Advanced Deep Learning

Representation Learning

U Kang
Seoul National University

In This Lecture

- Greedy Layer-Wise Unsupervised Pretraining
- Transfer Learning and Domain Adaptation
- Semi-Supervised Disentangling of Causal Factors
- Distributed Representation
- Exponential Gains from Depth
- Providing Clues to Discover Underlying Causes

Outline

➡ ☐ **Overview**

☐ Greedy Layer-Wise Unsupervised Pretraining

☐ Transfer Learning and Domain Adaptation

☐ Semi-Supervised Disentangling of Causal Factors

☐ Distributed Representation

☐ Exponential Gains from Depth

☐ Providing Clues to Discover Underlying Causes

Overview

- Machine learning:
  - Predictive Learning
  - Representation Learning

- What makes one representation better than another?
- Example:

  Division of CCX by VI ?

Overview

- Example:

  Division of CCX by VI ? $\Rightarrow$ Division of 210 by 6

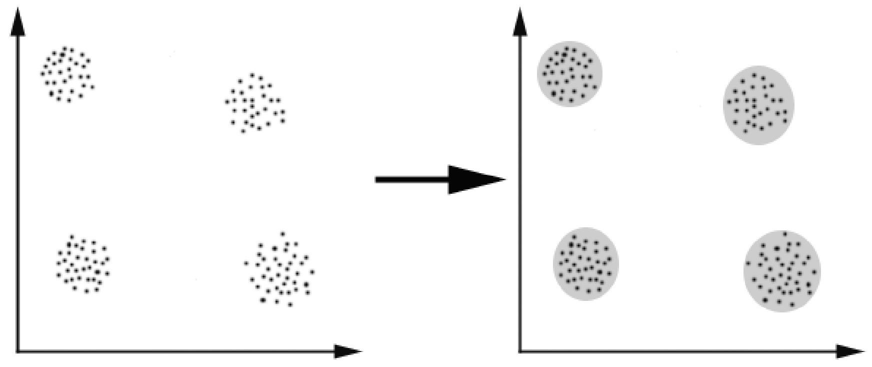
$$\begin{array}{r|l} 210 & 6 \\ \cline{2-2} \underline{-18} & 35 \\ 30 & \end{array}$$

- Processing tasks can be very easy/difficult depending on how the information is represented.

Representation Learning (RL)

- Most widely used unsupervised learning techniques and the representations they produce:

- **Clustering:** maps data points to a discrete set where the only meaningful operation is equality.



Representation Learning (RL)

- **Nonlinear dimensionality reduction** algorithms:

map data points to a low-dimensional space where Euclidean distance is meaningful.

- **Linear dimensionality reduction** algorithms like PCA: map data points to a low-dimensional space where Euclidean distance, linear combination, and dot products are all meaningful.

Representation Learning (RL)

- A good representation is one that makes a subsequent learning task easier
- Trade-off between preserving as much information as possible and attaining nice properties
- RL provides one way to perform unsupervised learning. We often have

  - Large amounts of unlabeled data

  - Relatively little labeled training data

Representation Learning (RL)

- Why RL is interesting:
  - Learn good representations for the unlabeled data
  - Use these representations to solve the supervised learning task.
- Unsupervised deep learning algorithm learns a representation as a side effect.

Outline

- ✓ Overview
- ➡ ☐ **Greedy Layer-Wise Unsupervised Pretraining**
- ☐ Transfer Learning and Domain Adaptation
- ☐ Semi-Supervised Disentangling of Causal Factors
- ☐ Distributed Representation
- ☐ Exponential Gains from Depth
- ☐ Providing Clues to Discover Underlying Causes

Greedy Layer-Wise Unsupervised Pretraining (GLWUP)

- Unsupervised learning: allowed researchers for the 1st time to train a deep supervised network without requiring architectural specializations (like convolution or recurrence).
- This procedure is called Unsupervised Pretraining; or GLWUP.

Greedy Layer-Wise Unsupervised Pretraining (GLWUP)

- GLWUP: canonical example of how a representation learned for one task can be useful for another task
  - Example: Trying to capture the shape of the input distribution (unsupervised task) useful for supervised learning with the same input domain.

- Used to sidestep the difficulty of jointly training the layers of a deep neural net for a supervised task.

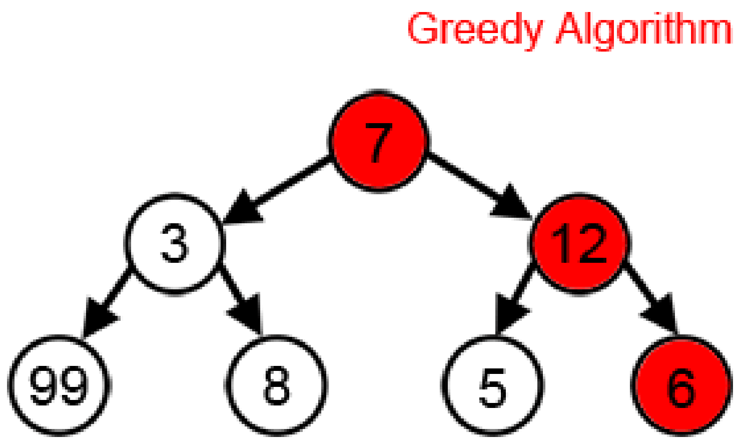
GLWUP: Explanation of the terms

- **Greedy** Algorithm:
- Break a problem into many components
- Solve the optimal version of each component in isolation

- It optimizes each piece of the solution independently, one piece at a time, rather than jointly optimizing all pieces.

GLWUP: Explanation of the terms

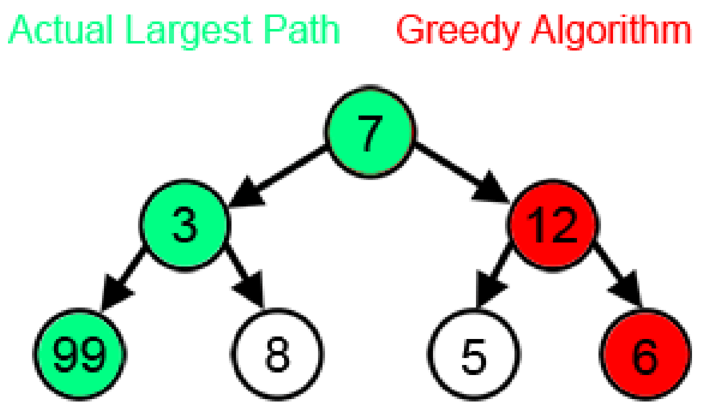
- **Greedy** Algorithm:
  - Example:



  - Problem: not guaranteed to yield an optimal complete solution

GLWUP: Explanation of the terms

- **Greedy** Algorithm:



  - But computationally much cheaper than algorithms that solve for the best joint solution
  - The quality of a greedy solution is often acceptable, if not optimal.

GLWUP: Explanation of the terms

- **Layer-Wise:** these independent pieces are the lay er of the network.

- **Greedy Layer-Wise** pretraining:
  - Proceeds one layer at a time
  - Training the k-th layer while keeping the previous ones fixed
  - The lower layers (trained first) are not adapted after the upper layers are introduced.

GLWUP: Explanation of the terms

- **Unsupervised:** Each layer is trained with an unsupervised representation learning algorithm

- Differences between unsupervised and supervised?
- Example:
- There is a bunch of different fruits:
- Supervised: Based on its color/shape weight, is that fruit an apple? > Boolean
- Unsupervised: How the different fruits can be clustered inside your grocery store?

Unsupervised vs Supervised

- **Unsupervised:**
  - Trying to "understand" the data
  - Data is unlabeled or value unknown
  - Goal: try to find correlations without any external inputs other than the raw data
  - Example: clustering

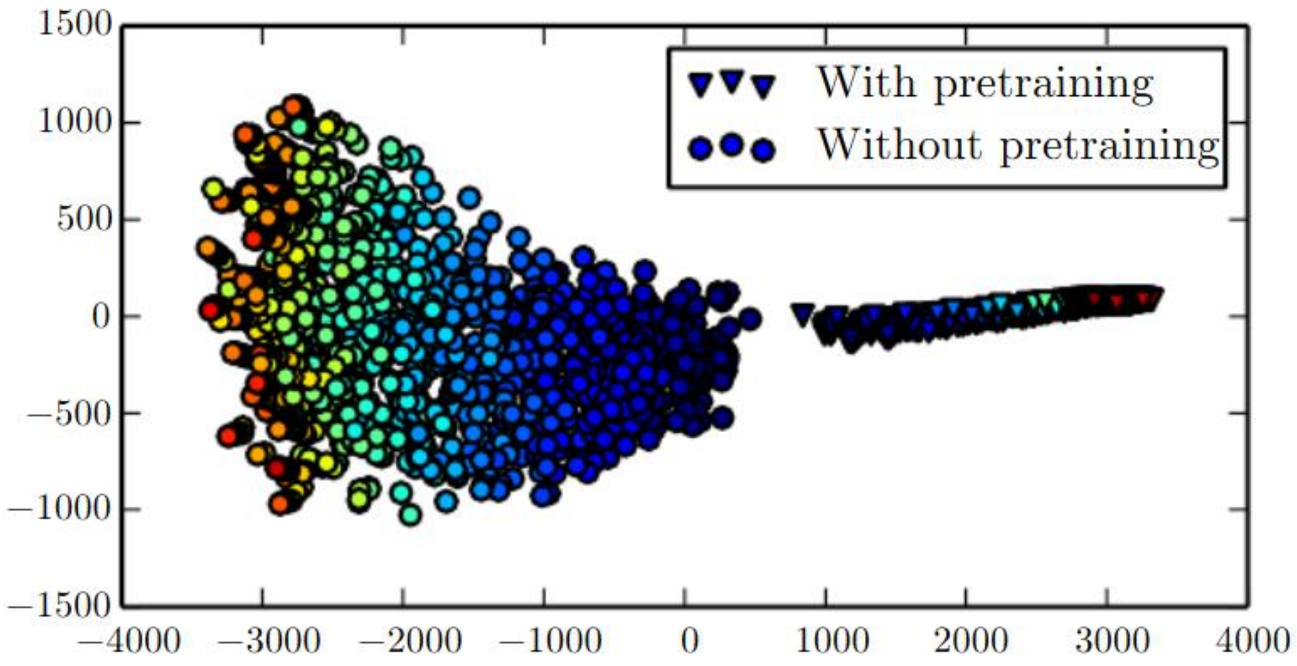
- **Supervised:**
  - Data is labelled with a class or value
  - Goal: predict class or value label
  - Example: Neural Network, SVM, Decisions Trees, Bayesian Classi fiers

GLWUP: Explanation of the terms

- **Pretraining:**
  - Supposed to be only a first step before a joint training algorithm is applied, to fine-tune all the layers together.
  - It is common that the word “pretraining” refers to the two-phase protocol: pretraining phase and supervised learning phase.

GLWUP: Explanation of the terms

- **Pretraining:**
  - It can be viewed as a regularizer and a form of parame ter initialization.



(GLWUP) Greedy Layer-Wise Unsupervised Pretraining

- GLWUP can be used as initialization for other unsupervised learning algorithms, such as deep autoencoders and probabilistic models with many layers of latent variables.

GLWUP: Algorithm

- **GLWUP** relies on a single-layer representation learning algorithm such as:
  - an RBM (**R**estricted **B**oltzmann **M**achine)
  - a single-layer autoencoder
  - a sparse coding model
  - another model that learns latent represen-tations

GLWUP: Algorithm

- **GLWUP Protocol:**

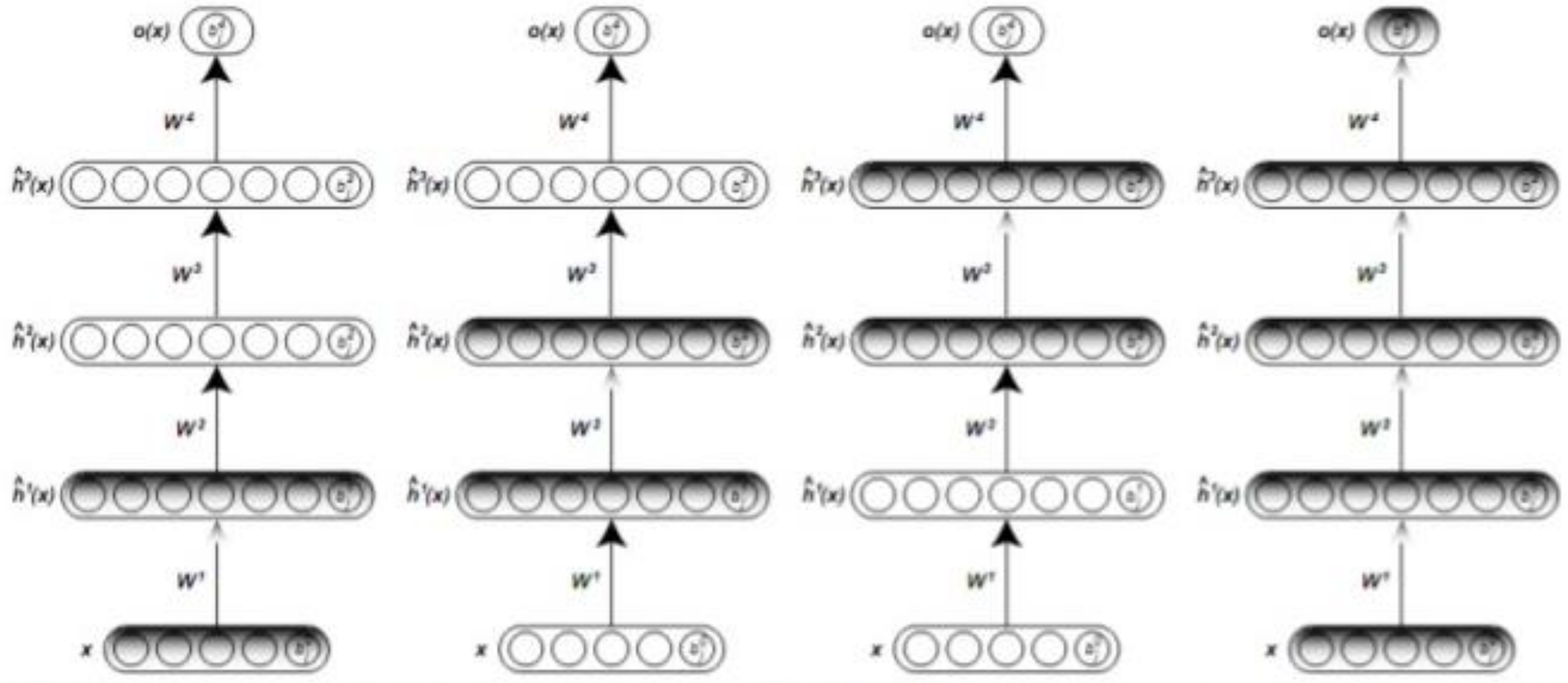
```
$f \leftarrow$ Identity function
$\tilde{\boldsymbol{X}} = \boldsymbol{X}$
for $k = 1, \ldots, m$ do
    $f^{(k)} = \mathcal{L}(\tilde{\boldsymbol{X}})$
    $f \leftarrow f^{(k)} \circ f$
    $\tilde{\boldsymbol{X}} \leftarrow f^{(k)}(\tilde{\boldsymbol{X}})$
end for
if fine-tuning then
    $f \leftarrow \mathcal{T}(f, \boldsymbol{X}, \boldsymbol{Y})$
end if
Return $f$
```

- Unsupervised feature learning algorithm L, which takes a training set of examples and returns an encoder or feature function f.

- The raw input data is X, with one row per example, and $f^{(1)}(X)$ is the output of the first stage encoder on X.
- Y: associated targets

GLWUP: Algorithm

- **Basic pictorial representation** of the training procedure for deep learning architecture:



(a) First hidden layer pre-training

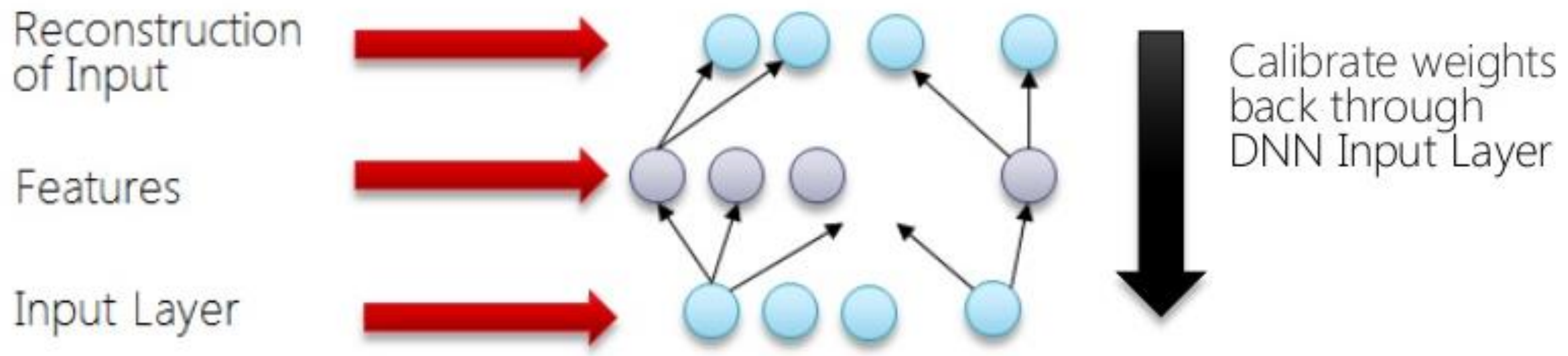
(b) Second hidden layer pre-training

(c) Third hidden layer pre-training

(d) Fine-tuning of whole network

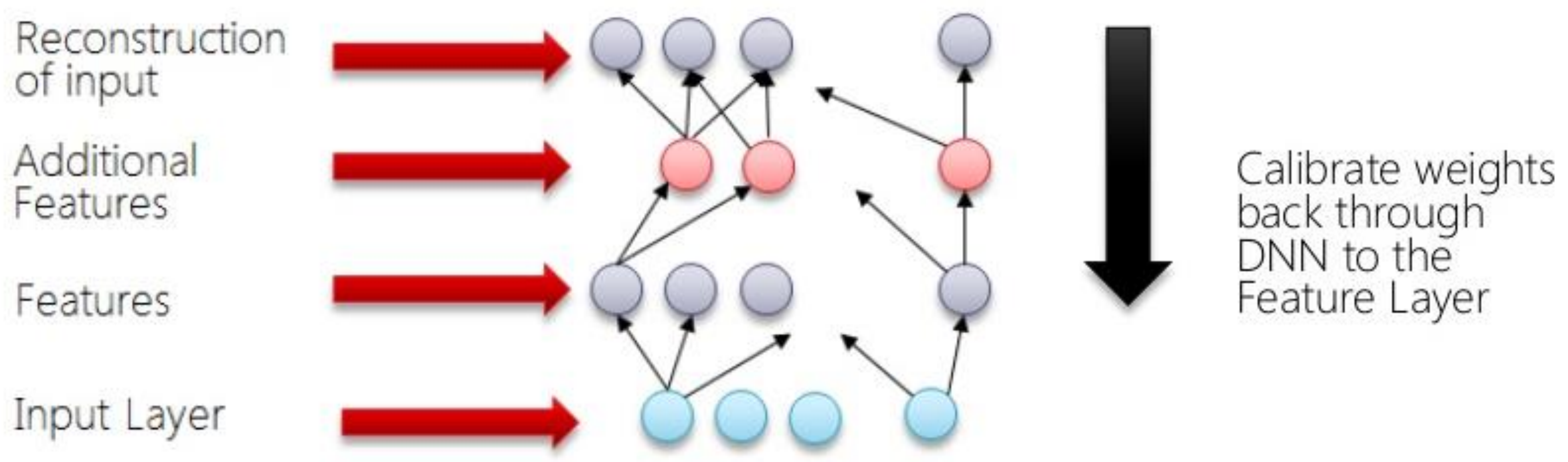
GLWUP: Algorithm

- **Step 1:** Train the first hidden layer of the DNN and reconstruct the input based upon the hidden layers weight



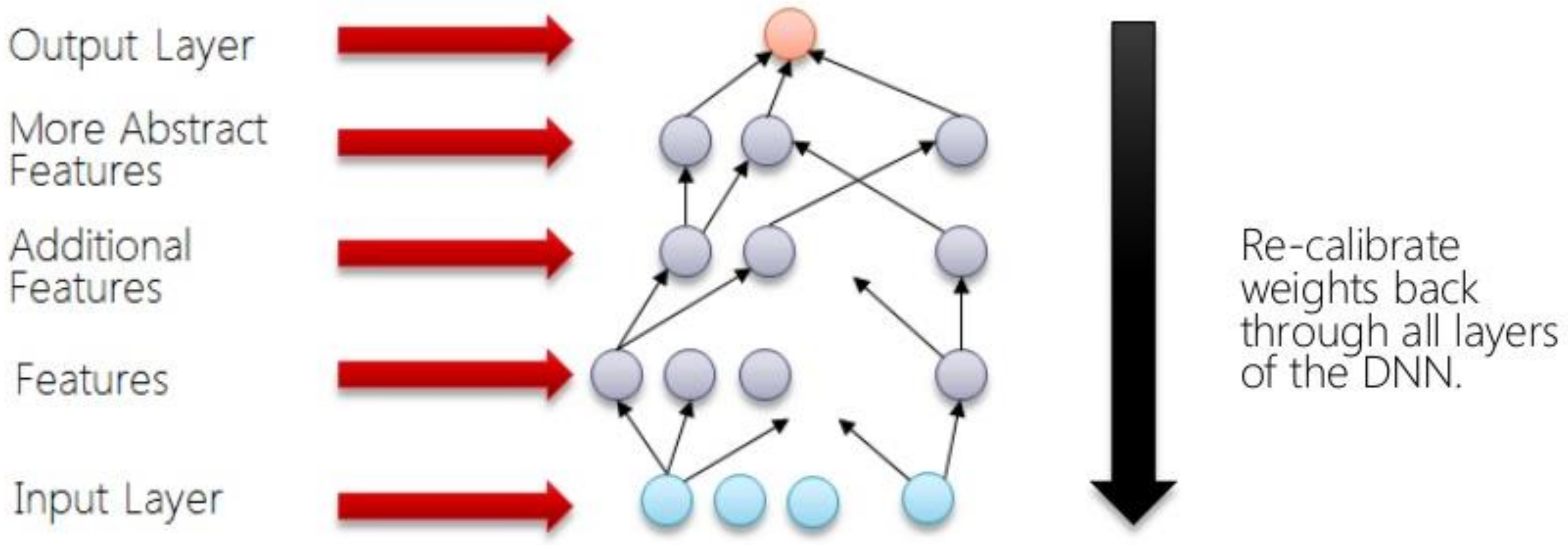
GLWUP: Algorithm

- **Step 2:** We now take the next hidden layer of "Additional Features" and train the layer using the inputs from the "Features" and reconstruct the Feature layer from the inputs.

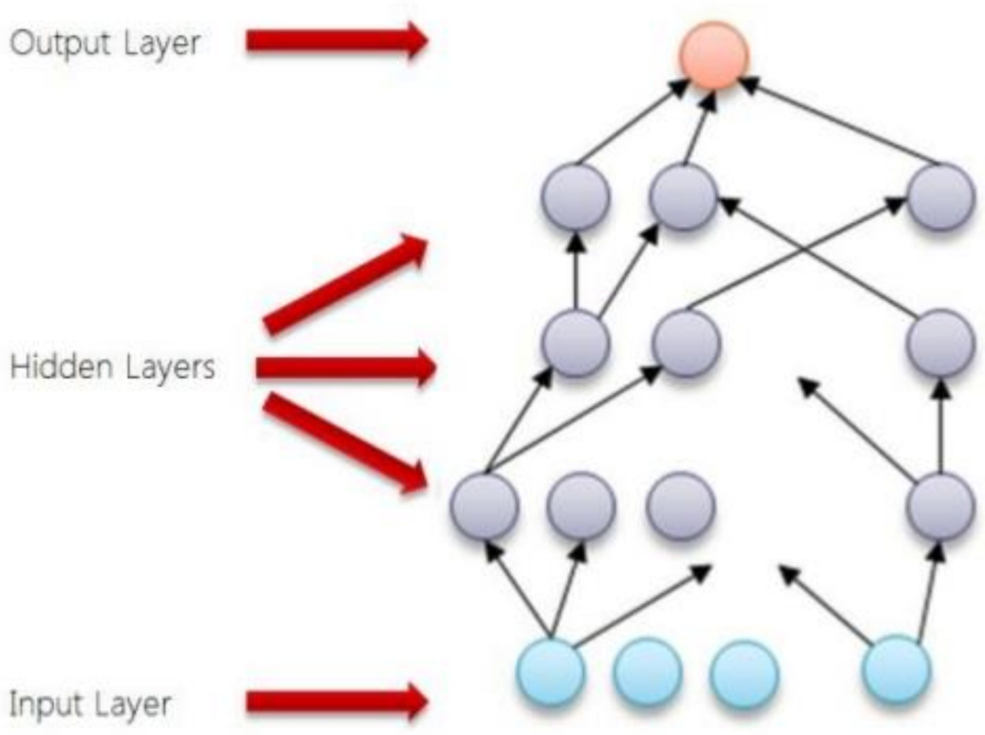


GLWUP: Algorithm

- **Step 3:** We continue to go through each hidden layer as described in step 2 until we reach the final output layer.



GLWUP: Algorithm



- Each layer is pretrained using unsupervised training, taking the output of the previous layer and producing as output a new representation of the data.

GLWUP Benefits

- Allows abstraction to develop naturally from one layer to another
- Help the network initialize with good parameters
- Refine the features (intermediate layers) so they become more relevant for the task

When and Why does Unsupervised Pretraining work?

- Unsupervised pretraining is sometimes helpful but often harmful.
- Two different ideas for UP:
  - 1st idea: the choice of initial parameters for a deep neural network can have a significant regularizing effect on the model (can improve optimization)
  - 2nd idea: Learning about the input distribution can help with learning about the mapping from inputs to outputs
- Many complicated interactions that are not entirely understood.

When and Why does Unsupervised Pretraining work?

- **1st idea** (the least understood):

- Initializing the model in a location that would cause it to approach one local minimum rather than another.

- Local minima are no longer considererd to be a serious problem for neural network optimization (neural network training procedures usually do not arrive at a critical point)

When and Why does Unsupervised Pretraining work?

- **2nd idea**:

- Some features that are useful for the unsupervised task may also be useful for the supervised learning task.

- Not yet understood at a mathematical, theoretical level, not always possible to predict which tasks will benefit from unsupervised learning.

When and Why does Unsupervised Pretraining work?

- **Point of view of UP as a regularizer**

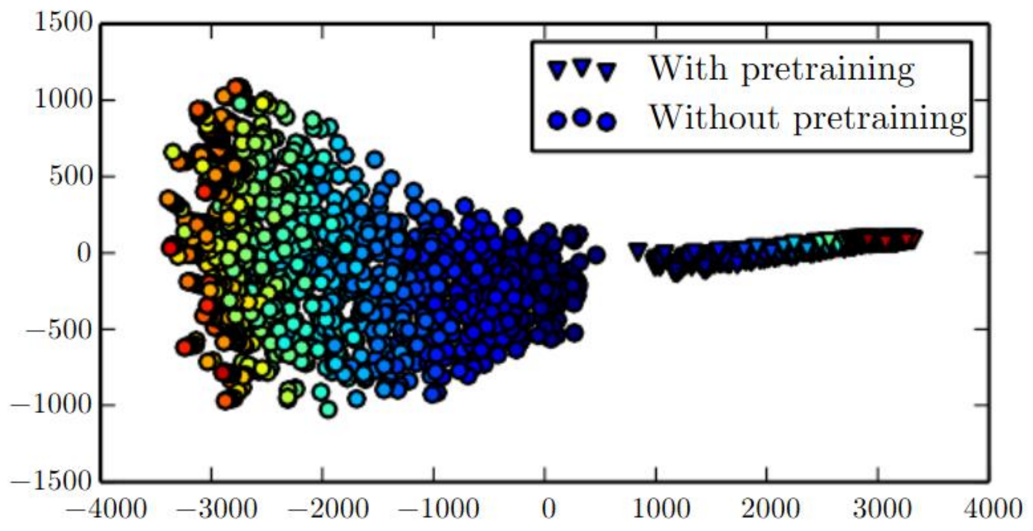
- Most helpful when the number of labeled examples is very small.

- Most useful when the function to be learned is extremely complicated.

When and Why does Unsupervised Pretraining work?

- **Where UP is known to cause an improvement:**

Reducing test set error: may be explained in terms of unsupervised pretraining taking the parameters into a region that would otherwise be inaccessible.



When and Why does Unsupervised Pretraining work?

- **Disadvantages:** UP operates with 2 separate training phases:

- Pre-training with unsupervised data (e.g.: RBMs)

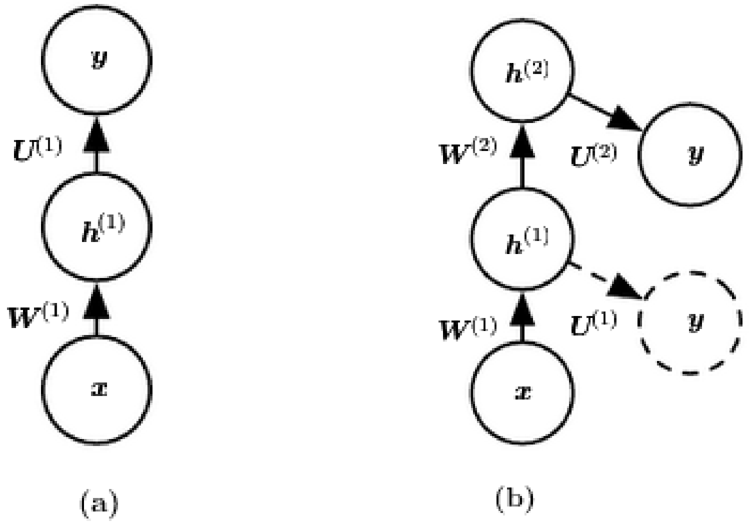
- Fine-tuning parameters with supervised data

- UP does not offer a clear way to adjust the strength of the regularization arising from the unsupervised stage.

- Each phase has its own hyperparameters.

Greedy Layer-Wise Supervised Pretraining

- Each added hidden layer is pretrained as part of a shallow supervised MLP, taking as input the output of the previously trained hidden layer.



- Very common approach for transfer learning

Outline

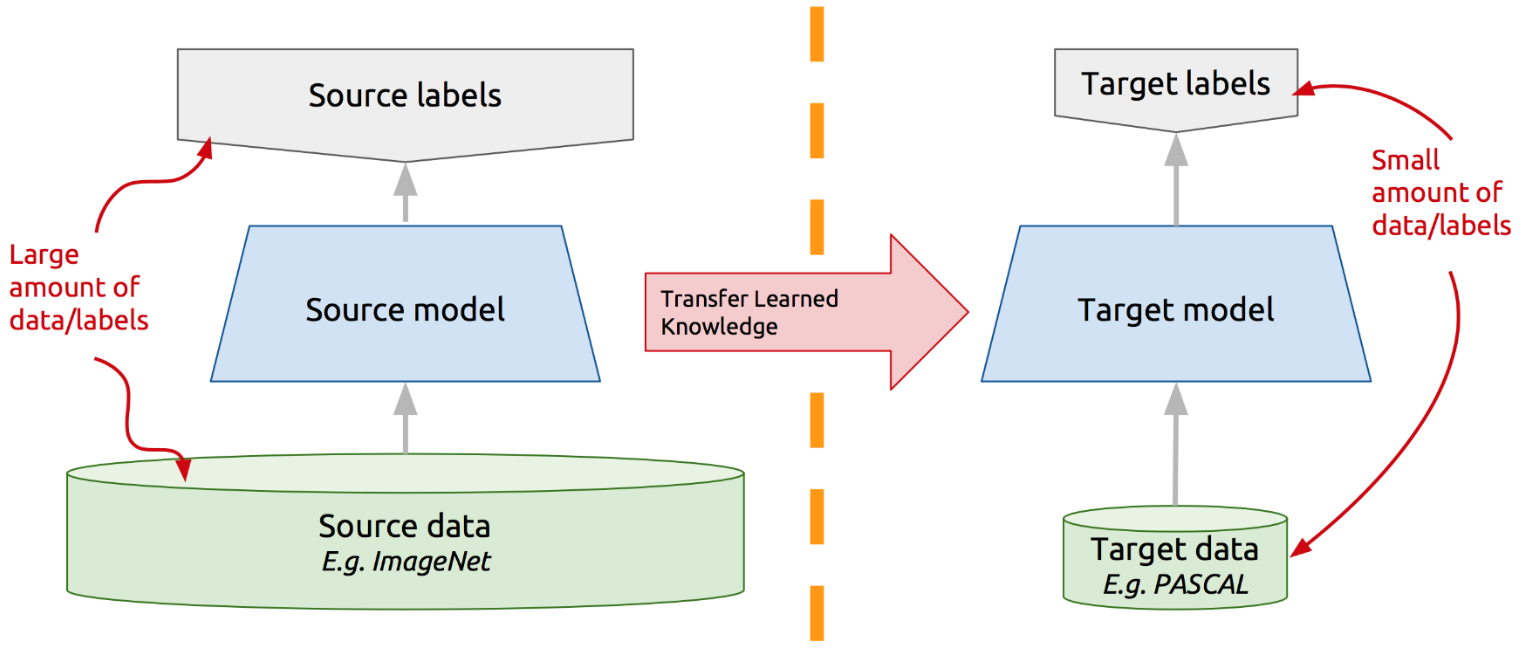
- ✓ Overview
- ✓ Greedy Layer-Wise Unsupervised Pretraining
- ☐ **Transfer Learning and Domain Adaptation**
- ☐ Semi-Supervised Disentangling of Causal Factors
- ☐ Distributed Representation
- ☐ Exponential Gains from Depth
- ☐ Providing Clues to Discover Underlying Causes

Transfer Learning

- Main idea
  - What has been learned in one setting (distribution P_1) is exploited to improve generalization in another setting (distribution P_2).
  - Assumption: many factors that explain the variations in P_1 are relevant to the variations that need to be captured for learning P_2.

Transfer Learning

- Example
  - Source task has a large amount of data.
  - Target task has a small amount of data.



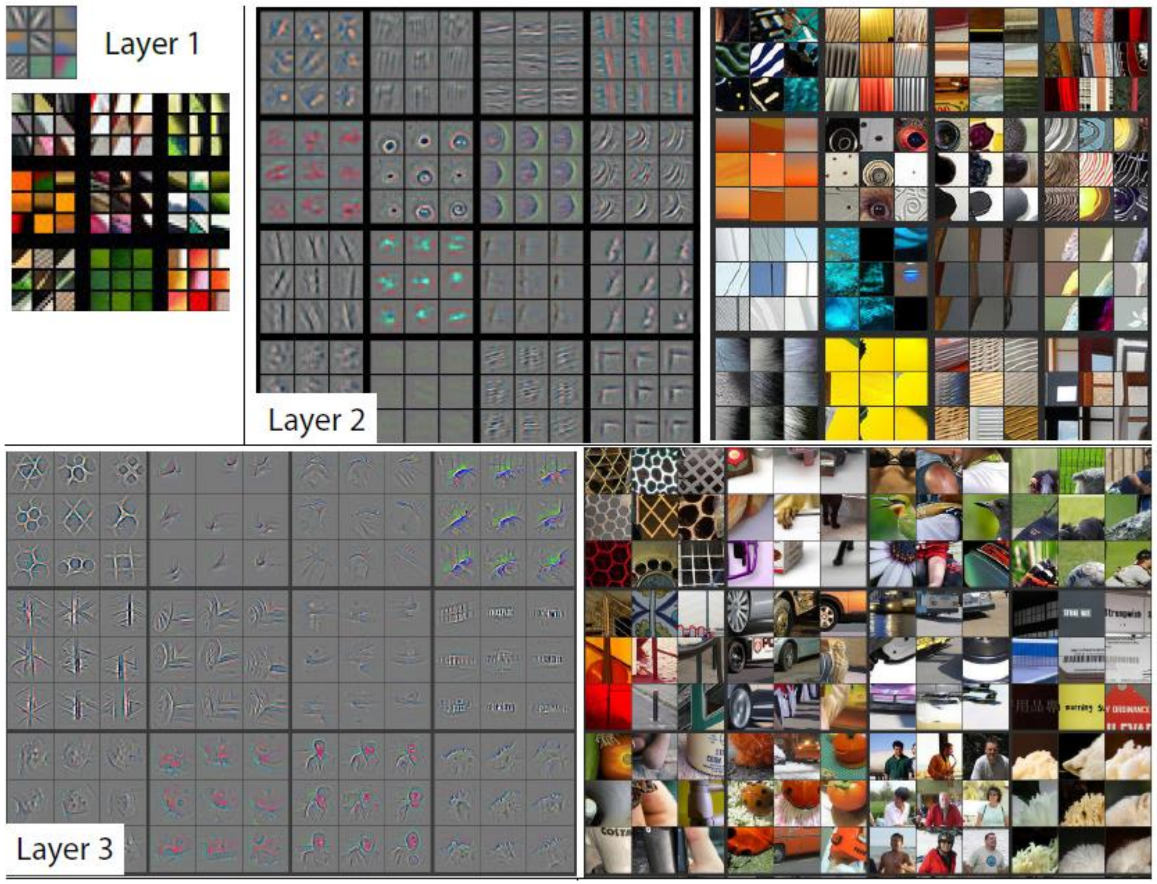
"Deep Learning for Computer Vision." *Summer seminar UPC TelecomBCN*

How transferable are features?

- Motivation
  - The core idea of transfer learning is that same representation may be useful in both source setting and target setting.
  - How can we find common representation?
  - In deep learning model
    - Lower layers: extract more general representation (e.g. edges, visual shapes)
    - Higher layers: extract more task specific representation (e.g. effects of geometric, lightning)

How transferable are features?

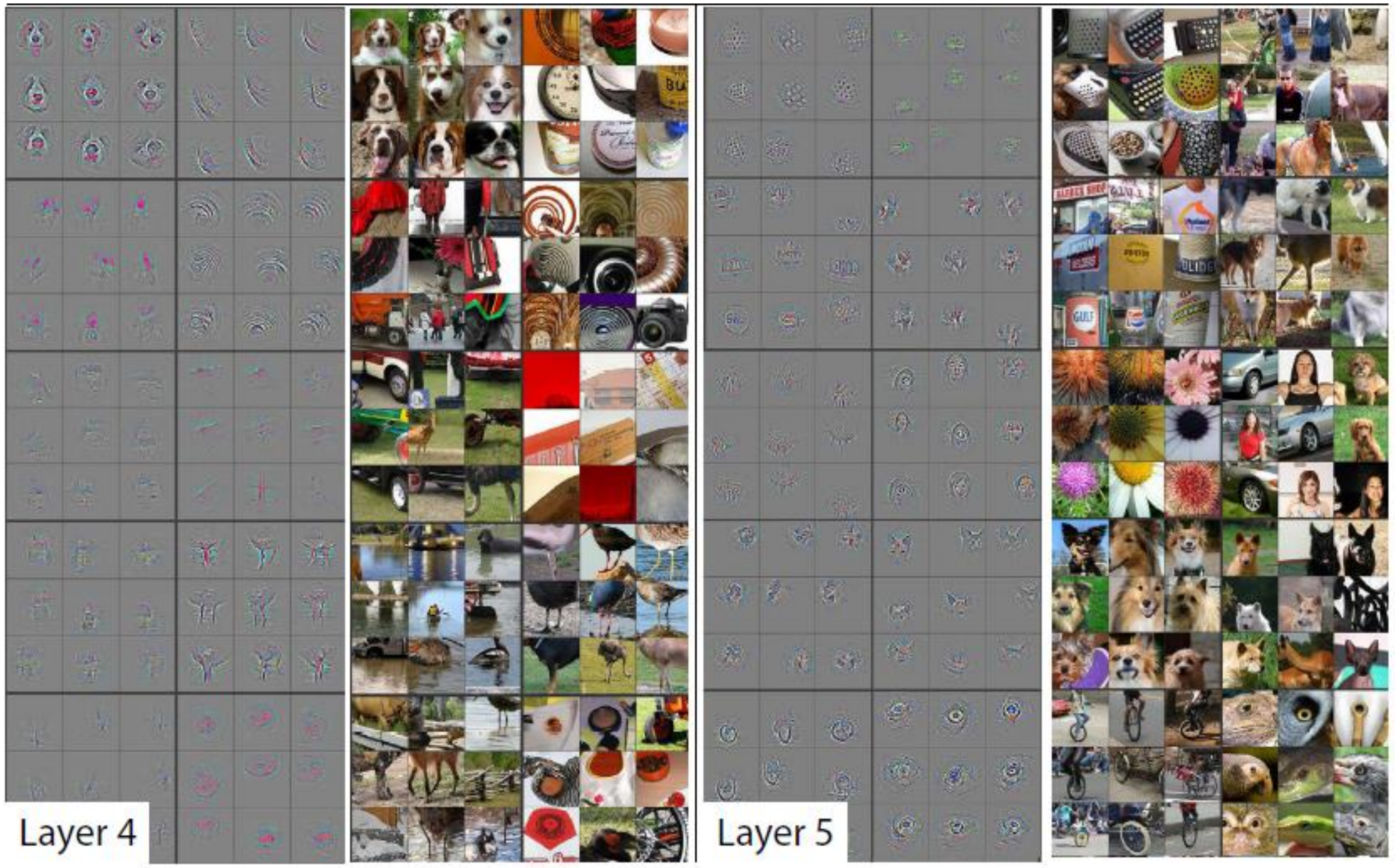
- Visualization of a fully trained CNN model



"Visualizing and understanding convolutional networks." ECCV 2014

How transferable are features?

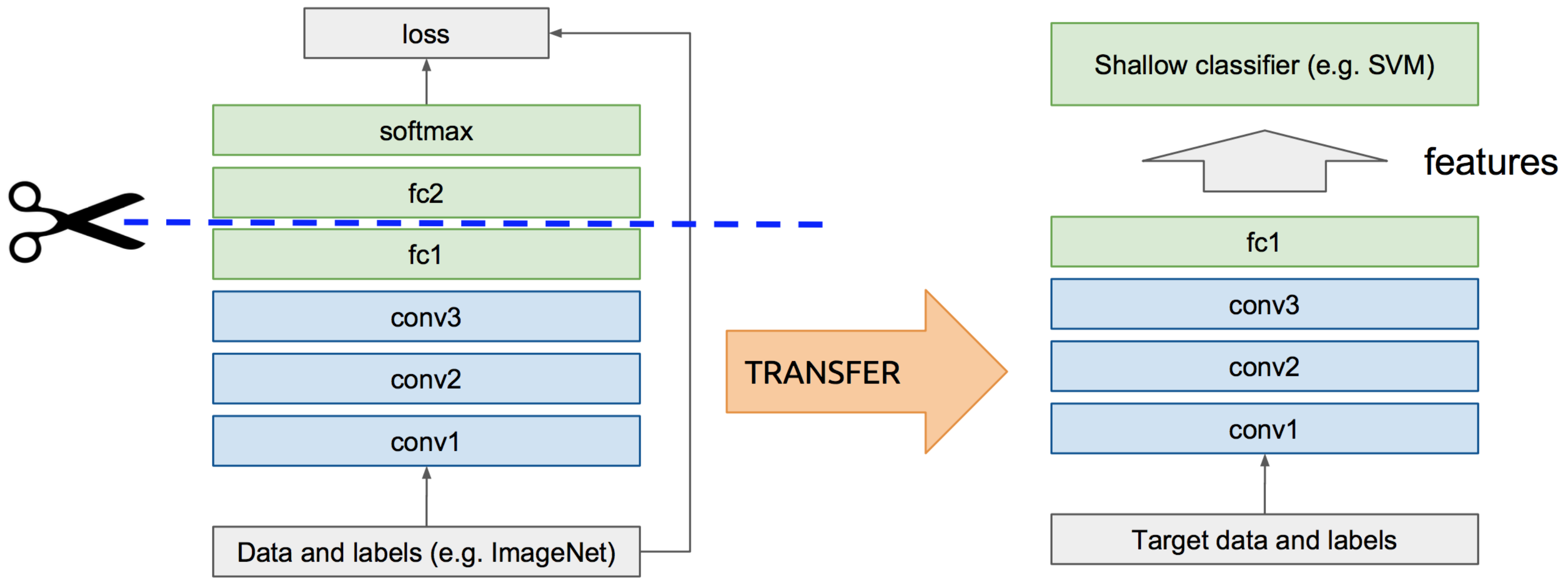
- Visualization of a fully trained CNN model



"Visualizing and understanding convolutional networks." ECCV 2014

Off-the-shelf

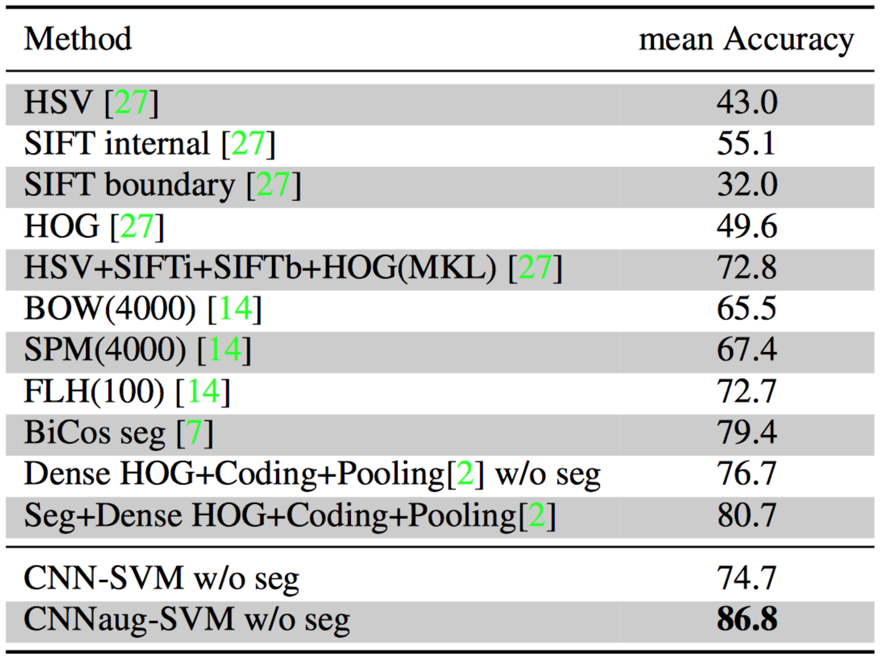
- Idea
  - Idea is that transfer some layers of a network trained on a different task to target model.



"Deep Learning for Computer Vision." *Summer seminar UPC TelecomBCN*

Off-the-shelf

- Result
  - Off-the-shelf outperforms other methods.



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[2] w/o seg	76.7
Seg+Dense HOG+Coding+Pooling[2]	80.7
CNN-SVM w/o seg	74.7
CNNaug-SVM w/o seg	**86.8**

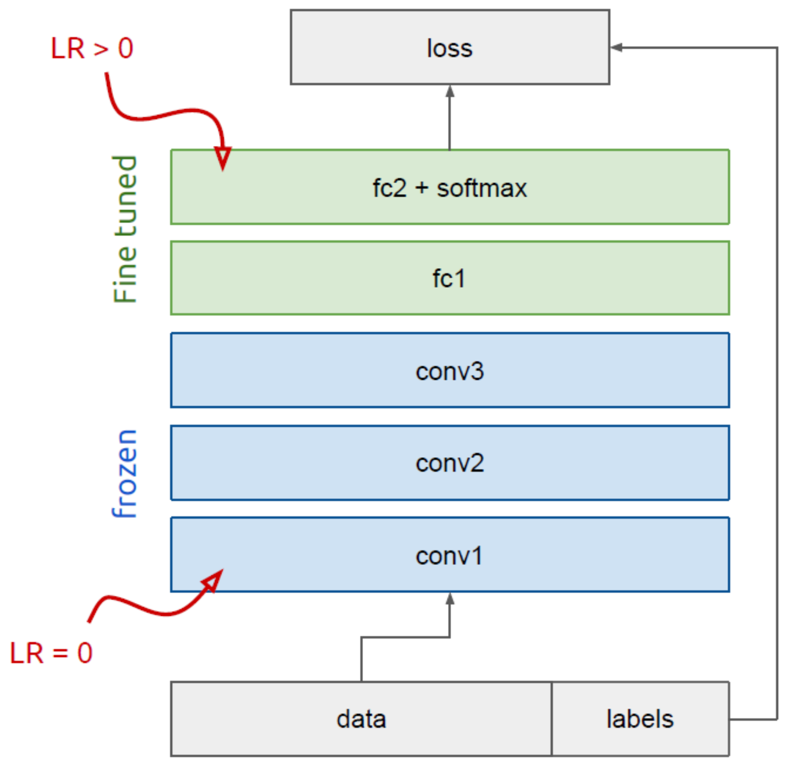
"CNN features off-the-shelf: an astounding baseline for recognition." *IEEE CVPR* 2014

Fine-tuning

- Idea
  - Cut-off the top layers and replace with target task dependent layers. (off-the-shelf)
  - Fine-tune whole network using back-propagation.
  - Freezing or Fine-tuning is optional.
    - Freeze: target task data are scarce, and we want to avoid overfitting
    - Fine-tune: target task data are enough
    - In general, each layer is set to have a different learning rate. (The learning rate of the lower layer is close to zero.)

Fine-tuning

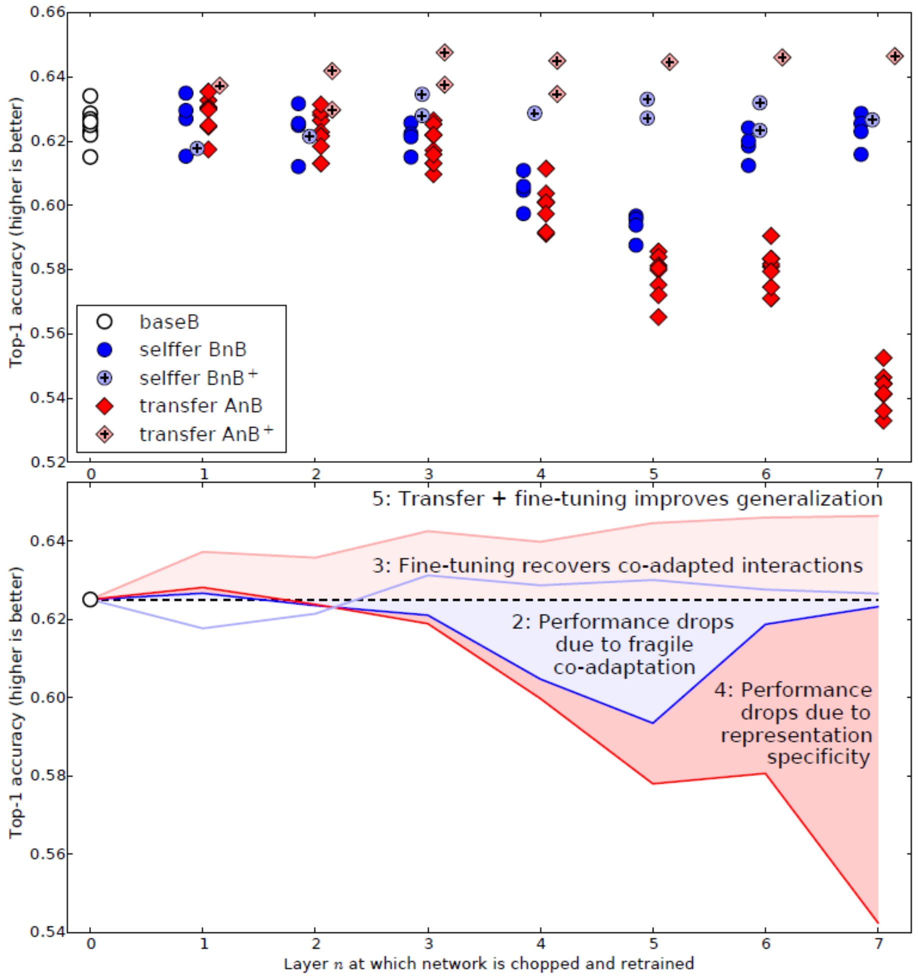
- Idea
  - The blue layers are transferred from source task.
  - Different learning rate could be set to each layer.



"Deep Learning for Computer Vision." *Summer seminar UPC TelecomBCN*

Fine-tuning

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



"How transferable are features in deep neural networks?" NIPS 2014

Domain Adaptation

- Domain adaptation is a major area of research in transfer learning.
- Definition
  - Source domain distribution and target domain distribution are different.
  - Task is same. (e.g. sentiment classification)
  - Labeled data are available only in source domain.
- Example
  - Sentiment classification
    - (sentiment review for food) → (sentiment review for electronics)

Domain Adaptation

- "Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach", ICML 2011.
    - Sentiment classification for reviews.
    - Data: Amazon review dataset
        - Domain: toys, software, food, electronics, etc.
    - Purpose
        - Transfer knowledge of sentiment classification from the source domain (e.g. food) to the target domain (e.g. electronics).

Domain Adaptation

- "Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach", ICML 2011.
  - Proposed method
    1. A Stacked Denoising Autoencoder (SDAE) is trained for all the available domains. (All domains have a common embedding space)
    2. Support Vector Machines (SVM) is trained on the source task.
    3. The classifier (SVM) which is trained at step 2 is transferred to the target domain.

Domain Adaptation

- "Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach", ICML 2011.
  - Results (lower is better)

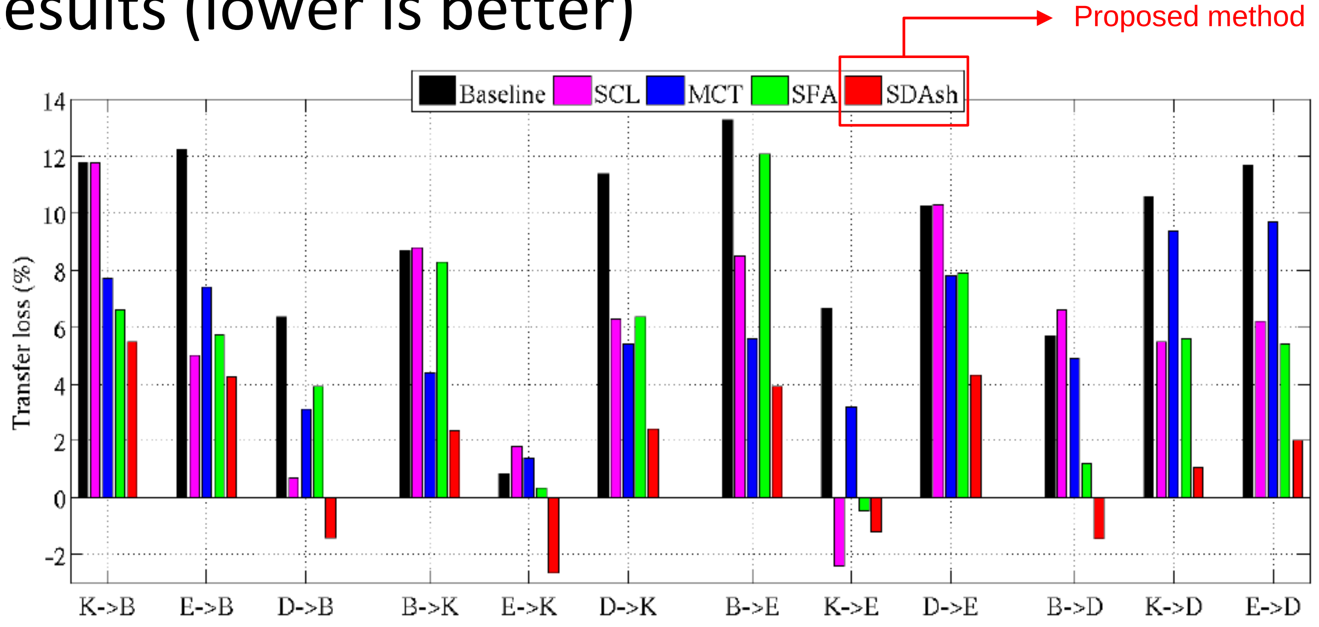


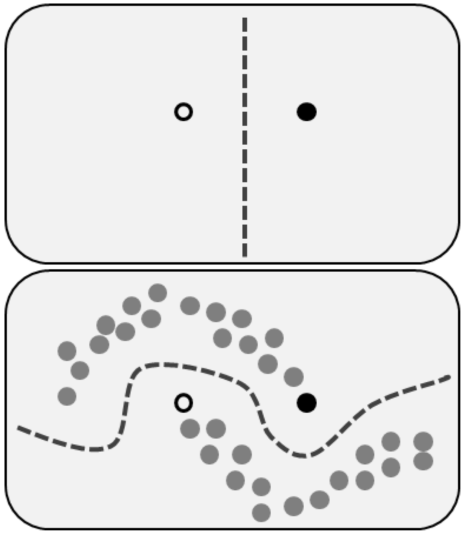
Figure 1. **Transfer losses on the Amazon benchmark** of 4 domains: *Kitchen*(K), *Electronics*(E), *DVDs*(D) and *Books*(B). All methods are trained on the labeled set of one domain and evaluated on the test sets of the others. SDA$_{sh}$ outperforms all others on 11 out of 12 cases.

Outline

- ✓ Overview
- ✓ Greedy Layer-Wise Unsupervised Pretraining
- ✓ Transfer Learning and Domain Adaptation
- ➡ ☐ **Semi-Supervised Disentangling of Causal Factors**
- ☐ Distributed Representation
- ☐ Exponential Gains from Depth
- ☐ Providing Clues to Discover Underlying Causes

Semi-supervised learning

- Definition
  - Using labeled data and unlabeled data for supervised learning (typically a small amount of labeled data with a large amount of unlabeled data)
- Influence of unlabeled data in semi-supervised learning

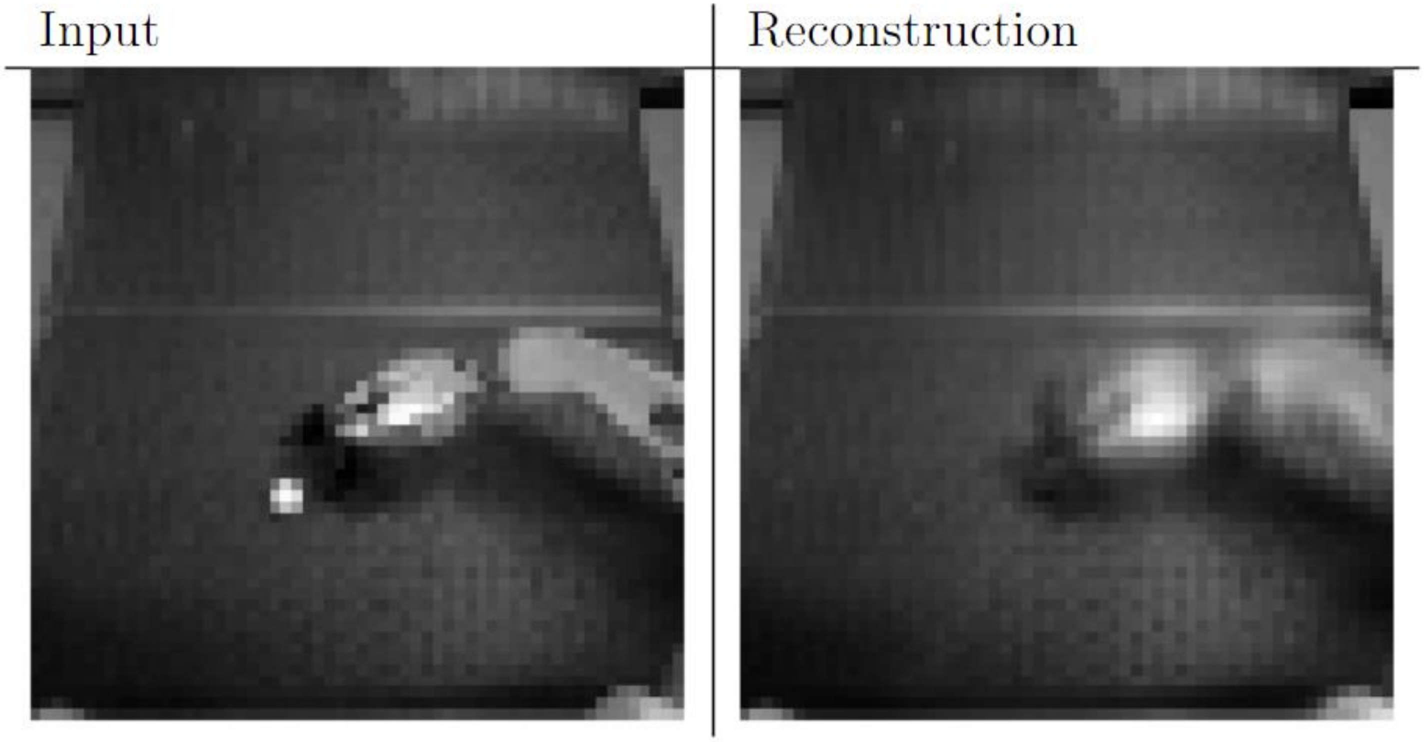


Modifying Definition of Saliency

- Emerging strategy for unsupervised learning is to modify the definition of which underlying causes are most salient.
- Autoencoders and generative models usually optimize a fixed criterion (e.g. MSE)
- These fixed criteria determine which causes are considered salient.
  - MSE in image reconstruction implies that an underlying cause is salient only if data significantly changes the brightness of a large number of pixels.

Modifying definition of Saliency

- An autoencoder trained with MSE has failed to reconstruct a small ball.

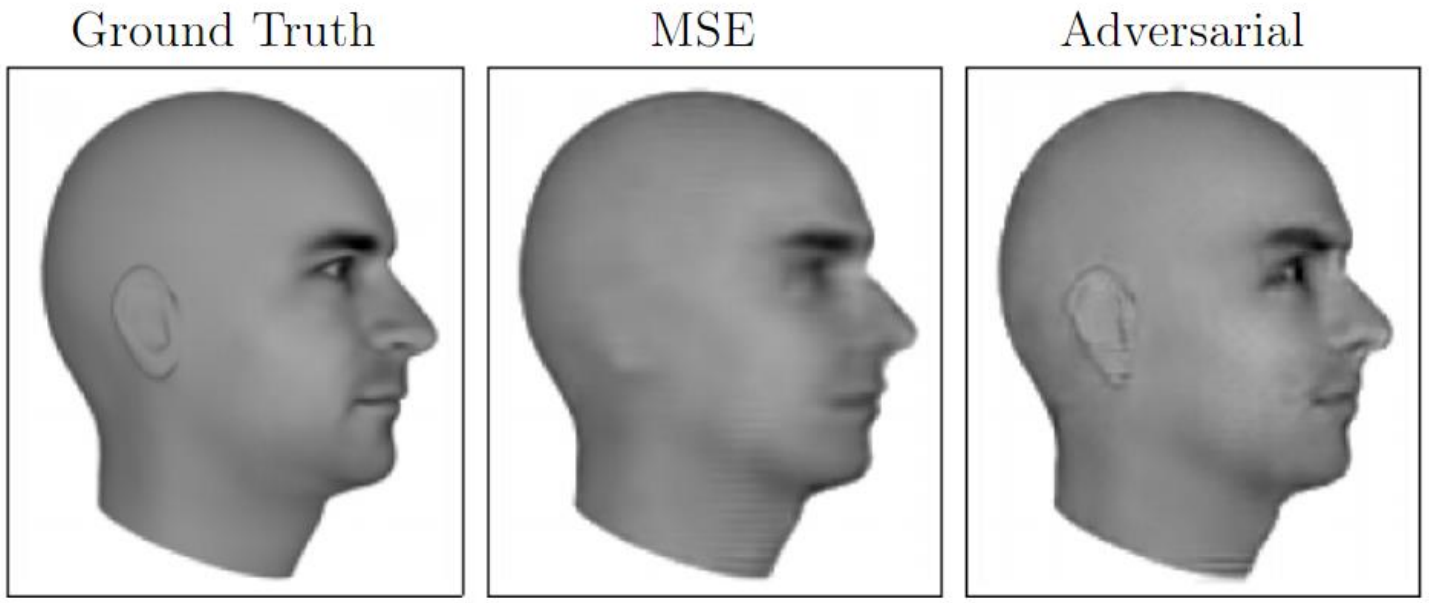


Modifying definition of Saliency

- If a group of pixels follows a highly recognizable pattern then that pattern could be considered salient. (even if that pattern does not involve extreme brightness or darkness)
- GAN detects saliency (chapter 20)
  - A generative model is trained to fool a discriminator.
  - The discriminator attempts to recognize all samples from the generative model as being fake and samples from the training set as being real.
  - Therefore, the network learns how to determine what is salient.

Modifying definition of Saliency

- MSE based model neglects to generate the ears because the ears do not cause an extreme difference in brightness.
- GAN generates ears.

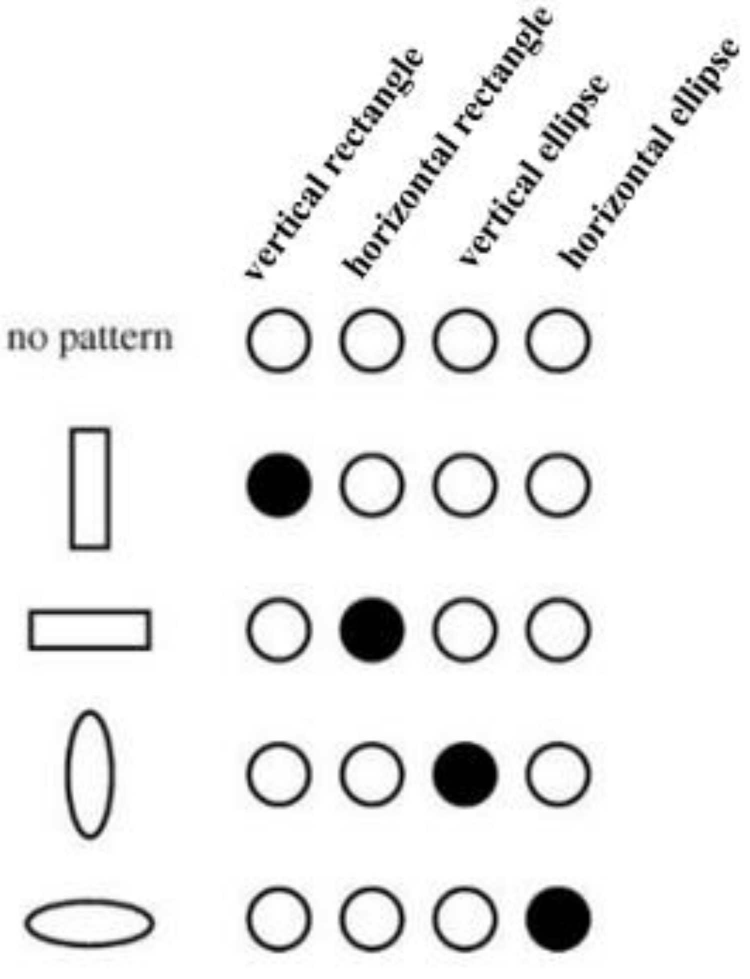


Outline

- ✓ Overview
- ✓ Greedy Layer-Wise Unsupervised Pretraining
- ✓ Transfer Learning and Domain Adaptation
- ✓ Semi-Supervised Disentangling of Causal Factors
- ➡ ☐ **Distributed Representation**
- ☐ Providing Clues to Discover Underlying Causes

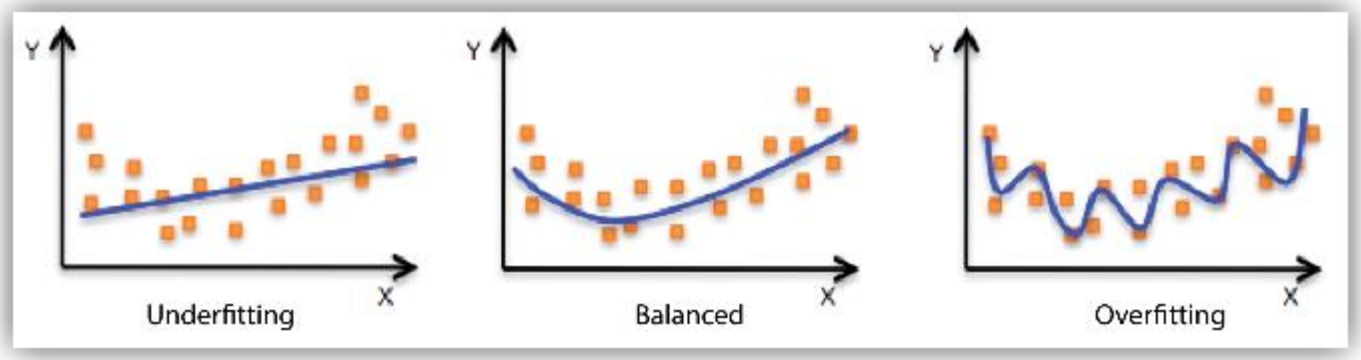
Nondistributed Representation

- Definition
  - Simple representation: **1 neuron dedicated to each thing**
  - Easy to learn
  - Easy to associate with other representations
  - **BUT** inefficient with componential structured data



Nondistributed Representation

- Generalization
  - Term used to describe a model's ability to react to new data. After being trained on a training set it can digest new data and make accurate predictions
  - It is central to the success of a model
  - If the model was trained too well on the training set it could cause **overfitting**
  - The inverse could also happen and is called **underfitting**



Nondistributed Representation

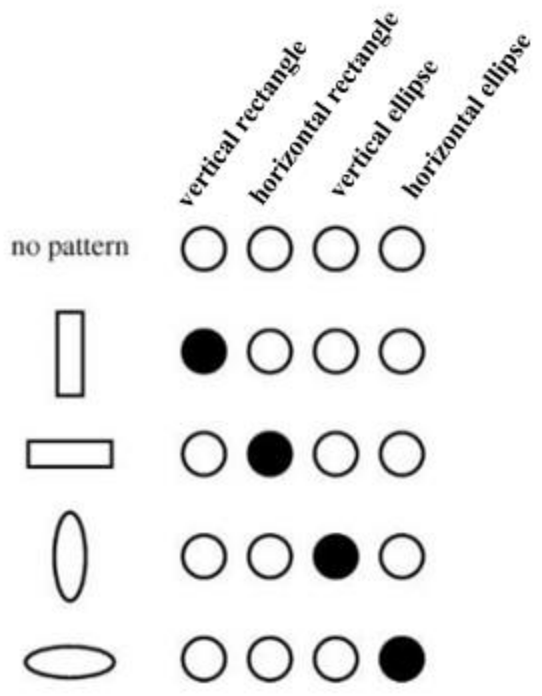
- Generalization in nondistributed representations
  - Some traditional nondistributed learning algorithms generalize only due to the **smoothness** assumption that states that:
    If u $\approx$ v then the target function f has the property : f(u) $\approx$ f(v)
  - The end result of this assumption is that if we have (x, y) for which we know that f(x) $\approx$ y then we choose an estimator $\hat{f}$ that approximately satisfies these constraints while changing as little as possible when moving to a nearby input x + ε
  - **However** this assumption causes the recurring **problem of dimensionality**: we may need at least as many examples as the number of regions.

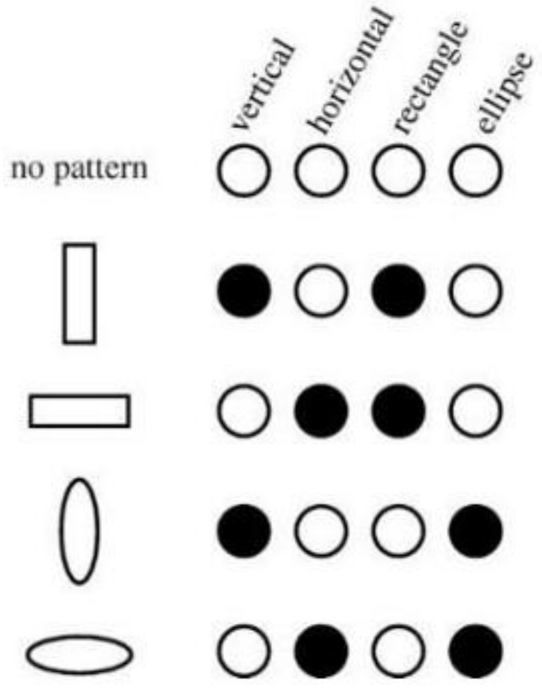
Distributed Representation

- What is a distributed representation ?
  - Each concept is represented by many neurons and each neuron participates in the representation of many concepts
  - Very useful in **representation learning**
  - It can use n features with k values to describe k^n different concepts

Distributed Representation

- What is a distributed representation ?

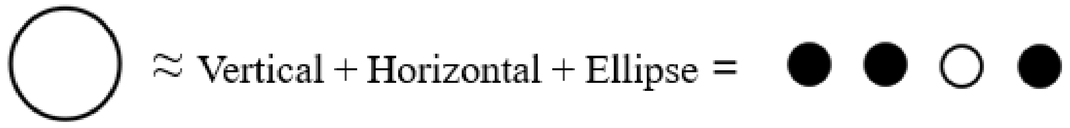




Nondistributed Representation VS Distributed Representation

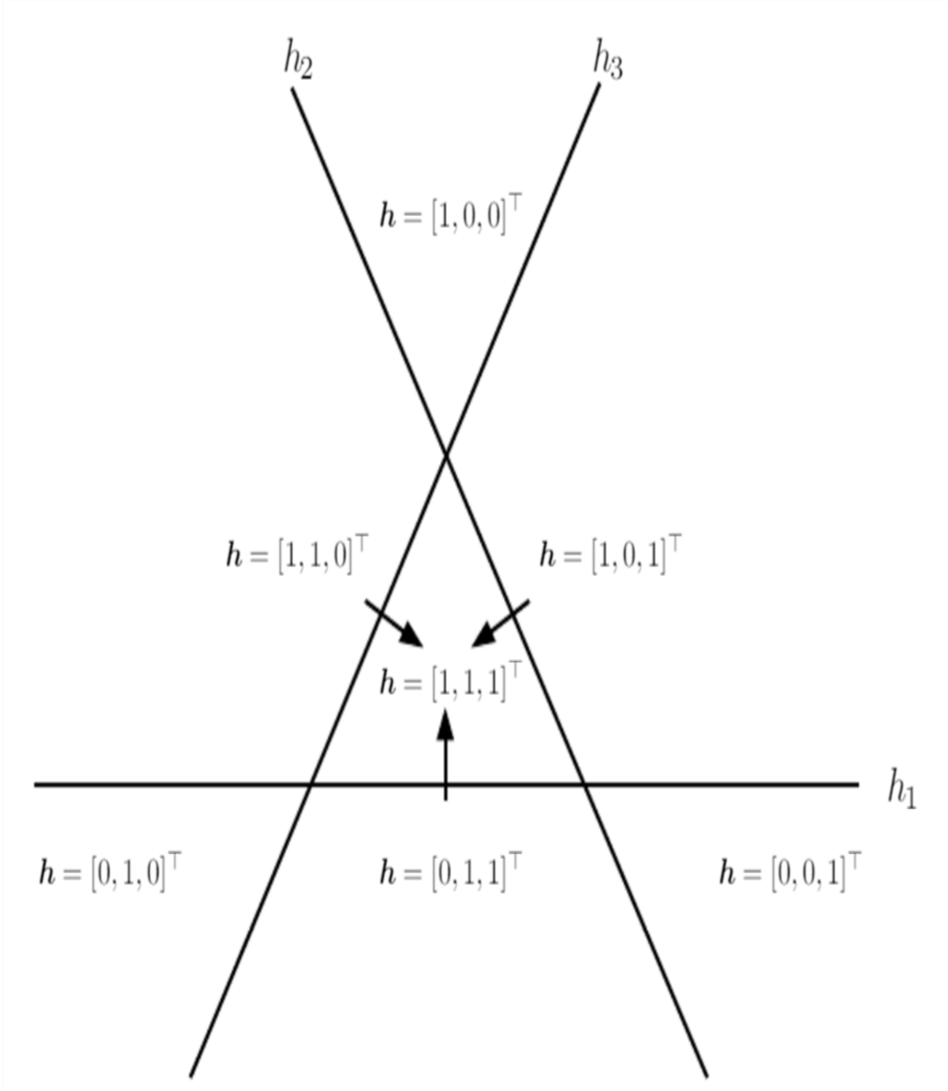
Distributed Representation

- What is a distributed representation ?
  - Nondistributed Representation: new shape would mean an increase in the dimensionality
  - Distributed Representation: we keep the same dimensionality



Distributed Representation

- Example : Learning algorithm based on distributed representation
  - 3 binary features h_1, h_2, h_3 that each divides $\mathbb{R}^2$ into 2 half-planes.
  - Each line represents the decision boundary for h_i



Distributed Representation

- Example : Learning algorithm based on distributed representation

General case of d input dimensions:

- For n features (for each dimension) it assigns unique codes to $O(n^d)$ different regions, while nearest neighbor with m examples assigns unique codes to only m regions

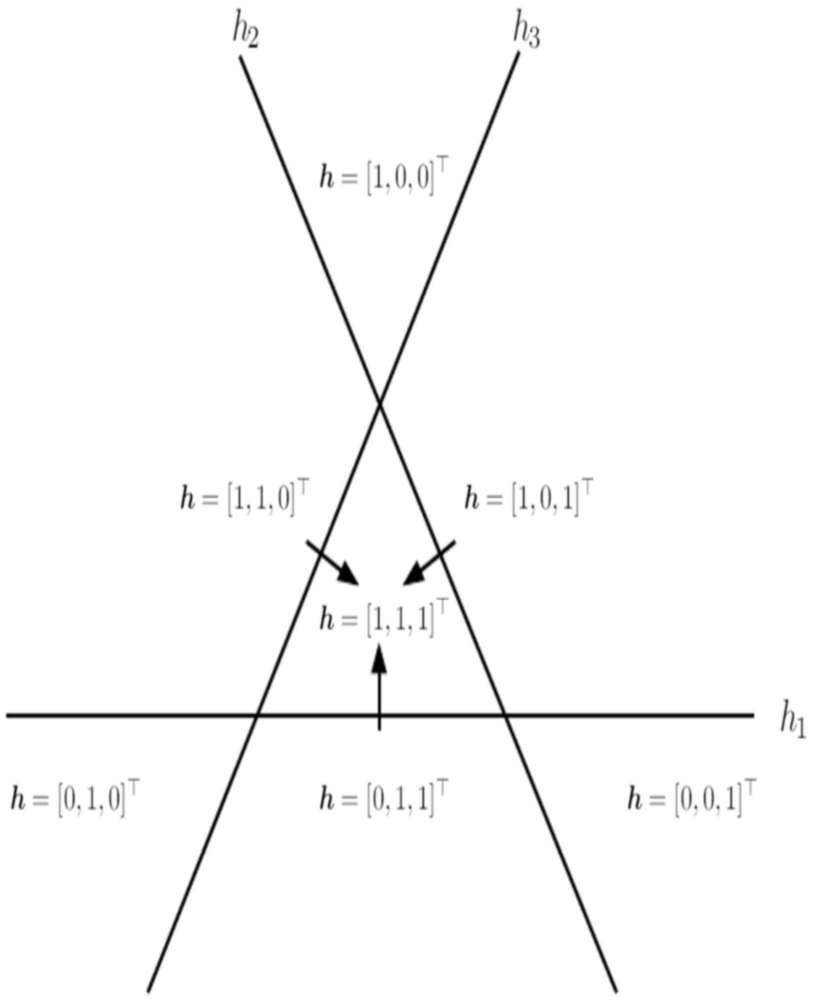
Advantages of Distributed Representation

- Generalization in distributed representations
  - *Generalization arise due to shared attributes* between different concepts
  - Neural language models that operate on distributed representations of words generalize much better than other models.

    Example: **“cat”** and **“dog”** in nondistributed representation are as far as each other as any other symbols. However if we now have a distributed representation that contains for instance **“has_fur”** or **“number_of_legs”** those would have the same values for both **“cat”** and **“dog”**
  - Distributed representations induce a rich *similarity* space in which semantically close concepts are close in distance

Advantages of Distributed Representation

- Statistical Advantage
  - Distributed representations can have a statistical advantage when a complicated structure can be compactly represented using a small number of parameters.
  - In the particular case depicted in the figure, the number of regions this binary feature representation can distinguish is $O(2^d)$



Advantages of Distributed Representation

- Statistical Advantage
  - This provides geometric argument to explain the generalization power of distributed representation:

    With $O(d)$ parameters we can distinctly represent $O(2^d)$ regions in input space.

    While if we had used one symbol for each region specifying $O(2^d)$ regions would require $O(2^d)$ examples.

Advantages of Distributed Representation

- Exponential Gains from Depth
  - Compositions of nonlinearity can give an exponential boost to statistical efficiency
  - Many networks with saturating nonlinearities with a single layer can be shown to be universal approximators (can approximate a large class of functions). However the required number of hidden units may be very large.

Outline

- ✓ Overview
- ✓ Greedy Layer-Wise Unsupervised Pretraining
- ✓ Transfer Learning and Domain Adaptation
- ✓ Semi-Supervised Disentangling of Causal Factors
- ✓ Distributed Representation
- ➡ ☐ **Providing Clues to Discover Underlying Causes**

Providing clues to discover underlying causes

- Regularization
  - Reduces overfitting by adding a complexity penalty to the loss function. The idea behind is that models that overfit the data are complex models that have for example too many parameters
  - To find the best model, a common method is to define a loss function or cost function that describes how well the model fits the data
  - The goal is to find the model that minimizes the function
  - Regularization can be motivated as a technique to improve the generalizability of a learned model

Providing clues to discover underlying causes

- What makes one representation better ?
  - An ideal representation disentangles the underlying causal factors of variation that generated the data.
  - Strategy = Introducing clues that help the learning to find these underlying causal factors of variation (e.g. supervised learning provide a label **y** to each **x)**
  - It has been shown that regularization strategies are necessary to obtain a good generalization
  - It is impossible to find a universally superior regularization strategy; a goal of deep learning is to find a set of generic regularization strategies that are applicable to a wide variety of AI tasks

Providing clues to discover underlying causes

- List of generic regularization strategies
  - Ways that learning algorithms can be encouraged to discover features that correspond to underlying factors
  - Smoothness:
    - Assumption that $f(\mathbf{x} + \mathbf{d}\varepsilon) \approx f(\mathbf{x})$, d unit and small ε
    - Allows the learner to generalize from training examples to nearby points in input space
    - Insufficient in terms of dimensionality
  - Linearity:
    - Allows predictions even very far from the observed data
    - Can lead to overly extreme predictions (e.g., in regression)

Providing clues to discover underlying causes

- List of generic regularization strategies
  - Causal factors
    - Factors of variation described by the learned representation h are treated as the causes of observed data x.
    - Advantageous for semi-supervised learning
  - Depth or a hierarchical organization of explanatory factors
    - Expresses our belief that ml task should be accomplished via a multi-step program, with each step referring back to the output of the processing accomplished via previous steps

Providing clues to discover underlying causes

- List of generic regularization strategies
  - Shared factors across tasks: sharing of statistical strength between the tasks
  - Manifolds: area of low-dimensionality where data lives
  - Natural Clustering: each connected manifold in the input space may be assigned to a single class
  - Temporal and spatial coherence: most important explanatory factors change slowly over time
  - Sparsity: impose a prior that any feature that can be interpreted as "present" or "absent" should be absent most of time
  - Simplicity of factor dependencies: in good high-level representations, the factors are related to each other through simple dependencies (e.g., factorial distributions)

What you need to know

- Greedy Layer-Wise Unsupervised Pretraining
- Transfer Learning and Domain Adaptation
- Semi-Supervised Disentangling of Causal Factors
- Distributed Representation
- Exponential Gains from Depth
- Providing Clues to Discover Underlying Causes

Questions?