Transfer Learning

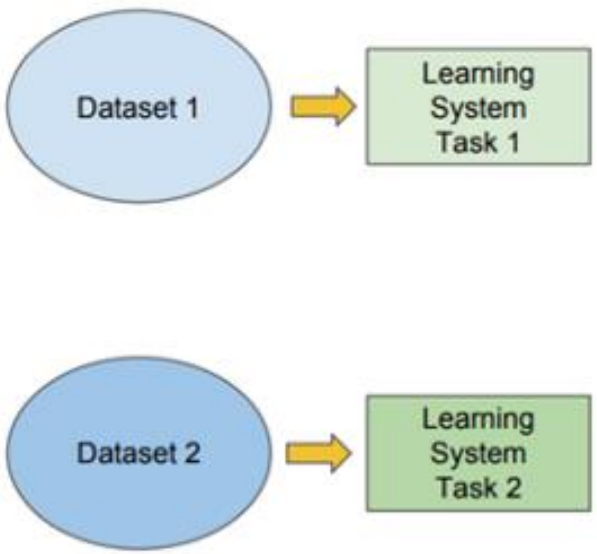
ImageNet Challenge

IMAGENET

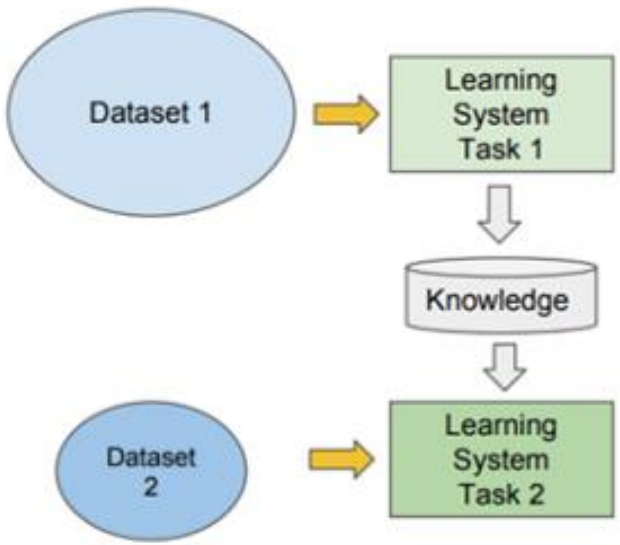
- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.

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



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 η, 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, η predicts its corresponding label: $\eta(x_i) = y_i$

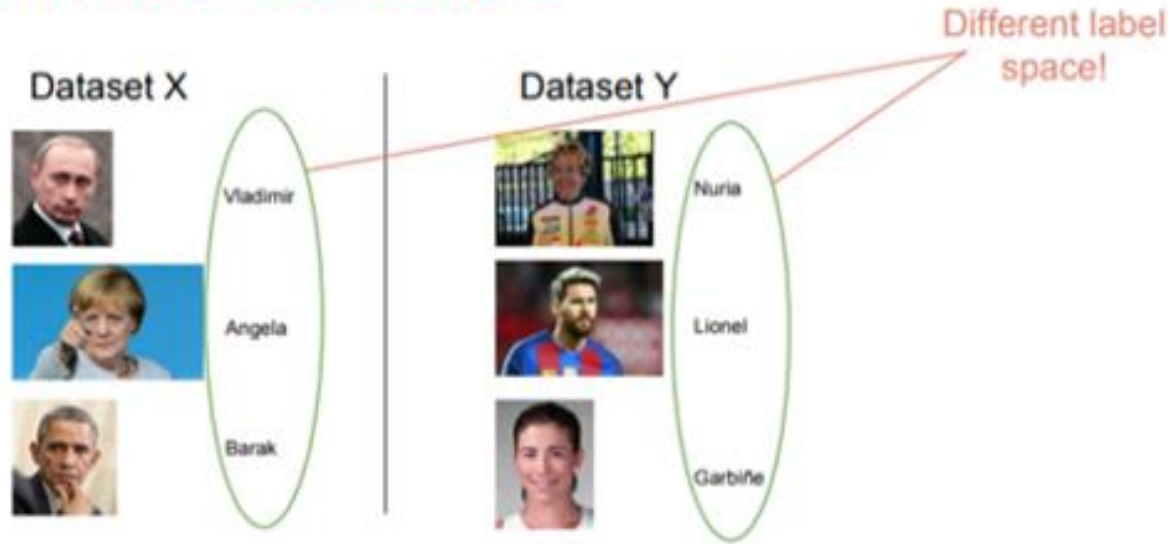
Given a source domain $\mathcal{D}_S$, a corresponding source task $\mathcal{T}_S$, as well as a target domain $\mathcal{D}_T$ and a target task $\mathcal{T}_T$, the objective of transfer learning now is to enable us to learn the target conditional probability distribution $P(Y_T|X_T)$ in $\mathcal{D}_T$ with the information gained from $\mathcal{D}_S$ and $\mathcal{T}_S$ where $\mathcal{D}_S \neq \mathcal{D}_T$ or $\mathcal{T}_S \neq \mathcal{T}_T$. In most cases, a limited number of labeled target examples, which is exponentially smaller than the number of labeled source examples are assumed to be available.

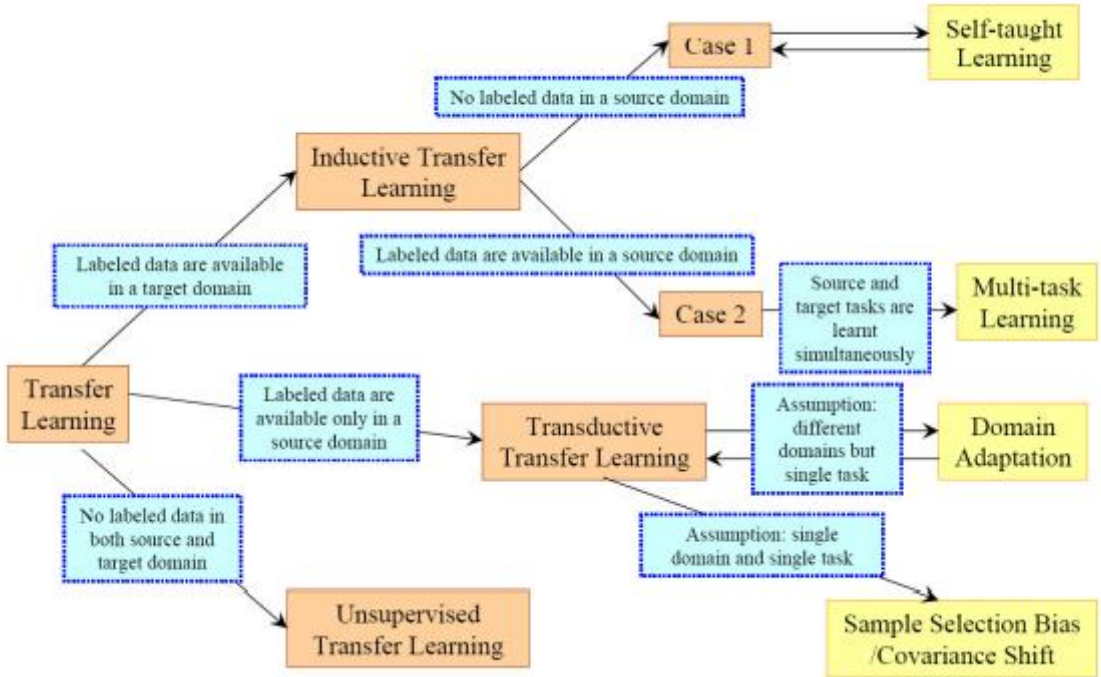
Given source and target domains $\mathcal{D}_S$ and $\mathcal{D}_T$ where $\mathcal{D} = \{\mathcal{X}, P(X)\}$ and source and target tasks $\mathcal{T}_S$ and $\mathcal{T}_T$ where $\mathcal{T} = \{\mathcal{Y}, P(Y|X)\}$ source and target conditions can vary in four ways, which we will illustrate in the following again using our document classification example:

1. $\mathcal{X}_S \neq \mathcal{X}_T$. The feature spaces of the source and target domain are different, e.g. the documents are written in two different languages. In the context of natural language processing, this is generally referred to as cross-lingual adaptation.
2. $P(X_S) \neq P(X_T)$. The marginal probability distributions of source and target domain are different, e.g. the documents discuss different topics. This scenario is generally known as domain adaptation.
3. $\mathcal{Y}_S \neq \mathcal{Y}_T$. The label spaces between the two tasks are different, e.g. documents need to be assigned different labels in the target task. In practice, this scenario usually occurs with scenario 4, as it is extremely rare for two different tasks to have different label spaces, but exactly the same conditional probability distributions.
4. $P(Y_S|X_S) \neq P(Y_T|X_T)$. The conditional probability distributions of the source and target tasks are different, e.g. source and target documents are unbalanced with regard to their classes. This scenario is quite common in practice and approaches such as over-sampling, under-sampling, or SMOTE are widely used

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





Learning Strategy	Related Areas	Source & Target Domains	Source Domain Labels	Target Domain Labels	Source & Target Tasks	Tasks
Inductive Transfer Learning	Multi-task Learning	The Same	Available	Available	Different but Related	Regression Classification
	Self-taught Learning	The Same	Unavailable	Available	Different but Related	Regression Classification
Unsupervised Transfer Learning		Different but Related	Unavailable	Unavailable	Different but Related	Clustering Dimensionality Reduction
Transductive Transfer Learning	Domain Adaptation, Sample Selection Bias & Co-variate Shift	Different but Related	Available	Unavailable	The Same	Regression Classification

	Inductive Transfer Learning	Transductive Transfer Learning	Unsupervised Transfer Learning
Instance-transfer	√	√	
Feature-representation-transfer	√	√	√
Parameter-transfer	√		
Relational-knowledge-transfer	√		

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

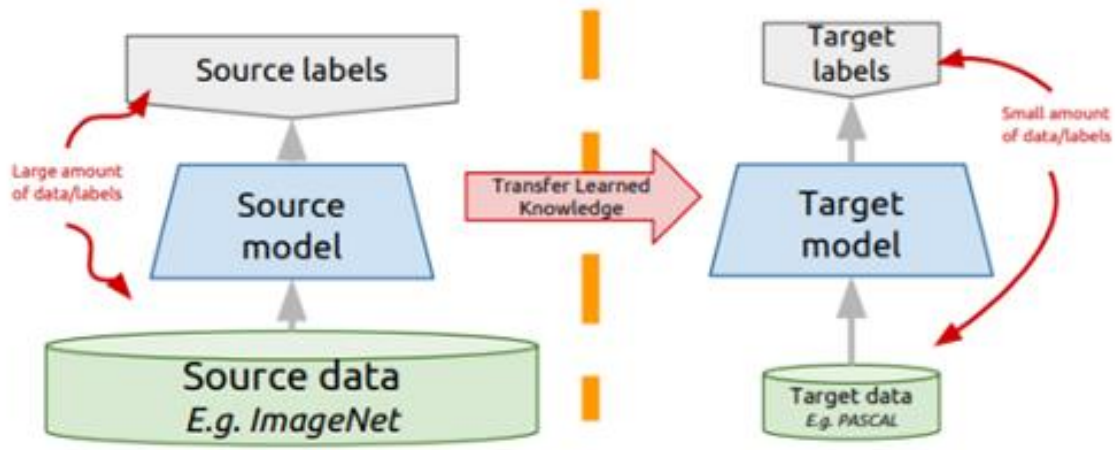
Transfer learning: idea

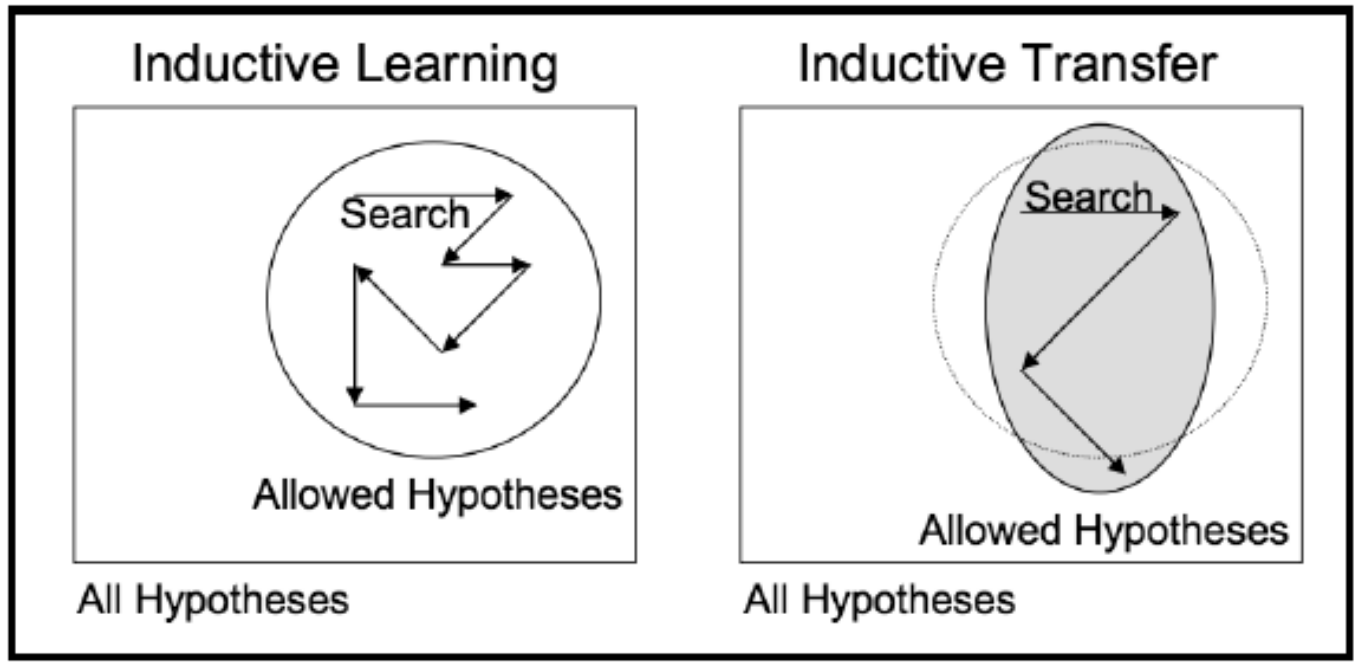
Instead of training a deep network from scratch for your task:

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

Variations:

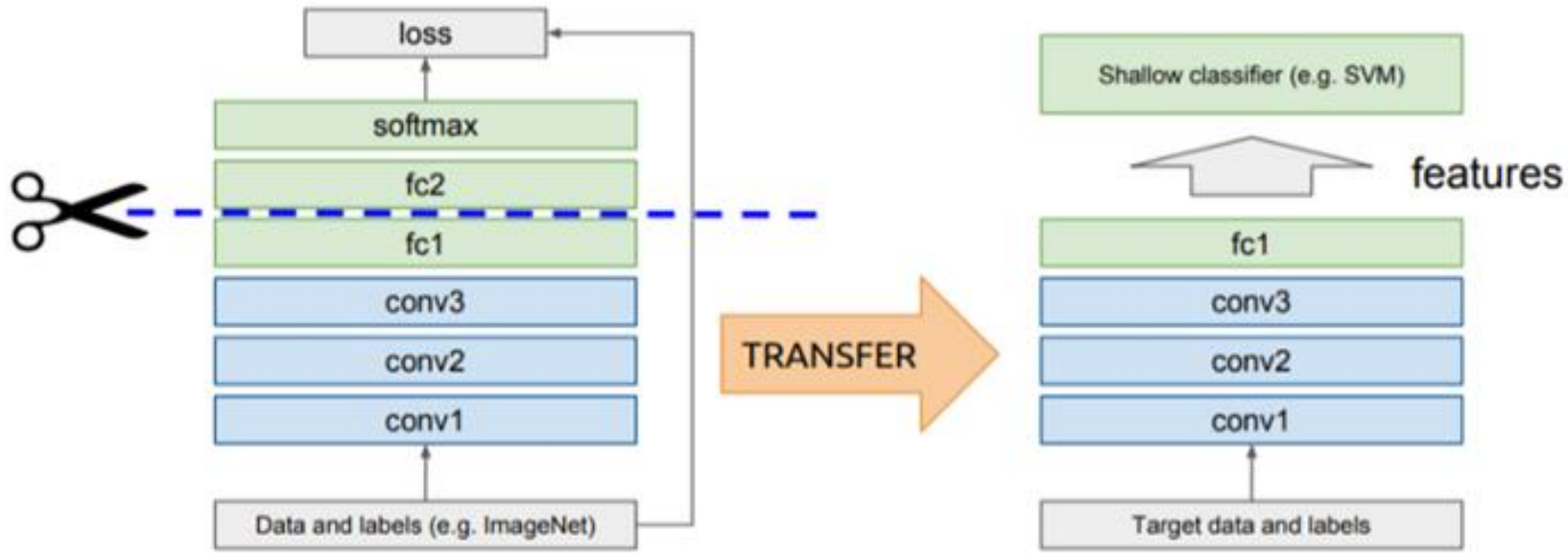
- Same domain, different task
- Different domain, same task




Inductive Learning
Search
Allowed Hypotheses
All Hypotheses
Inductive Transfer
Search
Allowed Hypotheses
All Hypotheses

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 $D_S = D_T$



Off-the-shelf features

Works surprisingly well in practice!

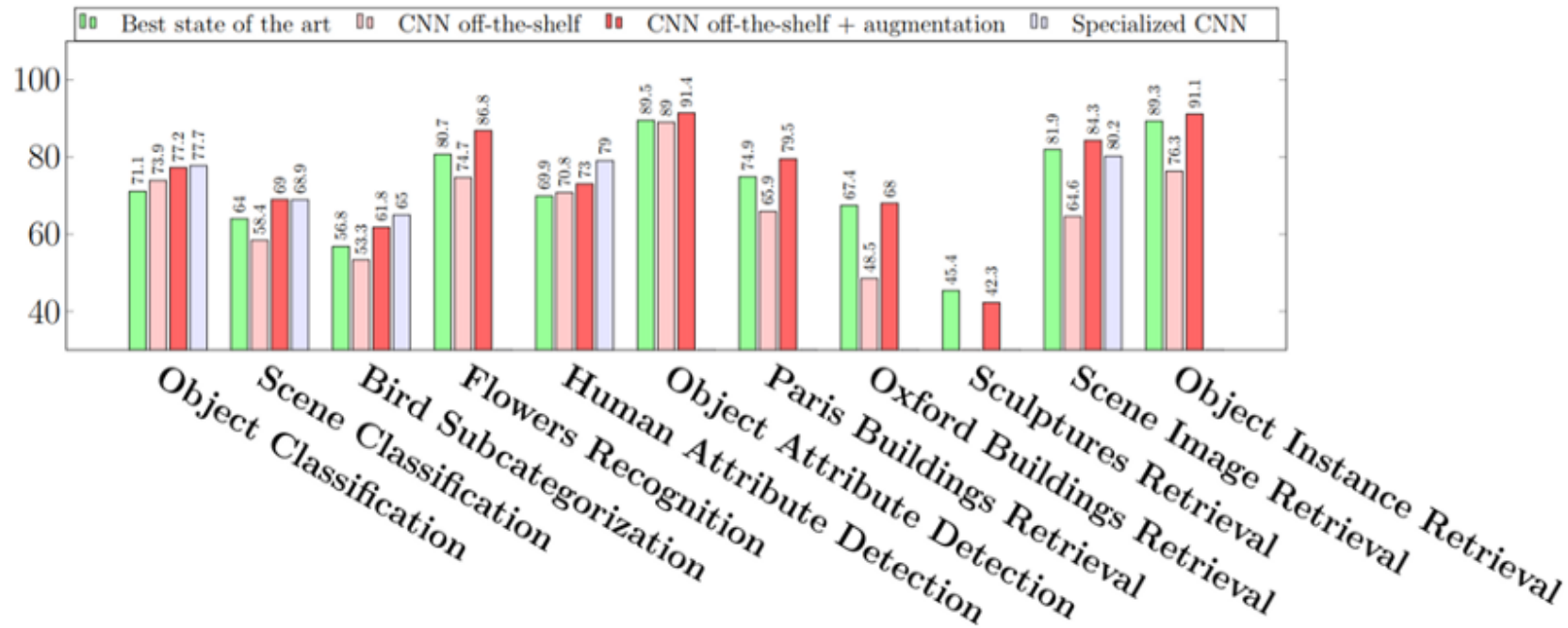
Surpassed or on par with state-of-the-art in several tasks in 2014

Image classification:

- PASCAL VOC 2007
- Oxford flowers
- CUB Bird dataset
- MIT indoors

Image retrieval:

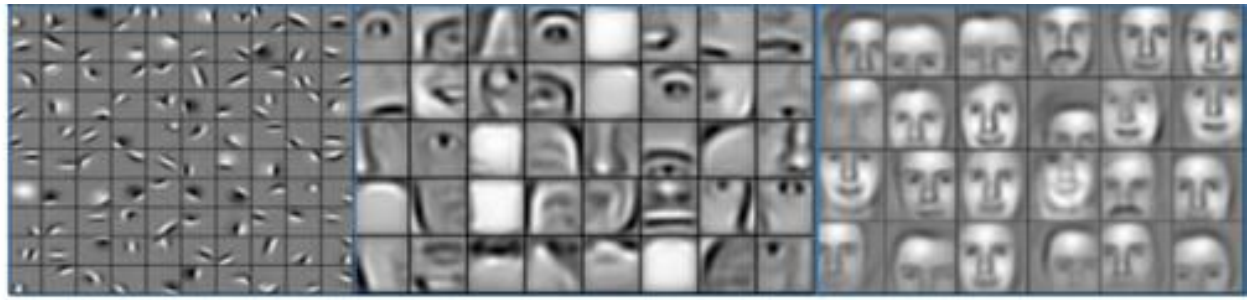
- Paris 6k
- Holidays
- UKBench

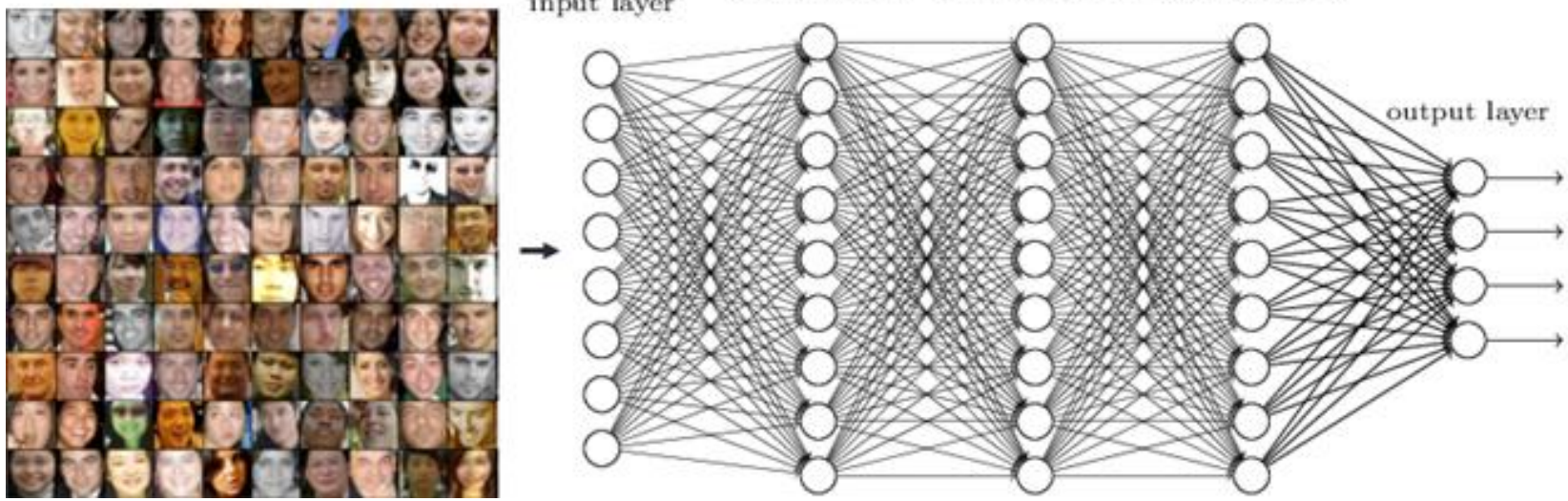


Method	mean Accuracy
HSV [27]	43.0
SIFT internal [27]	55.1
SIFT boundary [27]	32.0
HOG [27]	49.6
HSV+SIFTi+SIFTb+HOG(MKL) [27]	72.8
BOW(4000) [14]	65.5
SPM(4000) [14]	67.4
FLH(100) [14]	72.7
BiCos seg [7]	79.4
Dense HOG+Coding+Pooling[7] w/o seg	76.7
Seg+Dense HOG+Coding+Pooling[7]	80.7
CNN-SVM w/o seg	74.7
CNNaug-SVM w/o seg	**86.8**

Oxford 102 flowers dataset

Deep neural networks learn hierarchical feature representations





Fine-tuning: supervised domain adaptation

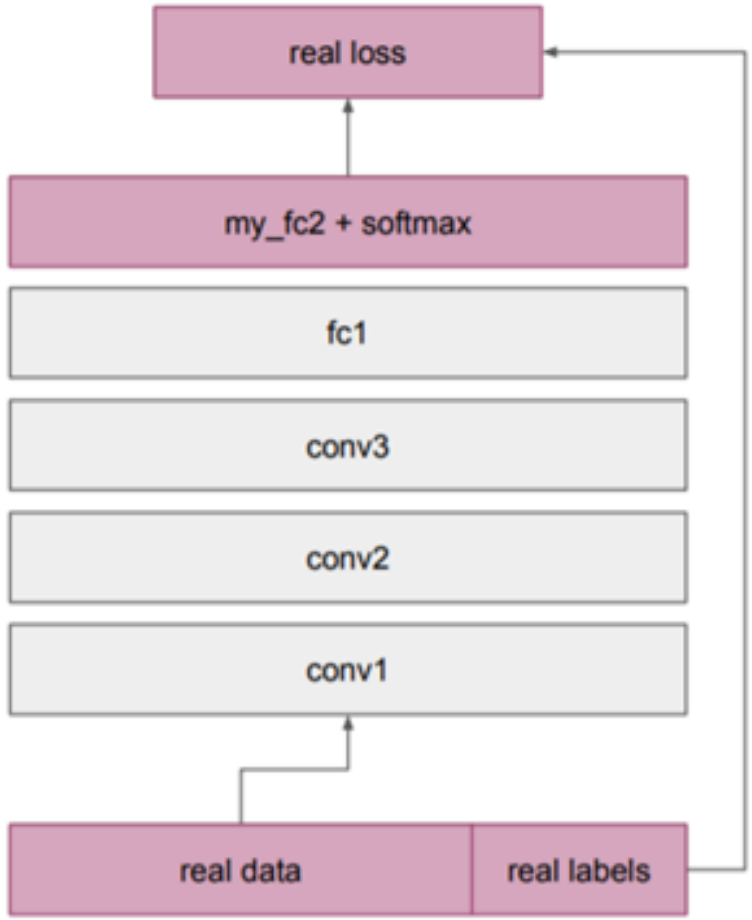
Train deep net on “nearby” task for which it is easy to get labels using standard backprop

- 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



Freeze or fine-tune?

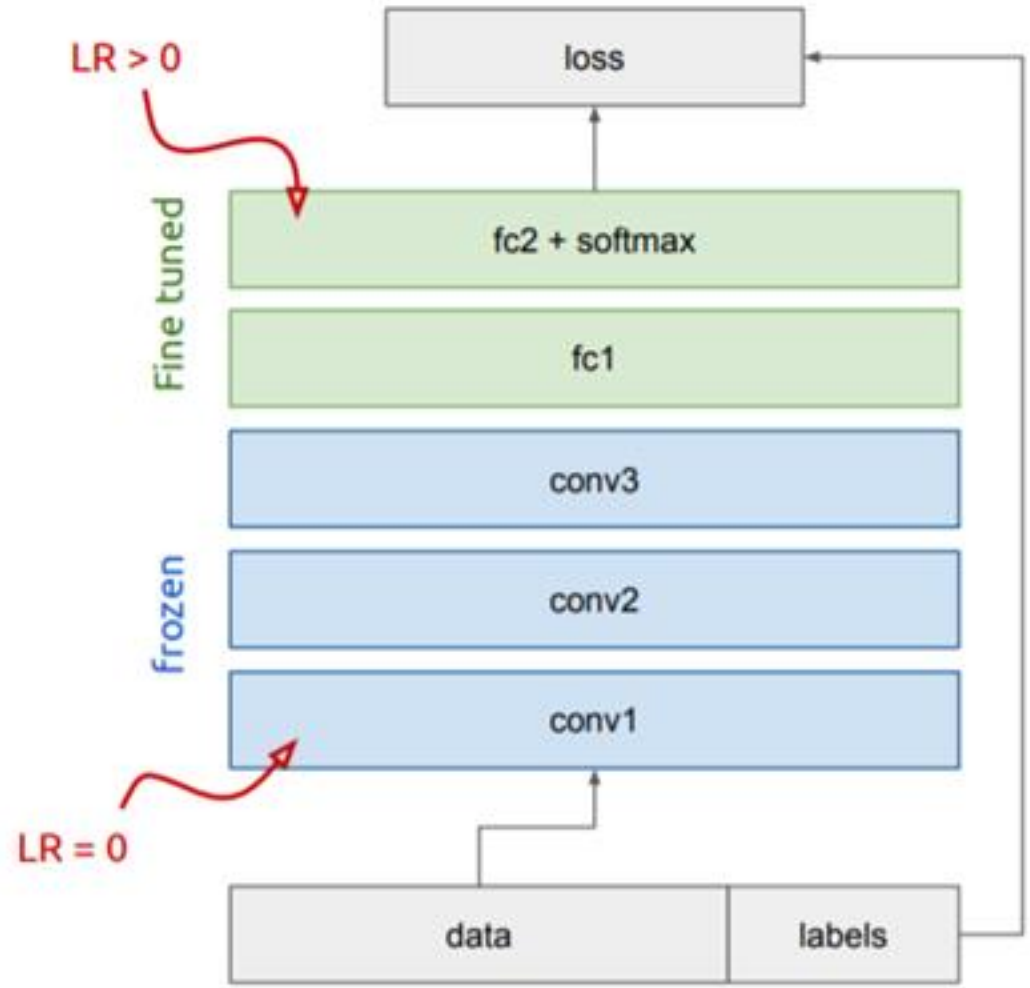
Bottom *n* layers can be frozen or fine tuned.

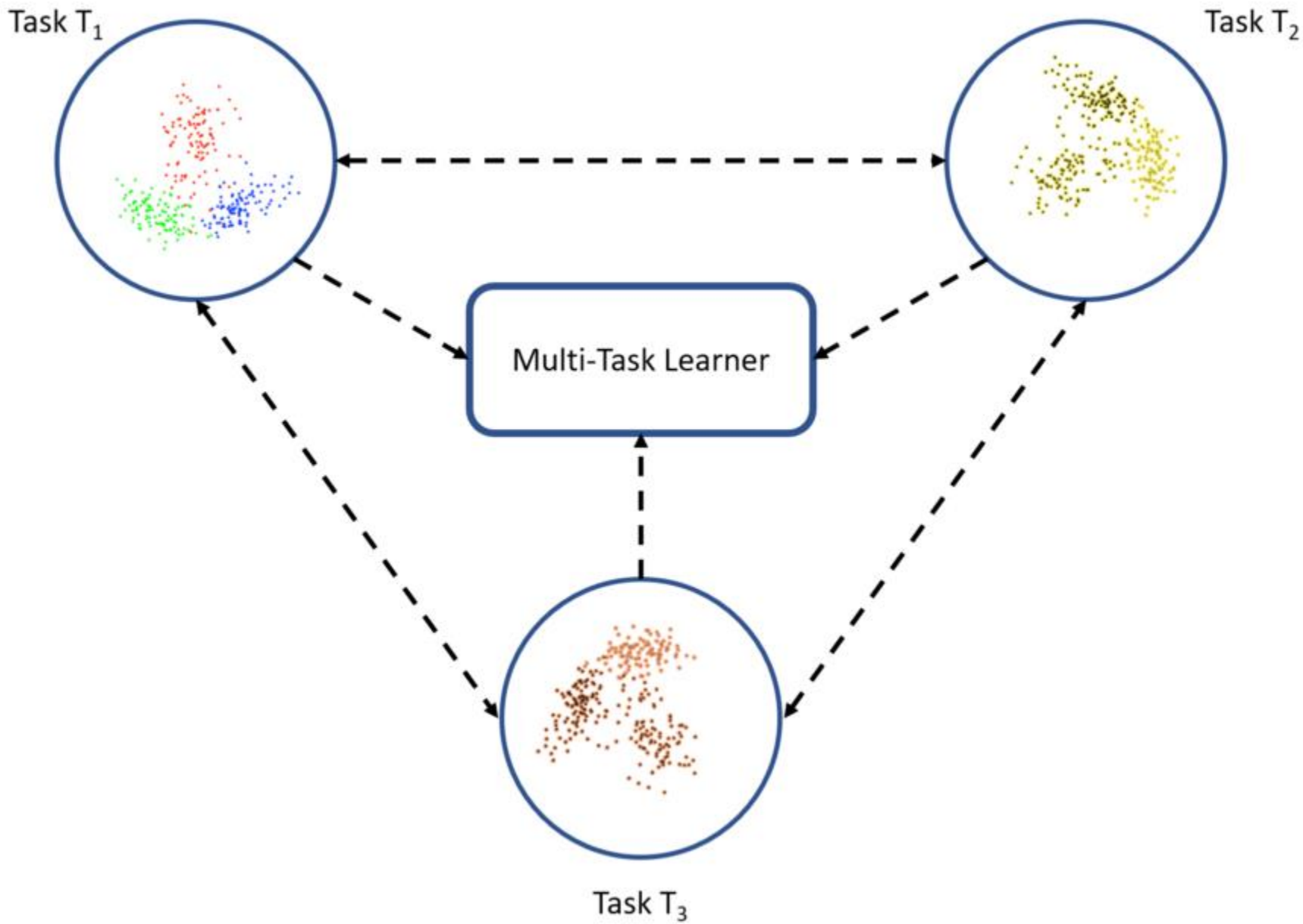
- **Frozen**: not updated during backprop
- **Fine-tuned**: updated during backprop

Which to do depends on target task:

- **Freeze**: target task labels are scarce, and we want to avoid overfitting
- **Fine-tune**: target task labels are more plentiful

In general, we can set learning rates to be different for each layer to find a tradeoff between freezing and fine tuning




Task T_1
Task T_2
Multi-Task Learner
Task T_3

How transferable are features?

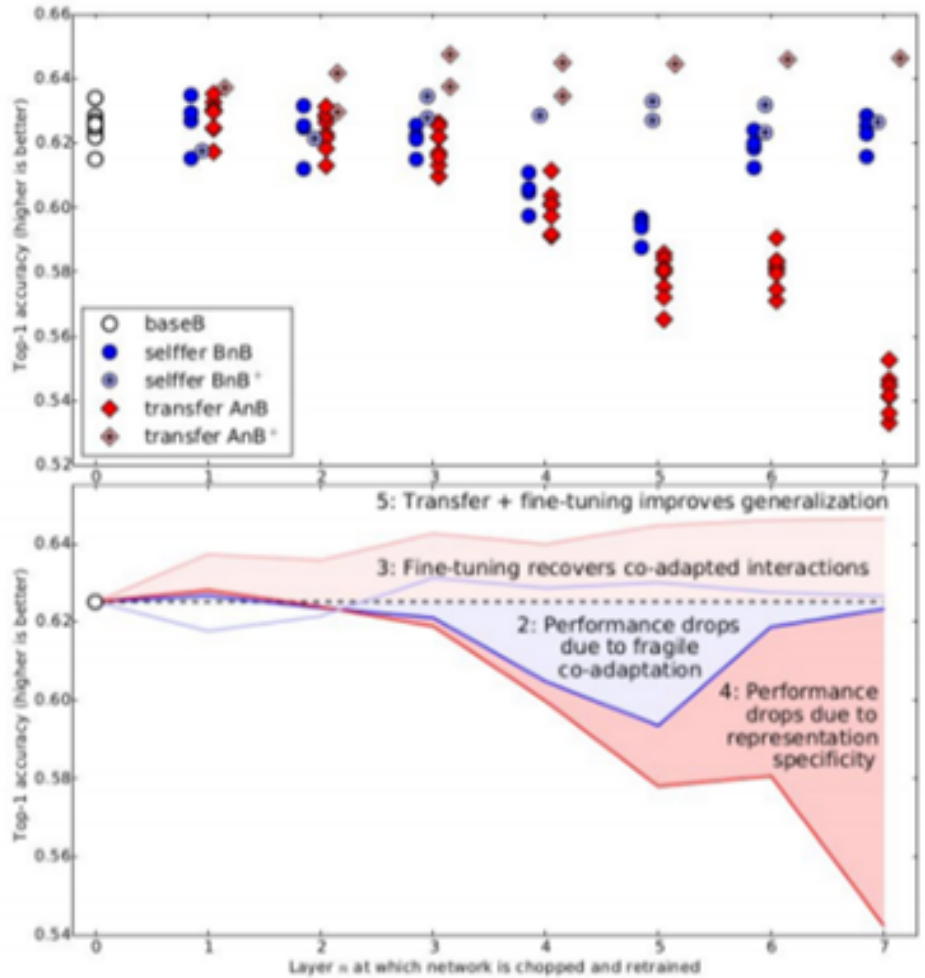
Transferability is negatively affected by two distinct issues:

- 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