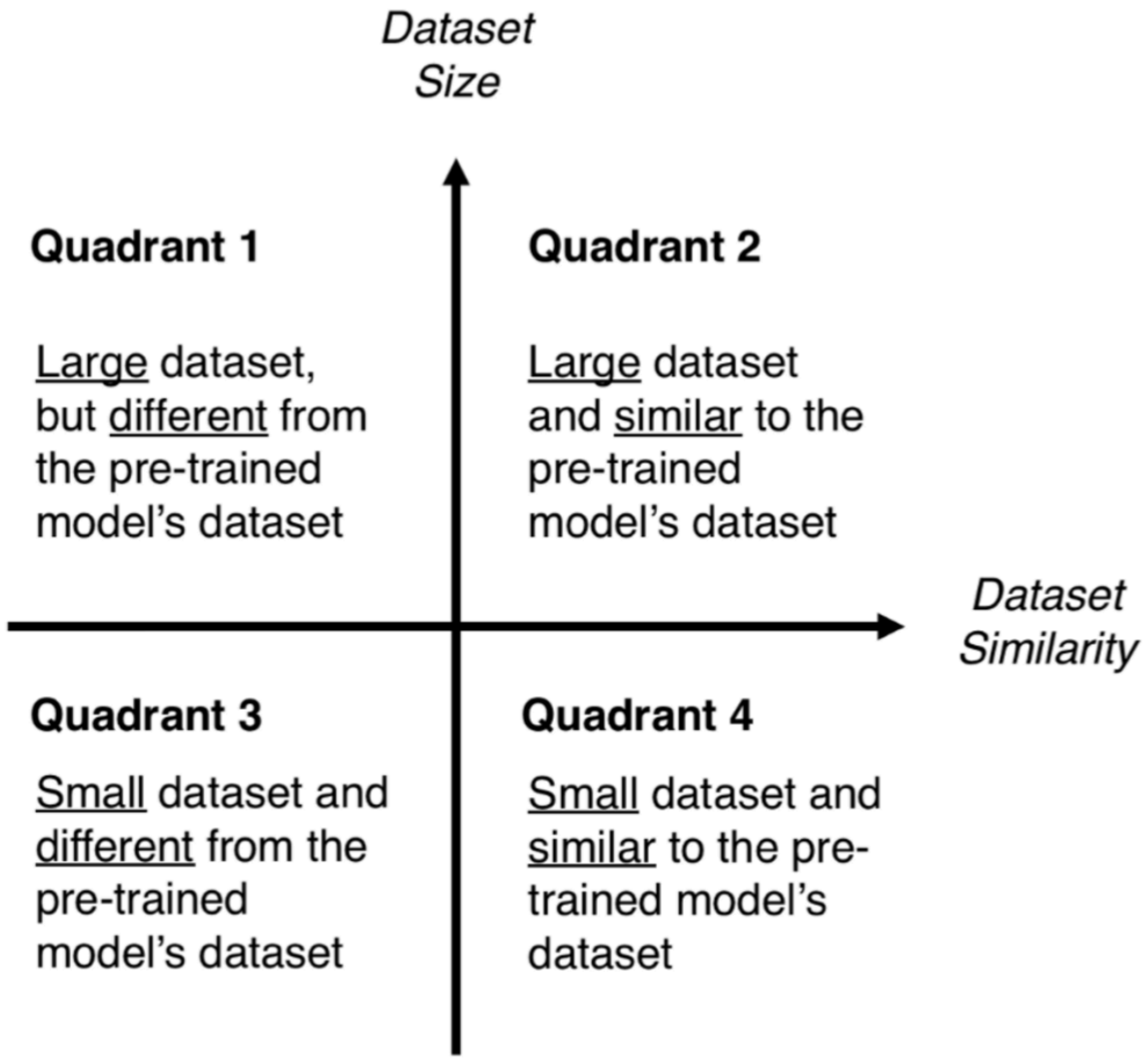
Aligns $D_S$ with $D_T$

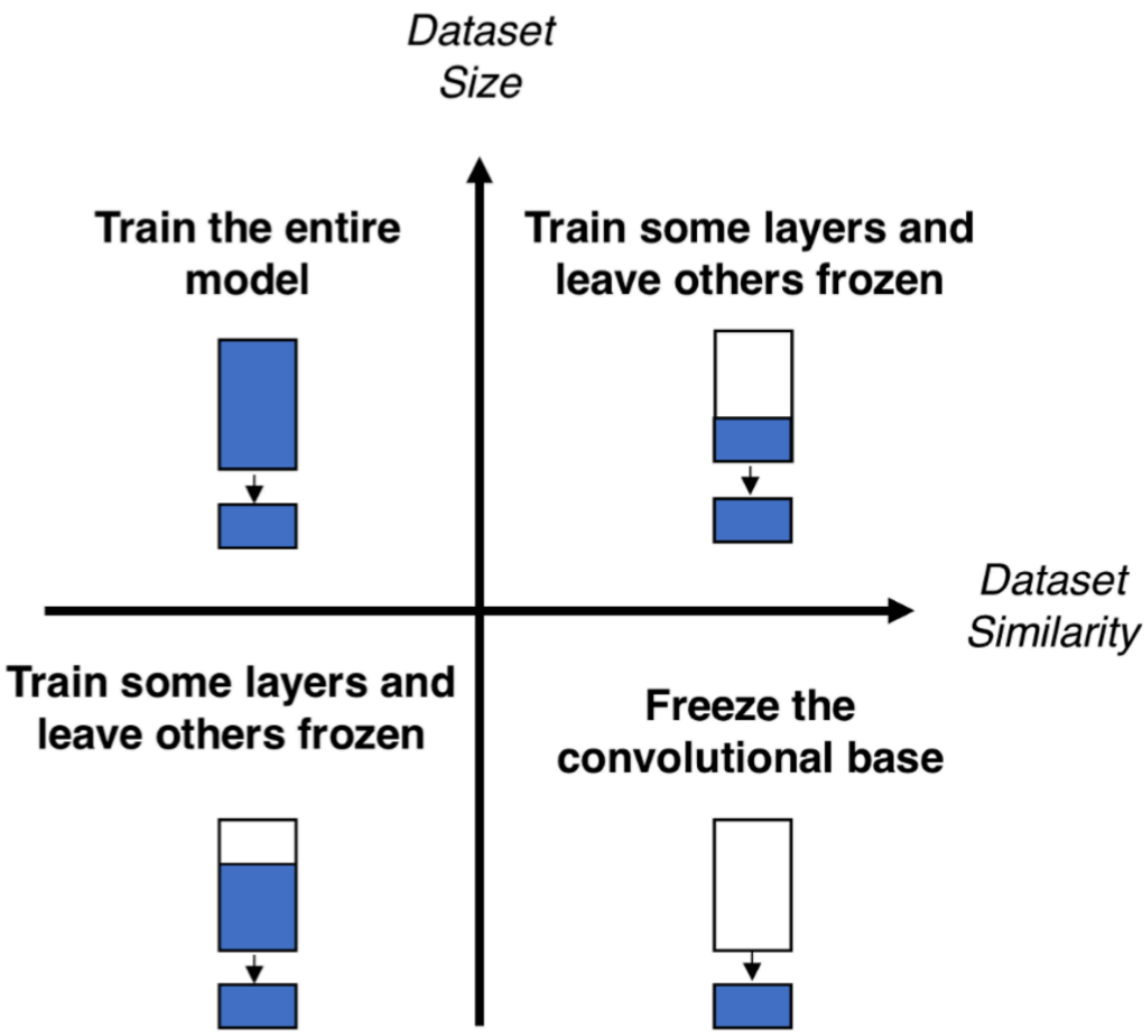


# Transfer Learning fine-tuning pre-trained models



# Transfer Learning fine-tuning pre-trained models



# How transferable are features?

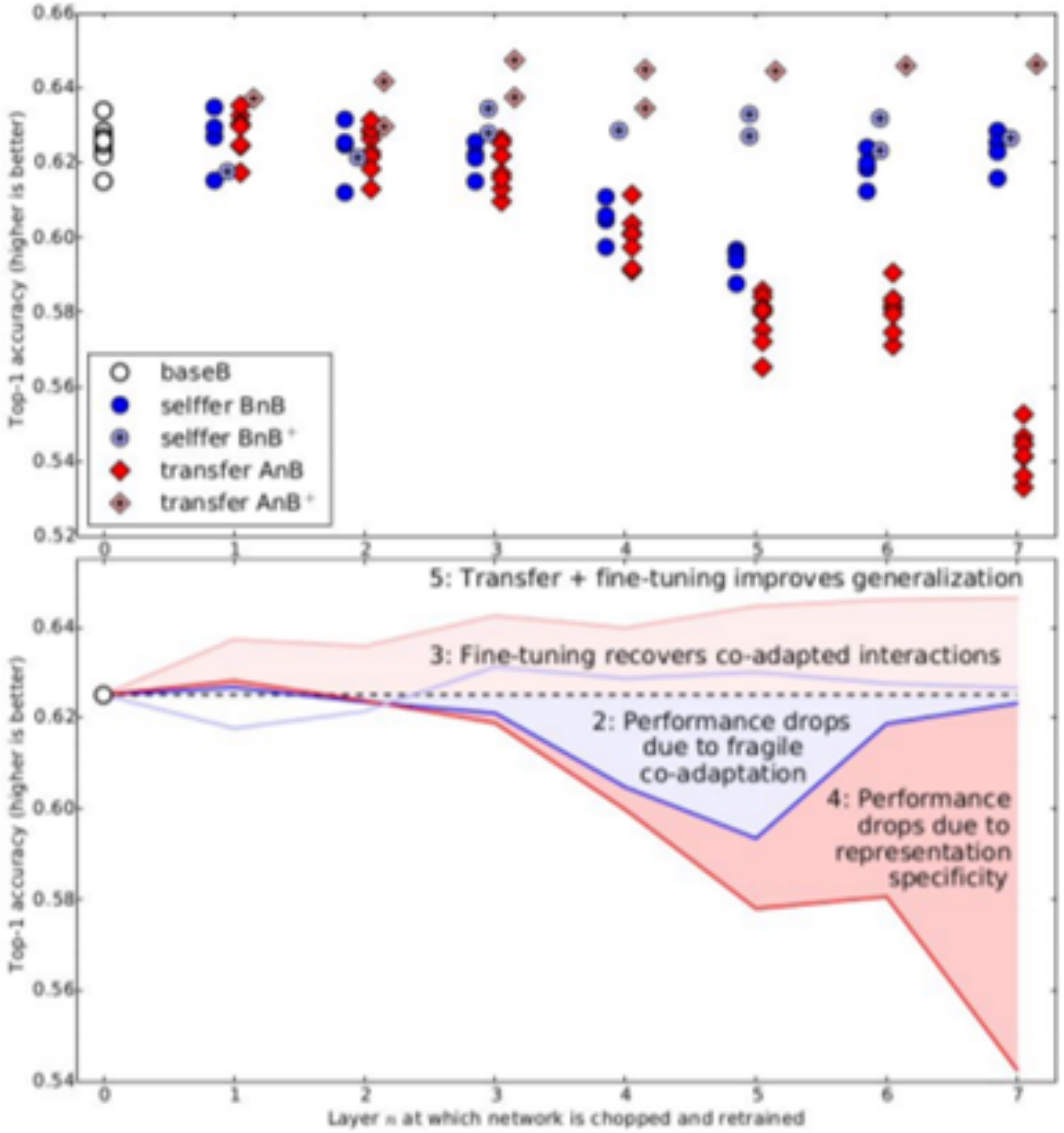
Transferability is negatively affected by two distinct issues:

- The specialization of higher layer neurons
- Optimization difficulties related to splitting networks between co-adapted neurons

Fine-tuning improves generalization when sufficient examples are available.

Transfer learning and fine tuning often lead to better performance than training from scratch on the target dataset.

Even features transferred from distant tasks are often better than random initial weights!




Yosinki et al. **How transferable are features in deep neural networks.** NIPS 2014. https://arxiv.org/abs/1411.1792

# Next Lecture

**Thursday**
**Lab 1 + 1st Programming Assignment**

*Transfer Learning*

See you next class! ☺