Semisupervised Learning, Transfer Learning, and the Future at a Glance

Shan-Hung Wu
shwu@cs.nthu.edu.tw

Department of Computer Science,
National Tsing Hua University, Taiwan

Machine Learning

Outline

1. **Semisupervised Learning**
   - Label Propagation
   - Semisupervised GAN
   - Semisupervised Clustering
2. **Transfer Learning**
   - Multitask Learning & Weight Initiation
   - Domain Adaptation
   - Zero Shot Learning
   - Unsupervised TL
3. **The Future at a Glance**

Outline

Semisupervised Learning

- Labels may be expansive to get in some applications
  - E.g., drug design, medical diagnosis, etc.

Semisupervised Learning

- Labels may be expansive to get in some applications
  - E.g., drug design, medical diagnosis, etc.
- On the other hand, unlabeled data may be plentiful and cheap

Semisupervised Learning

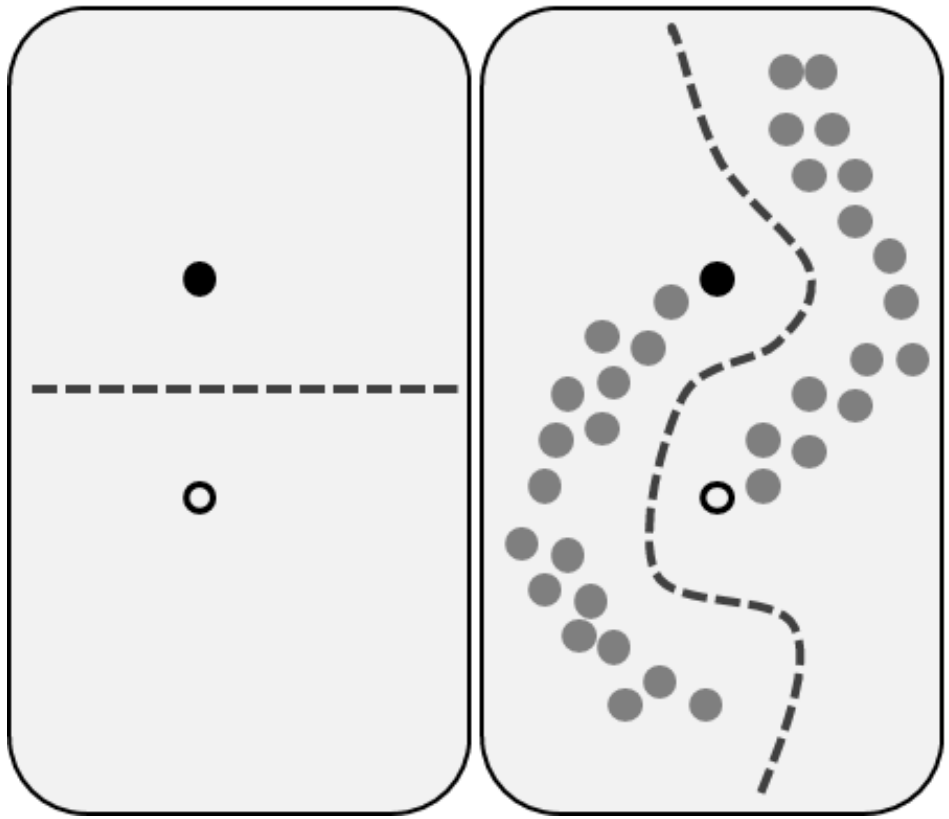
- Labels may be expansive to get in some applications
  - E.g., drug design, medical diagnosis, etc.
- On the other hand, unlabeled data may be plentiful and cheap
- ***Semisupervised learning***: exploit the unlabeled data to improve the performance of a supervised learner

Semisupervised Learning

- Labels may be expansive to get in some applications
  - E.g., drug design, medical diagnosis, etc.
- On the other hand, unlabeled data may be plentiful and cheap
- ***Semisupervised learning***: exploit the unlabeled data to improve the performance of a supervised learner
- Training set: $\mathbb{X} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{L} \cup \{\boldsymbol{x}^{(i)}\}_{i=L+1}^{N}$
  - Usually, $L \ll N$

Why Unlabeled Examples Help?

Why Unlabeled Examples Help?

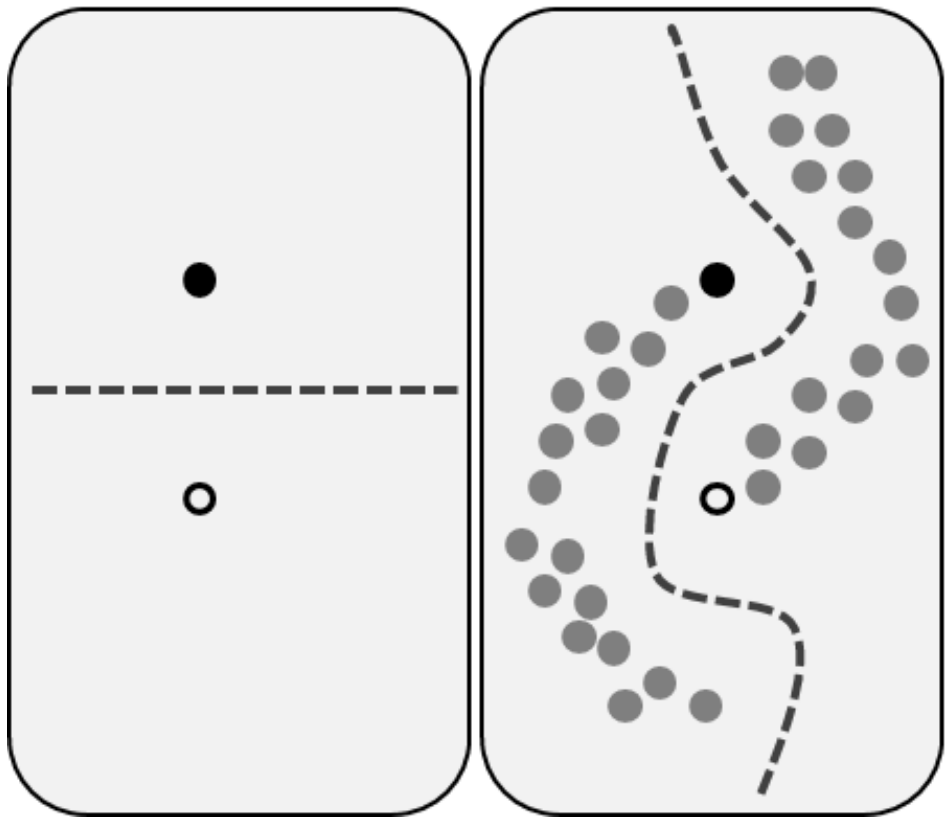


- Unlabeled examples help explain the marginal distribution of $\mathbf{x}$

Outline

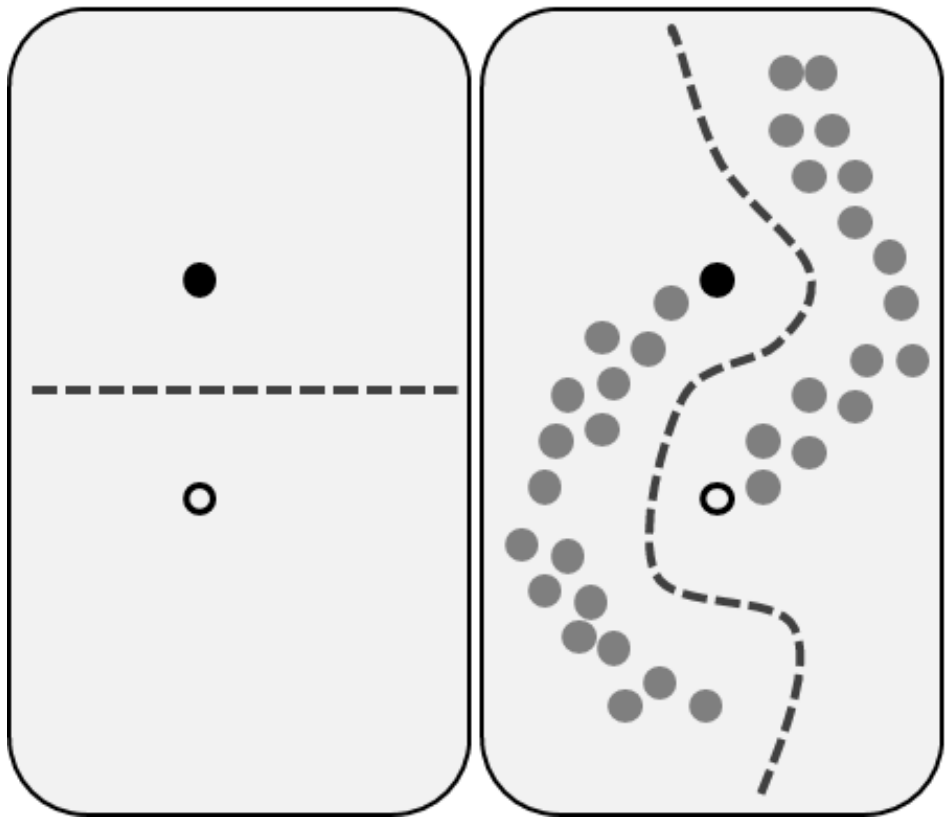
Label Propagation

- Assume that points of the same class reside in the same manifold



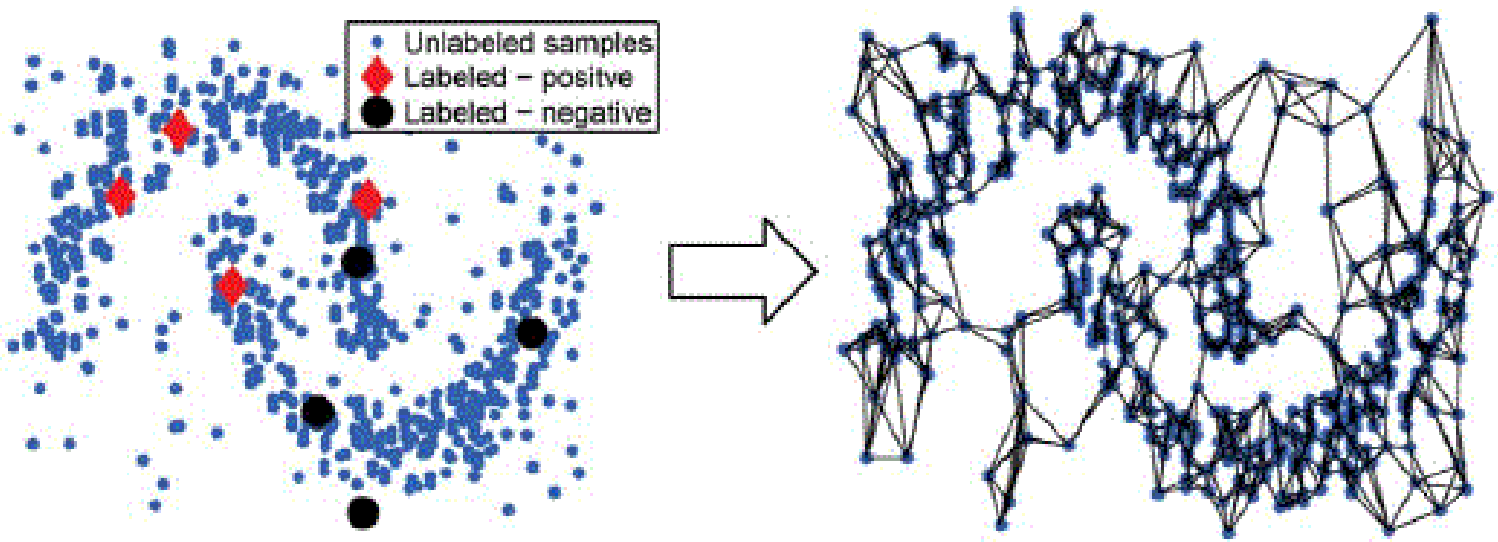
Label Propagation

- Assume that points of the same class reside in the same manifold
- So, the label should "propagate" along the local tangent spaces



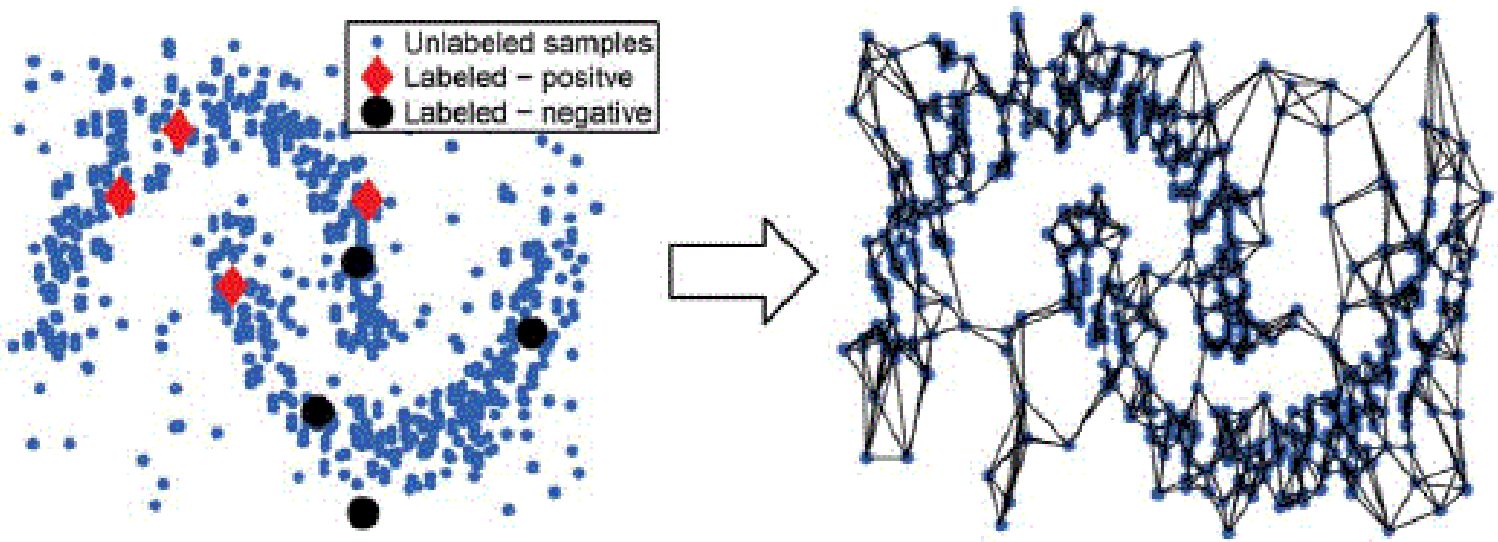
Label Propagation along Local Similarity Graph

1. Construct a ***local*** similarity graph for all points
   - E.g., K-NN graph, Parzen-Window graph, etc.
   - Using Euclidean distance (as a manifold is "locally linear")



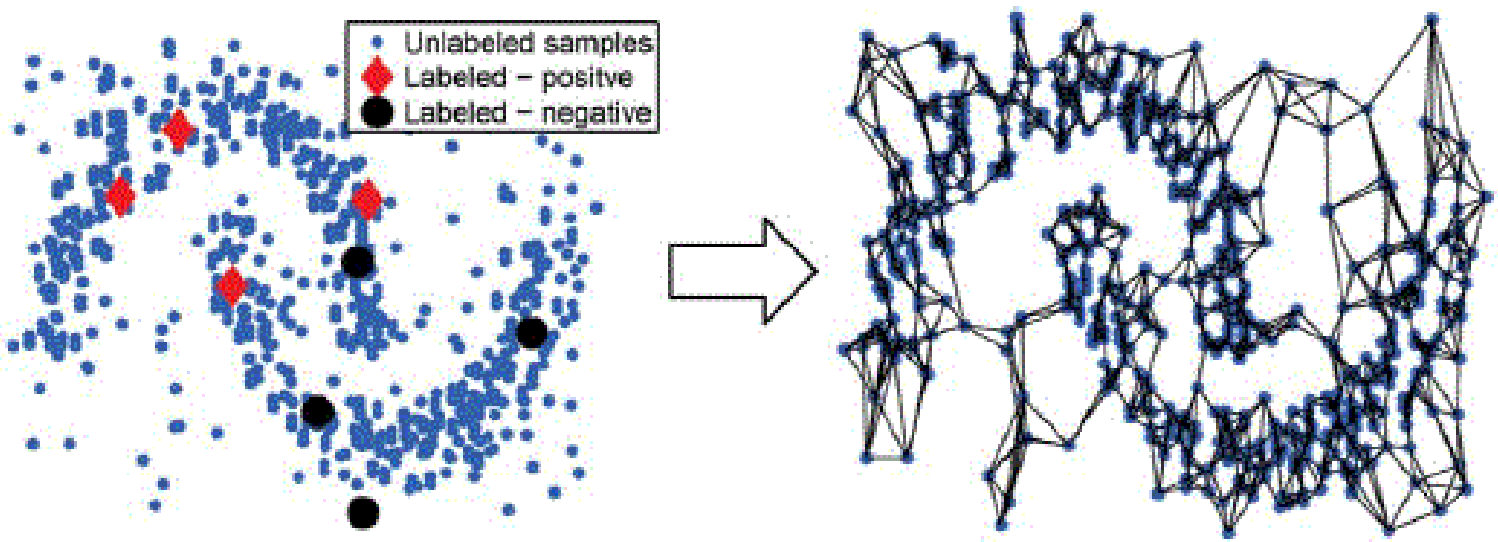
Label Propagation along Local Similarity Graph

1. Construct a ***local*** similarity graph for all points
   - E.g., K-NN graph, Parzen-Window graph, etc.
   - Using Euclidean distance (as a manifold is "locally linear")
2. Find f such that it tends to assign the same label to any two connected points $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(j)}$
   - Lebel propagates along locally linear (tangent) spaces



Label Propagation along Local Similarity Graph

1. Construct a ***local*** similarity graph for all points
   - E.g., K-NN graph, Parzen-Window graph, etc.
   - Using Euclidean distance (as a manifold is "locally linear")
2. Find f such that it tends to assign the same label to any two connected points $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(j)}$
   - Lebel propagates along locally linear (tangent) spaces
   - How?



Regularization using Graph Laplacian

- Let $\boldsymbol{S} \in \mathbb{R}^{N \times N}$ be the adjacent matrix of the local similarity graph
  - $S_{i,j}$ the edge weight between point i and j

Regularization using Graph Laplacian

- Let $\boldsymbol{S} \in \mathbb{R}^{N \times N}$ be the adjacent matrix of the local similarity graph
  - $S_{i,j}$ the edge weight between point i and j
- We can add a penalty term in the cost function [2]:

$$\Omega[f] = \frac{1}{2} \sum_{i,j=1}^{N} S_{i,j} \left(f(\boldsymbol{x}^{(i)}) - f(\boldsymbol{x}^{(j)}) \right)^2$$

Regularization using Graph Laplacian

- Let $\boldsymbol{S} \in \mathbb{R}^{N \times N}$ be the adjacent matrix of the local similarity graph
  - $S_{i,j}$ the edge weight between point i and j
- We can add a penalty term in the cost function [2]:

$$\Omega[f] = \frac{1}{2} \sum_{i,j=1}^{N} S_{i,j} \left(f(\boldsymbol{x}^{(i)}) - f(\boldsymbol{x}^{(j)}) \right)^2$$

- Let $\boldsymbol{f} \in \mathbb{R}^N$ be the prediction vector, where $f_i = f(\boldsymbol{x}^{(i)})$
- The above penalty term can be simplified to

$$\Omega[f] = \boldsymbol{f}^\top \boldsymbol{L} \boldsymbol{f}$$

where $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{S}$ is called the ***graph Laplacian*** matrix [Proof]

- $D_{i,i} = \sum_j S_{i,j}$ the diagonal "degree" matrix

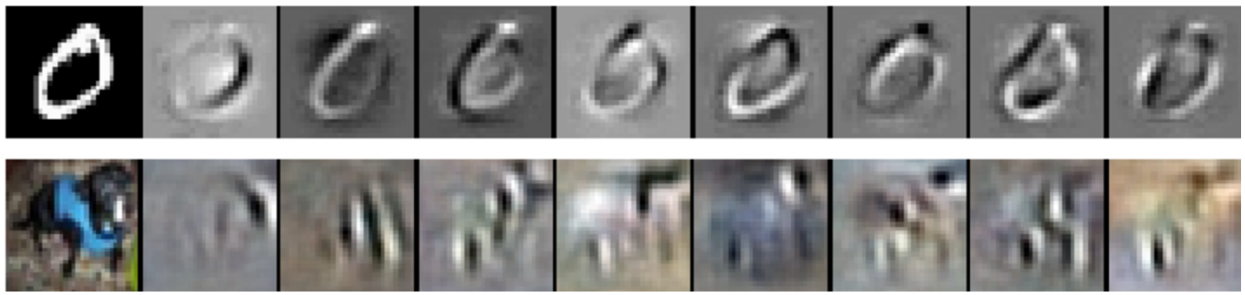
Semisupervised Tangent Prop

- Another simple way is to extend Tangent Prop [12]

Semisupervised Tangent Prop

- Another simple way is to extend Tangent Prop [12]

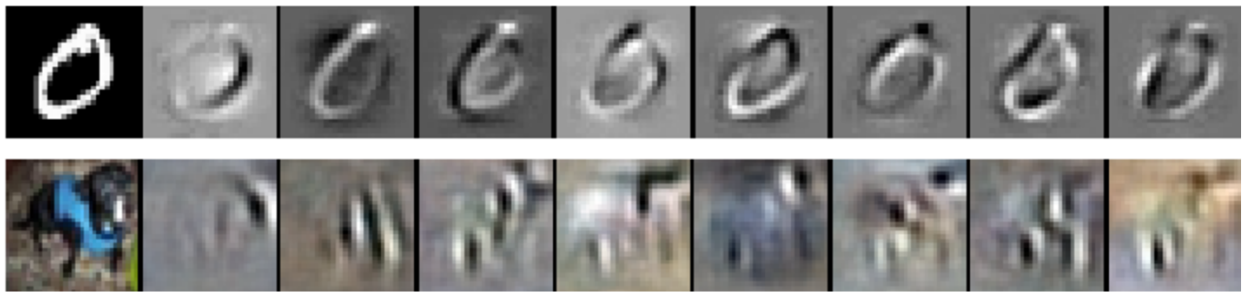
1. Find tangent vectors $\{\boldsymbol{v}^{(i,j)}\}_j$ of ***all*** (including unlabeled) points $\boldsymbol{x}^{(i)}$
   - Using, e.g., contractive or denoising autoencoders



Semisupervised Tangent Prop

- Another simple way is to extend Tangent Prop [12]

1. Find tangent vectors $\{\boldsymbol{v}^{(i,j)}\}_j$ of ***all*** (including unlabeled) points $\boldsymbol{x}^{(i)}$
   - Using, e.g., contractive or denoising autoencoders



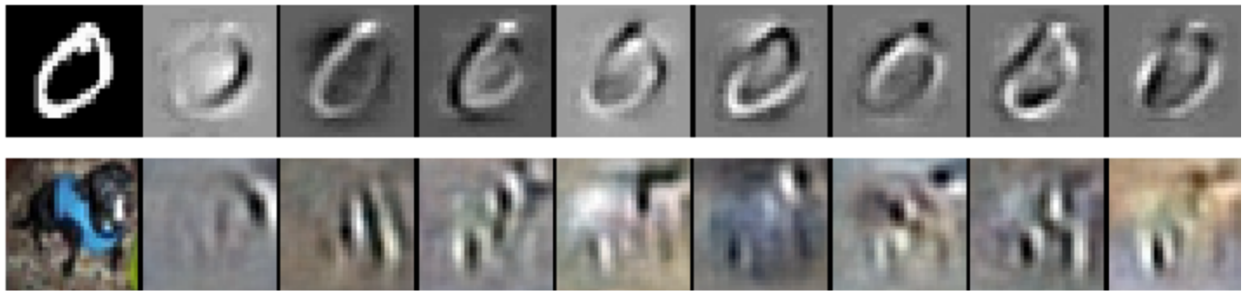
2. Train f with cost penalty:

$$\Omega[f] = \sum_{i=1}^{N} \sum_{j} \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^{(i)})^{\top} \boldsymbol{v}^{(i,j)}$$

Semisupervised Tangent Prop

- Another simple way is to extend Tangent Prop [12]

1. Find tangent vectors $\{\boldsymbol{v}^{(i,j)}\}_j$ of ***all*** (including unlabeled) points $\boldsymbol{x}^{(i)}$
   - Using, e.g., contractive or denoising autoencoders



2. Train f with cost penalty:

$$\Omega[f] = \sum_{i=1}^{N} \sum_{j} \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^{(i)})^{\top} \boldsymbol{v}^{(i,j)}$$

- Points in the same manifold (backed by both labeled and unlabeled points) share the same label

Outline

Semisupervised GAN

- Generator and discriminator do not need to play a zero-sum game

Semisupervised GAN

- Generator and discriminator do not need to play a zero-sum game
- Semisupervised GAN [11]:
- Discriminator learns one extra class "fake" in addition to K classes
  - Softmax output units $\boldsymbol{a}^{(L)} = \hat{\boldsymbol{\rho}} \in \mathbb{R}^{K+1}$ for $\mathrm{P}(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\Theta}) \sim \text{Categorical}(\boldsymbol{\rho})$

Semisupervised GAN

- Generator and discriminator do not need to play a zero-sum game
- Semisupervised GAN [11]:
- Discriminator learns one extra class "fake" in addition to K classes
  - Softmax output units $\boldsymbol{a}^{(L)} = \hat{\boldsymbol{\rho}} \in \mathbb{R}^{K+1}$ for $\mathrm{P}(\mathbf{y} \mid \mathbf{x}, \Theta) \sim \text{Categorical}(\boldsymbol{\rho})$
- Cost function (L labeled, M fake, $N - L$ unlabeled):

$$\arg\min_{\Theta_{\text{gen}}} \max_{\Theta_{\text{dis}}} \sum_{n=1}^{L} \sum_{j=1}^{K} 1(y^{(n)} = j) \log \hat{\rho}_j^{(n)} + \sum_{m=1}^{M} \log \hat{\rho}_{K+1}^{(m)} + \sum_{n=L+1}^{N} \log(1 - \hat{\rho}_{K+1}^{(n)})$$

Semisupervised GAN

- Generator and discriminator do not need to play a zero-sum game
- Semisupervised GAN [11]:
- Discriminator learns one extra class "fake" in addition to K classes
  - Softmax output units $\boldsymbol{a}^{(L)} = \hat{\boldsymbol{\rho}} \in \mathbb{R}^{K+1}$ for $\mathrm{P}(\mathbf{y} \mid \mathbf{x}, \Theta) \sim \text{Categorical}(\boldsymbol{\rho})$
- Cost function (L labeled, M fake, $N - L$ unlabeled):

$$\arg\min_{\Theta_{\text{gen}}} \max_{\Theta_{\text{dis}}} \sum_{n=1}^{L} \sum_{j=1}^{K} 1(y^{(n)} = j) \log \hat{\rho}_j^{(n)} + \sum_{m=1}^{M} \log \hat{\rho}_{K+1}^{(m)} + \sum_{n=L+1}^{N} \log(1 - \hat{\rho}_{K+1}^{(n)})$$

- Real, labeled points should be classified correctly

Semisupervised GAN

- Generator and discriminator do not need to play a zero-sum game
- Semisupervised GAN [11]:
- Discriminator learns one extra class "fake" in addition to K classes
  - Softmax output units $\boldsymbol{a}^{(L)} = \hat{\boldsymbol{\rho}} \in \mathbb{R}^{K+1}$ for $\mathrm{P}(\mathbf{y} \mid \mathbf{x}, \Theta) \sim \text{Categorical}(\boldsymbol{\rho})$
- Cost function (L labeled, M fake, $N - L$ unlabeled):

$$\arg\min_{\Theta_{\text{gen}}} \max_{\Theta_{\text{dis}}} \sum_{n=1}^{L} \sum_{j=1}^{K} 1(y^{(n)} = j) \log \hat{\rho}_j^{(n)} + \sum_{m=1}^{M} \log \hat{\rho}_{K+1}^{(m)} + \sum_{n=L+1}^{N} \log(1 - \hat{\rho}_{K+1}^{(n)})$$

- Real, labeled points should be classified correctly
- Generated point should be identified as fake

Semisupervised GAN

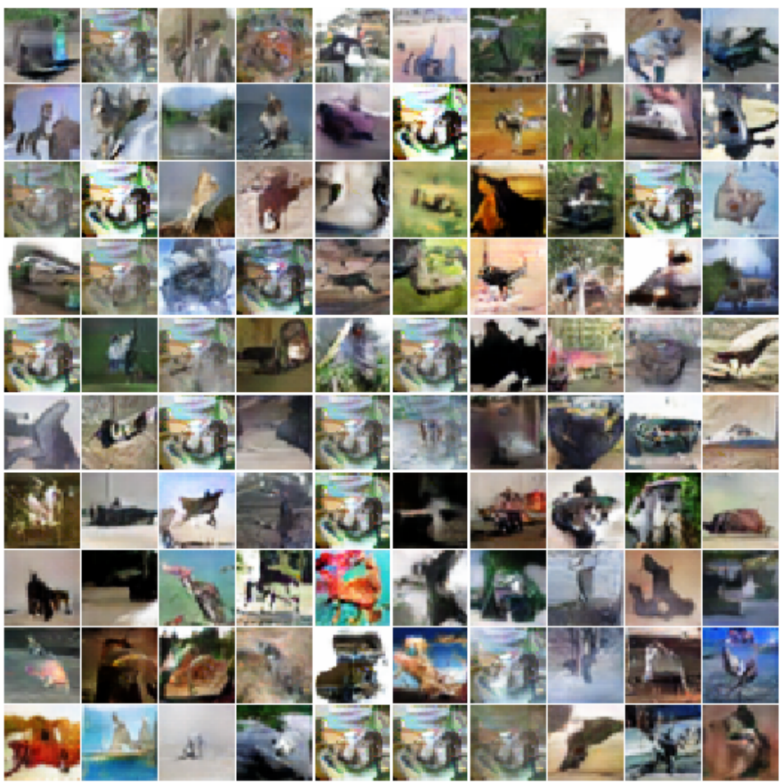
- Generator and discriminator do not need to play a zero-sum game
- Semisupervised GAN [11]:
- Discriminator learns one extra class "fake" in addition to K classes
  - Softmax output units $\boldsymbol{a}^{(L)} = \hat{\boldsymbol{\rho}} \in \mathbb{R}^{K+1}$ for $\mathrm{P}(\mathbf{y} \mid \mathbf{x}, \Theta) \sim \text{Categorical}(\boldsymbol{\rho})$
- Cost function (L labeled, M fake, $N - L$ unlabeled):

$$\arg\min_{\Theta_{\text{gen}}} \max_{\Theta_{\text{dis}}} \sum_{n=1}^{L} \sum_{j=1}^{K} 1(y^{(n)} = j) \log \hat{\rho}_j^{(n)} + \sum_{m=1}^{M} \log \hat{\rho}_{K+1}^{(m)} + \sum_{n=L+1}^{N} \log(1 - \hat{\rho}_{K+1}^{(n)})$$

- Real, labeled points should be classified correctly
- Generated point should be identified as fake
- Real, unlabeled points can be in any class except $K+1$

Performance

- State-of-the-art classification performance given:
  - 100 labeled points (out of 60K) in MNIST
  - 4K labeled points (out of 50K) in CIFAR-10
- With generators:



Outline

Clustering

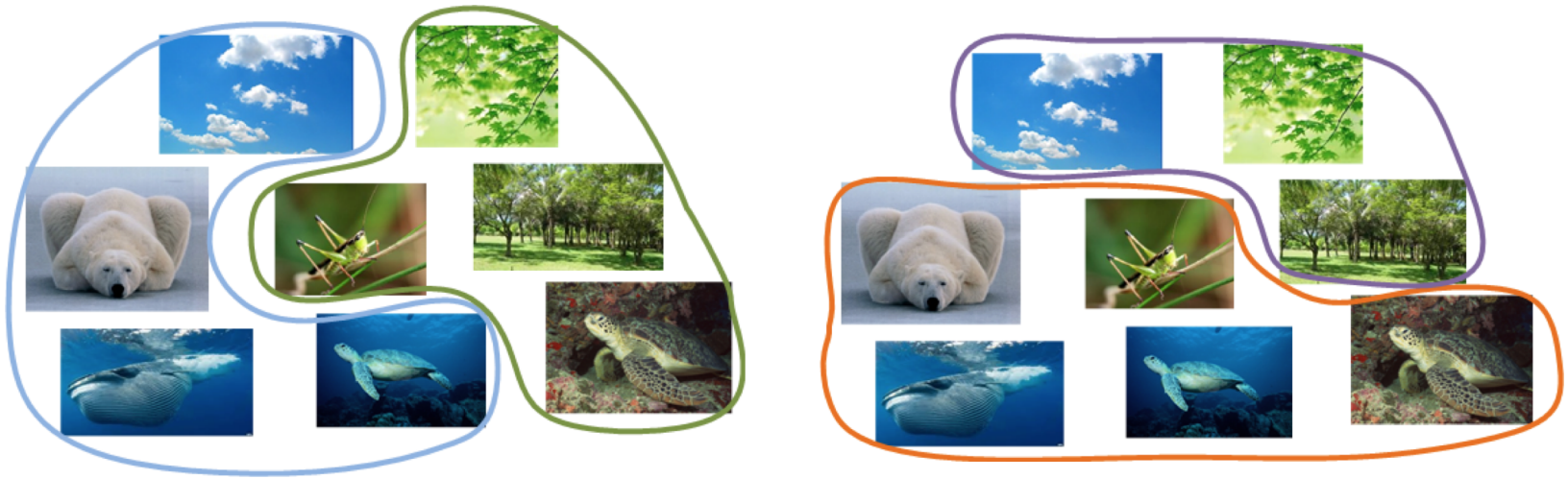
- Clustering is an ill-posed problem

Clustering

- Clustering is an ill-posed problem
- E.g., how to cluster the following images into two group?

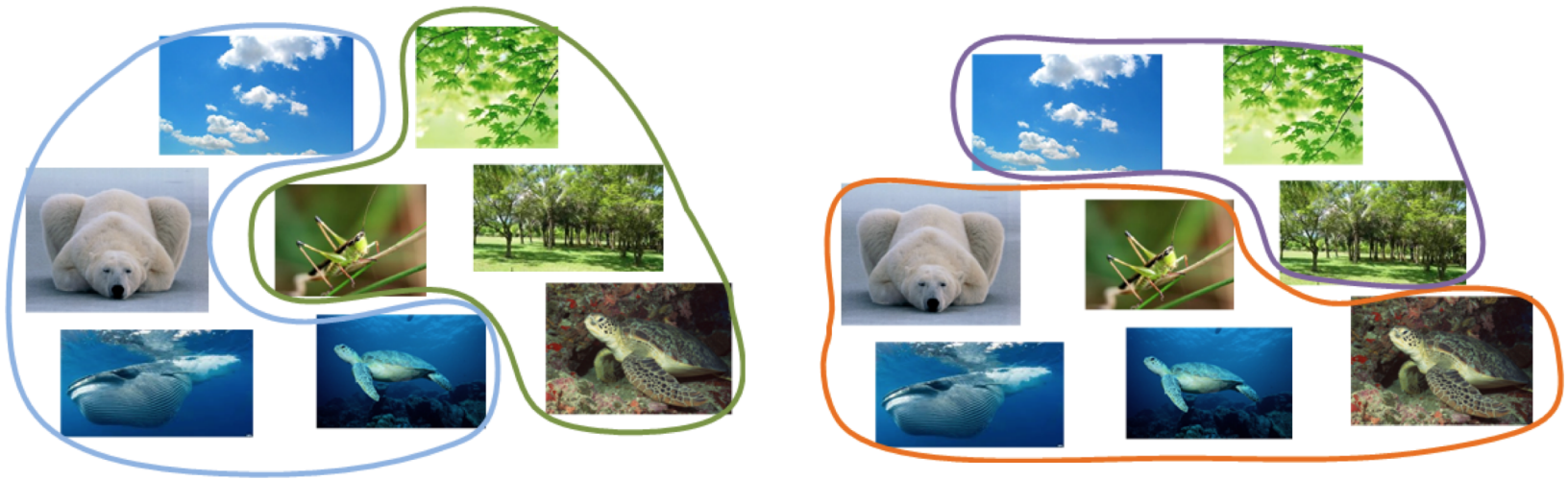
Semisupervised Clustering

- Different users may have different answers:



Semisupervised Clustering

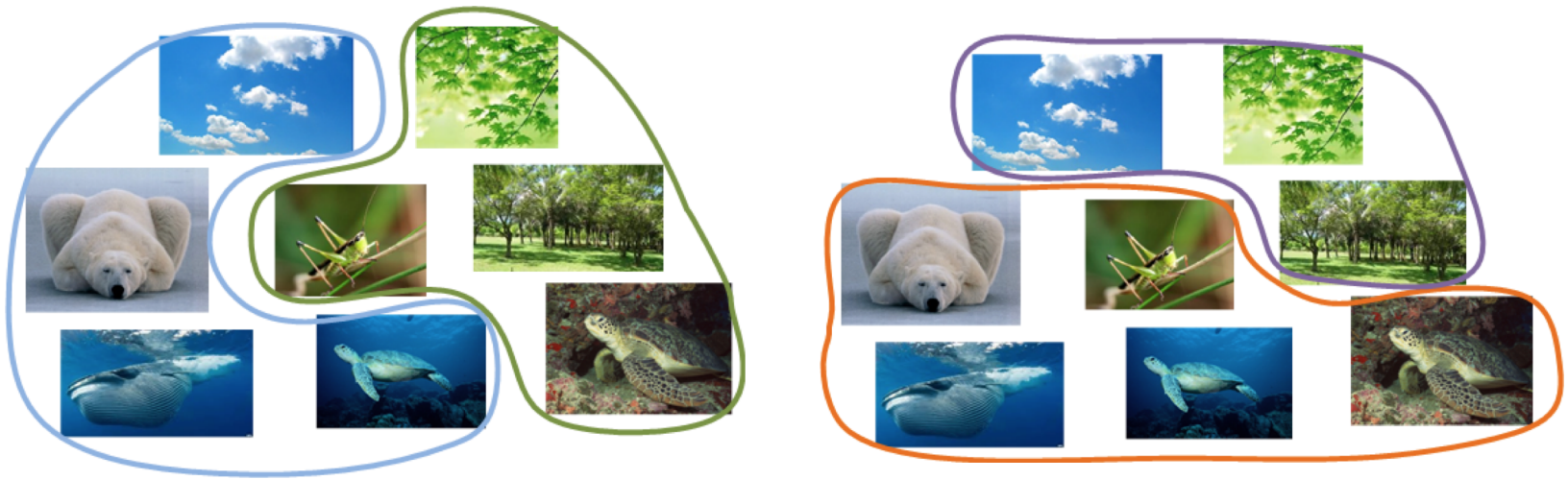
- Different users may have different answers:



- User-perceived clusters $\neq$ clusters learned from data
-

Semisupervised Clustering

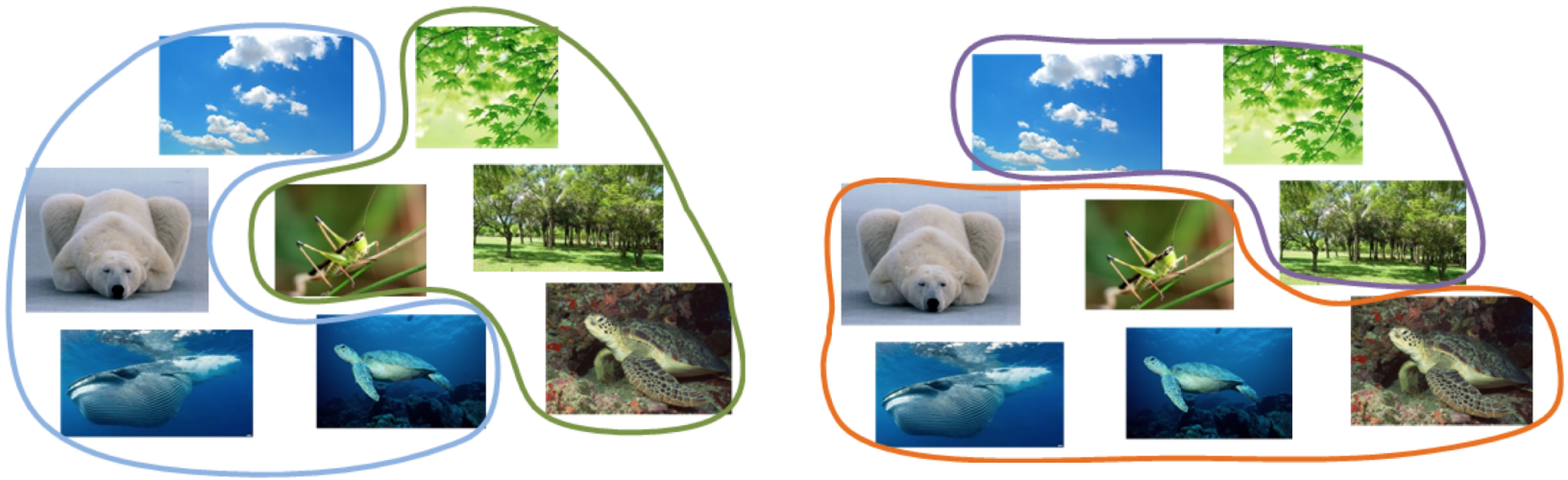
- Different users may have different answers:



- User-perceived clusters $\neq$ clusters learned from data
- ***Semisupervised clustering***: to ask some ***side information*** from the user to better uncover the user perspective

Semisupervised Clustering

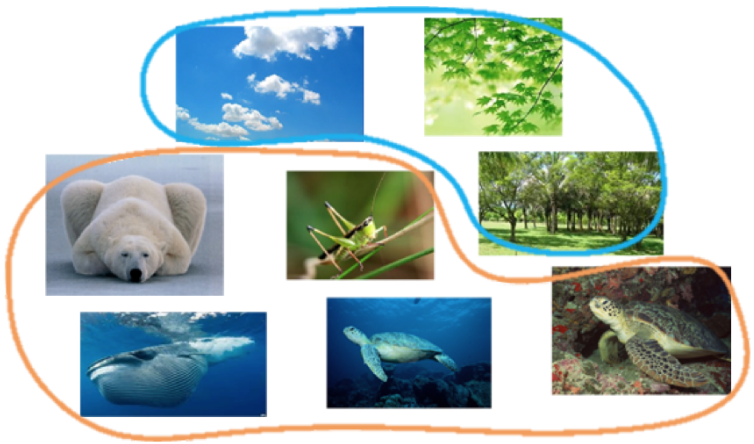
- Different users may have different answers:



- User-perceived clusters $\neq$ clusters learned from data
- ***Semisupervised clustering***: to ask some ***side information*** from the user to better uncover the user perspective
  - In what form?

Point-Level Supervision

- Side info: must-links and/or cannot-links

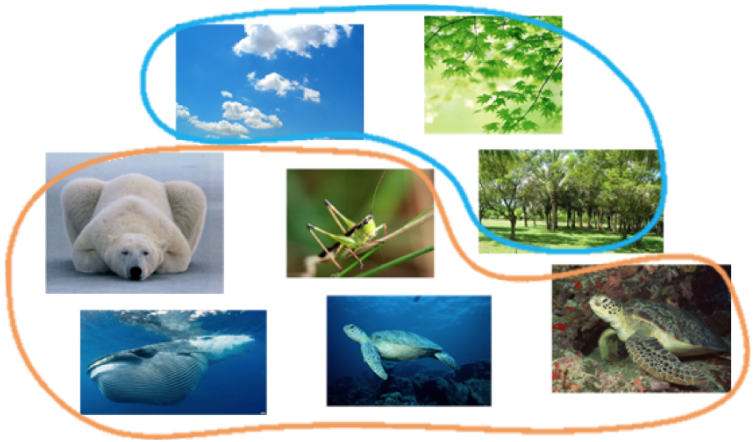


User-perceived Clusters

Data-driven Clusters

Point-Level Supervision

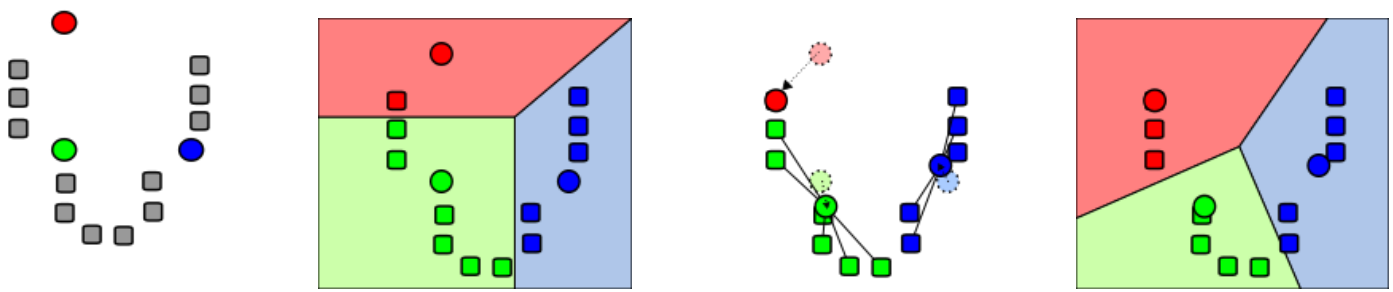
- Side info: must-links and/or cannot-links



User-perceived Clusters

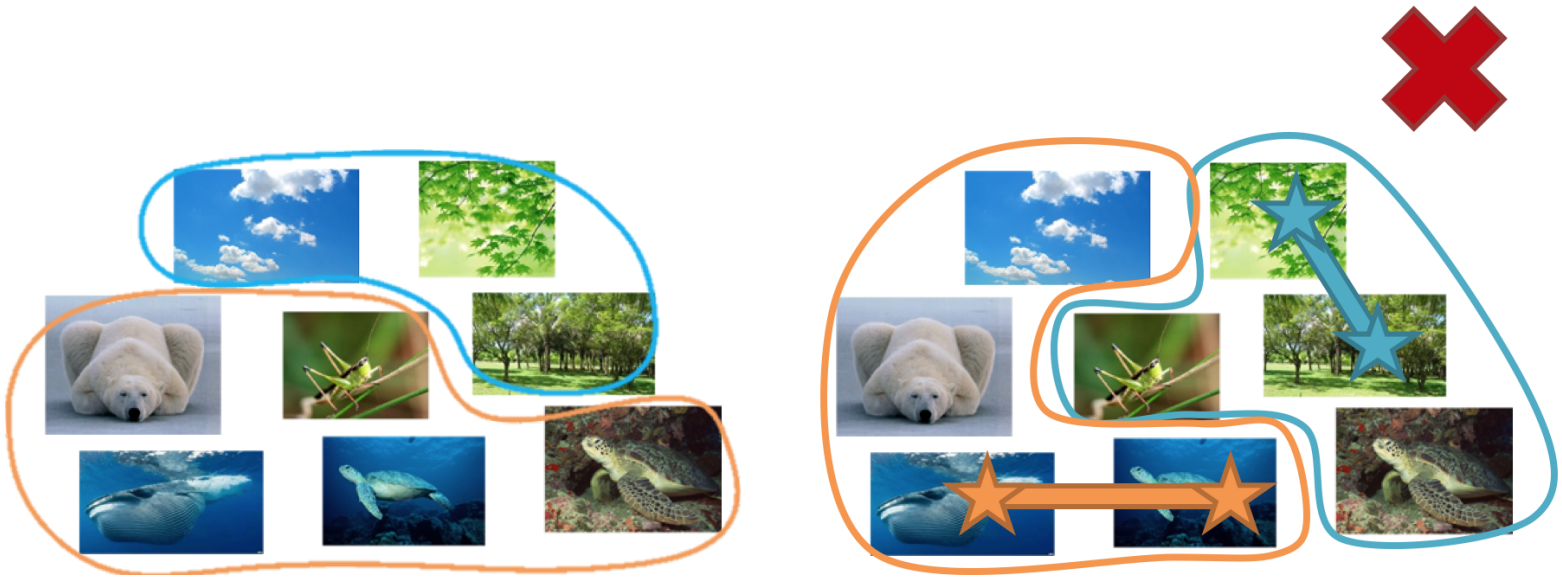
Data-driven Clusters

- Constrained K-means [13]: to assign points to clusters without violating the constraints



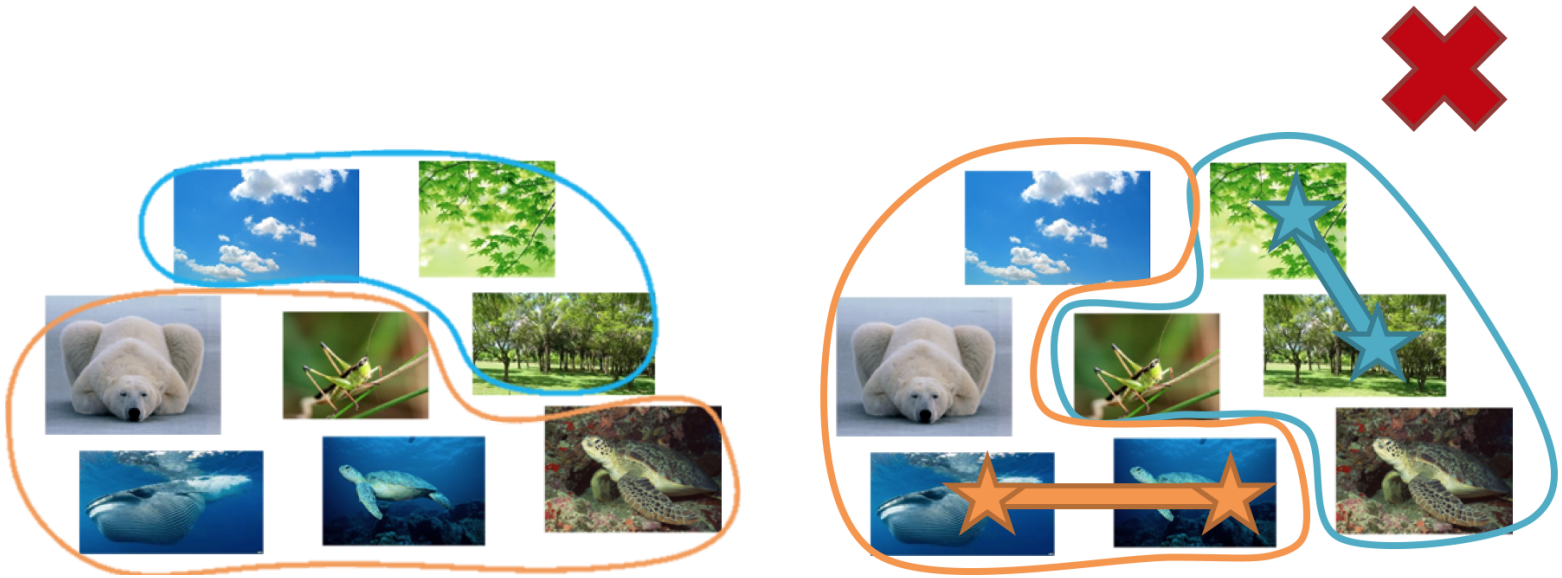
Sampling Bias

- Sampling of pairwise constraints matters:



Sampling Bias

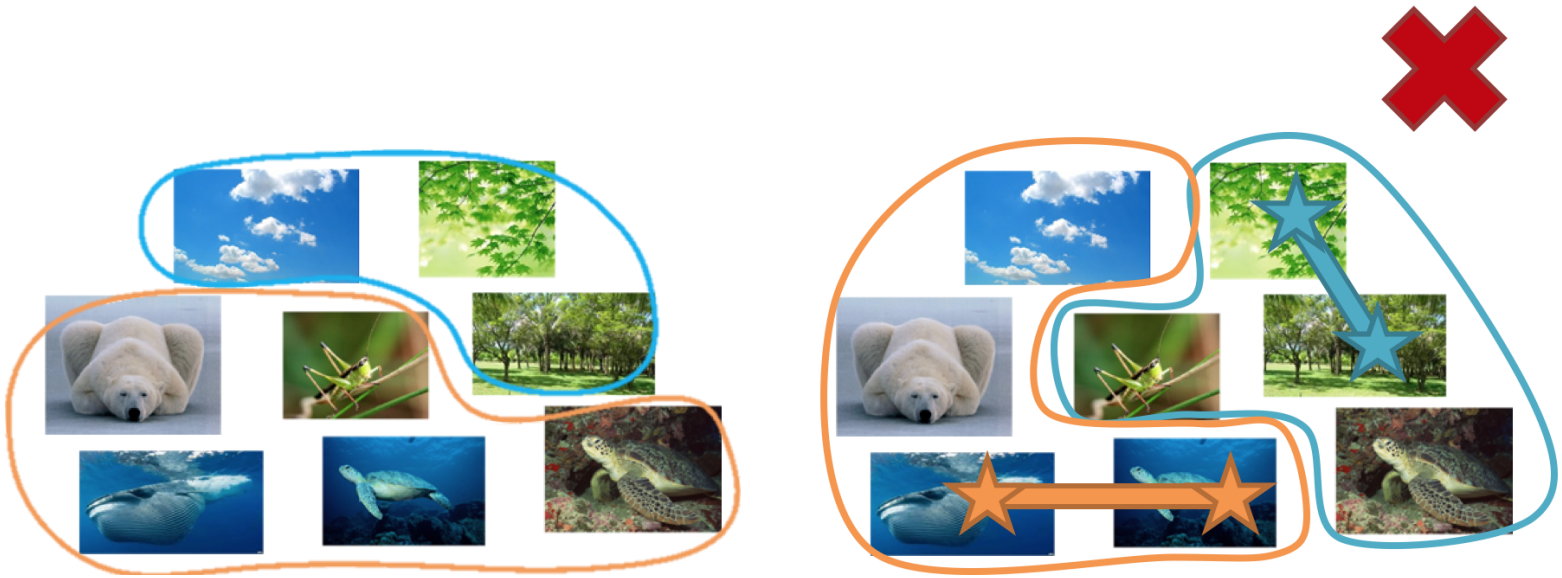
- Sampling of pairwise constraints matters:



- In many applications, the sampling ***cannot*** be uniform

Sampling Bias

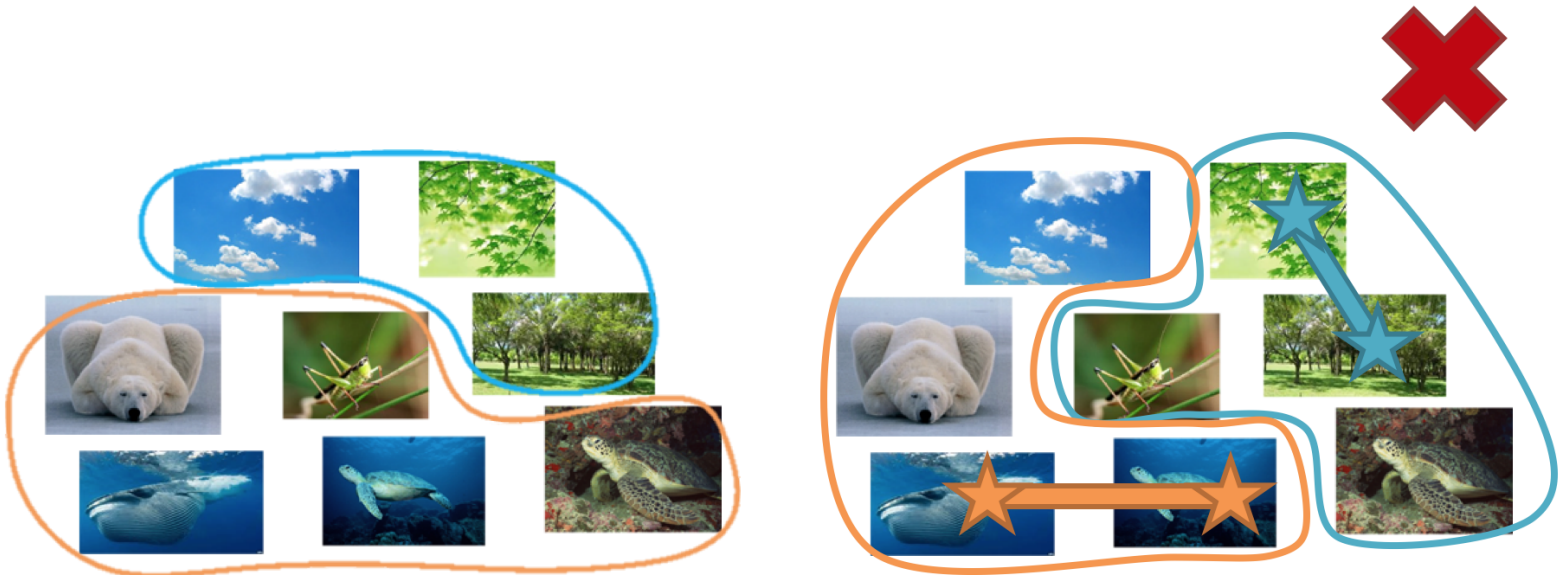
- Sampling of pairwise constraints matters:



- In many applications, the sampling ***cannot*** be uniform
- E.g., suppose we want to cluster products in an e-commerce website
  - Use click-streams provided by the user to get must-links implicitly

Sampling Bias

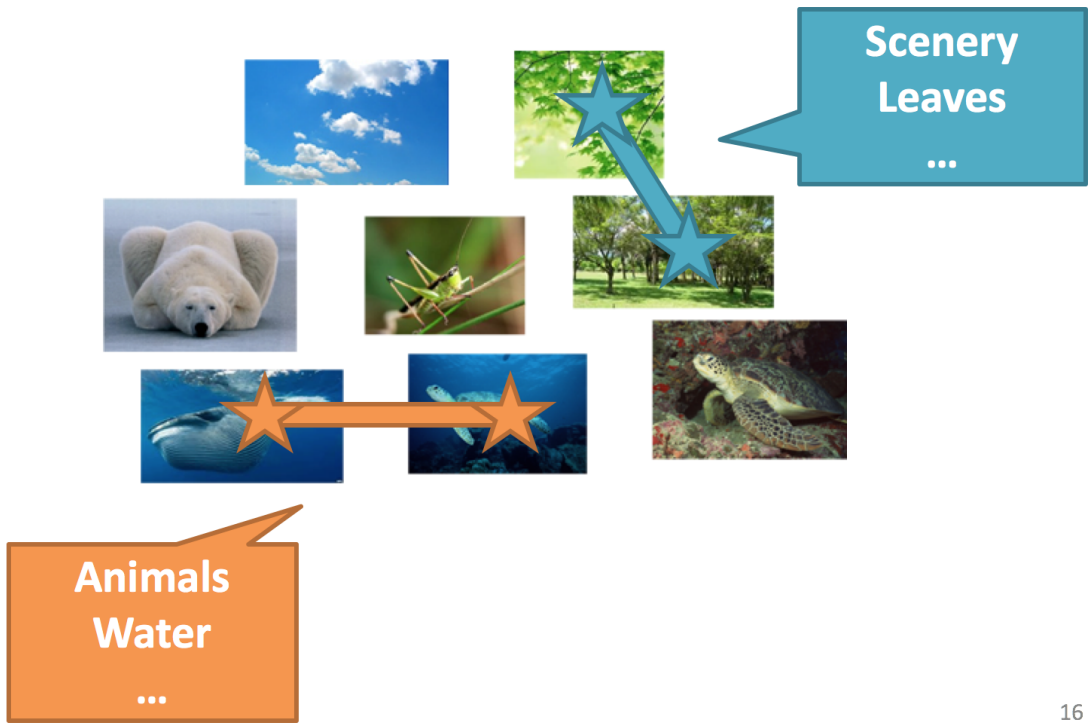
- Sampling of pairwise constraints matters:



- In many applications, the sampling ***cannot*** be uniform
- E.g., suppose we want to cluster products in an e-commerce website
  - Use click-streams provided by the user to get must-links implicitly
- User not likely to click products uniformly
  - Instead, e.g., clicks products with the lowest prices

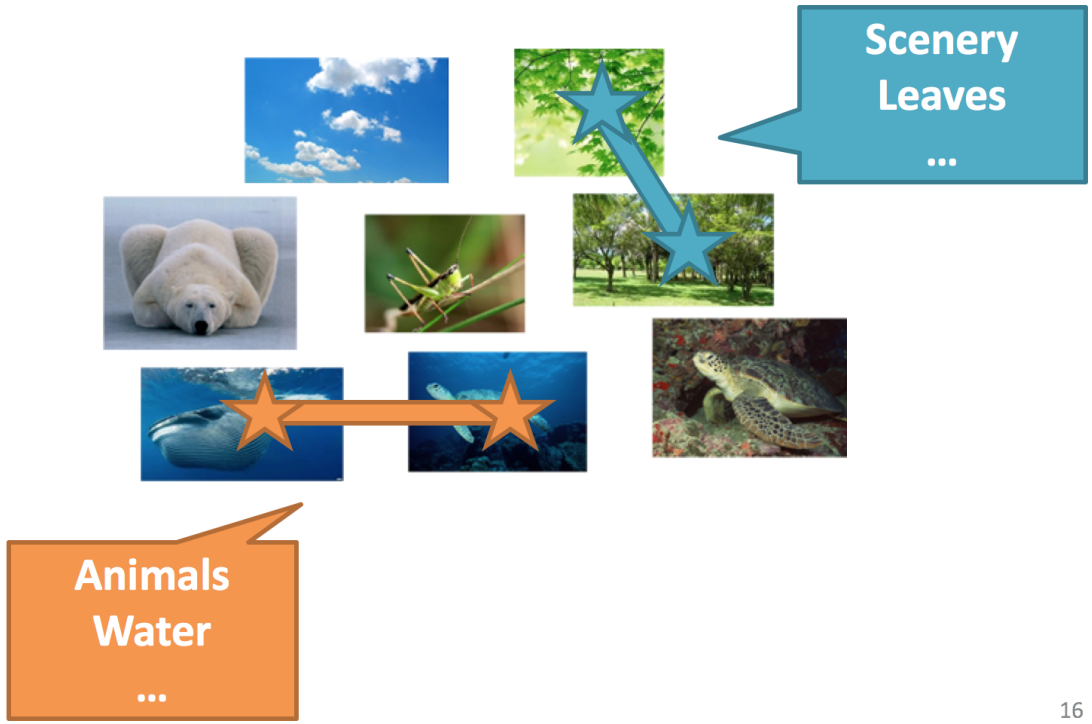
Feature-Level Supervision I

- Side info: perception vectors $\{\boldsymbol{p}^{(n)} \in \mathbb{R}^B\}_{n=1}^N$
  - E.g., bag-of-word vectors of the "reasons" (text) behind must-/cannot-links
  - B the vocabulary size
  - $\boldsymbol{p}^{(n)} \neq \mathbf{0}$ if point n is covered by a must-/cannot-link



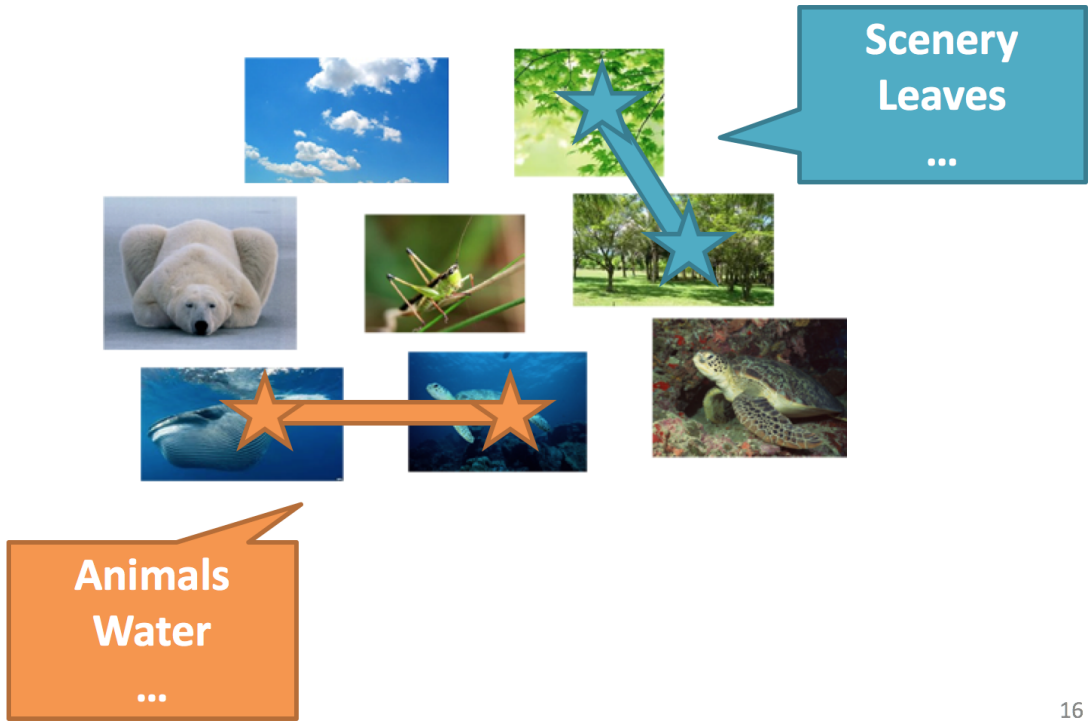
16

Feature-Level Supervision II



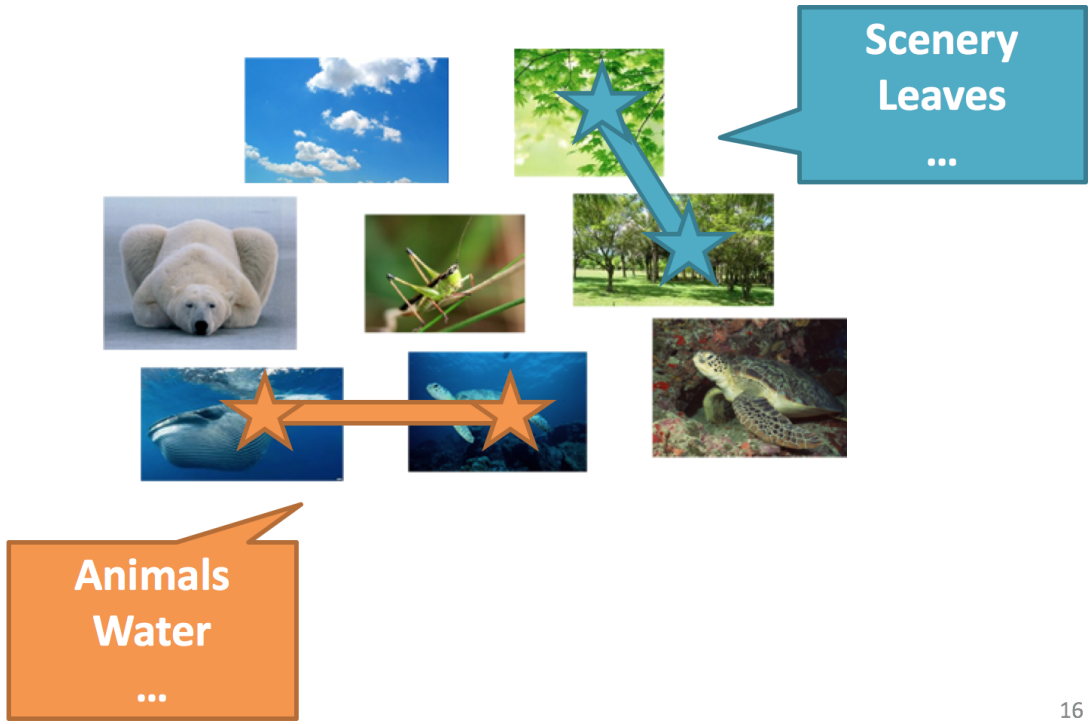
- How to get perception vectors when clustering products in an e-commerce website?

Feature-Level Supervision II



- How to get perception vectors when clustering products in an e-commerce website?
  - Use click-streams provided by the user as must-links
  - Use the ***query*** that triggers clicks as the perception vector

Feature-Level Supervision II



- How to get perception vectors when clustering products in an e-commerce website?
  - Use click-streams provided by the user as must-links
  - Use the ***query*** that triggers clicks as the perception vector
- How to learn form the perception vectors?

Perception-Embedding Clustering

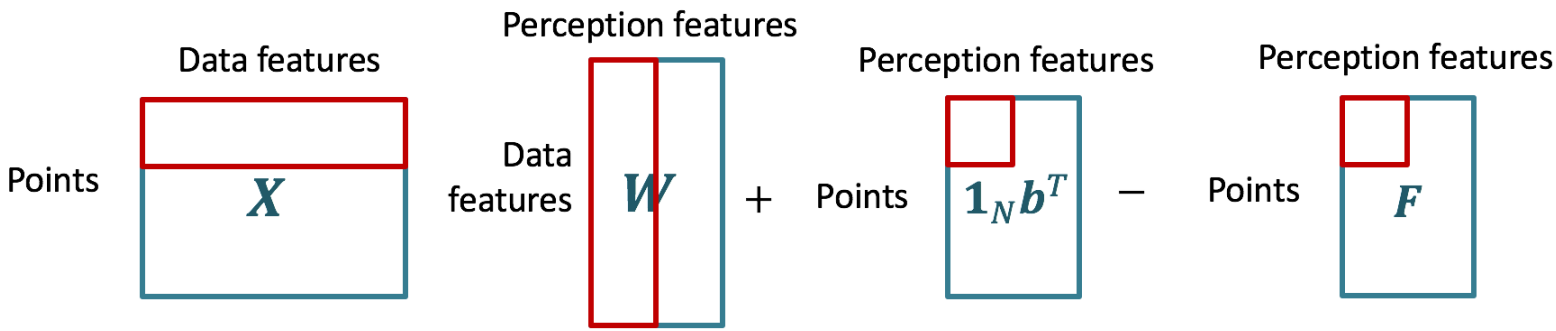
- Perception-embedding clustering [4]: to map every $\boldsymbol{x}^{(n)} \in \mathbb{R}^D$ to a dense $\boldsymbol{f}^{(n)} \in \mathbb{R}^B$ and cluster based on $\boldsymbol{f}^{(n)}$'s

Perception-Embedding Clustering

- Perception-embedding clustering [4]: to map every $\boldsymbol{x}^{(n)} \in \mathbb{R}^D$ to a dense $\boldsymbol{f}^{(n)} \in \mathbb{R}^B$ and cluster based on $\boldsymbol{f}^{(n)}$'s
- Cost function for mapping function:

$$\arg\min_{\boldsymbol{F},\boldsymbol{W},\boldsymbol{b}} \|\boldsymbol{X}\boldsymbol{W} + \boldsymbol{1}_N \boldsymbol{b}^\top - \boldsymbol{F}\|^2 + \lambda \|\boldsymbol{S}(\boldsymbol{F} - \boldsymbol{P})\|^2$$

  - $\boldsymbol{X} \in \mathbb{R}^{N\times D}$, $\boldsymbol{W} \in \mathbb{R}^{D\times B}$, $\boldsymbol{b} \in \mathbb{R}^B$, $\boldsymbol{S} \in \mathbb{R}^{N\times N}$, and $\boldsymbol{F}, \boldsymbol{P} \in \mathbb{R}^{N\times B}$



Perception-Embedding Clustering

- Perception-embedding clustering [4]: to map every $\boldsymbol{x}^{(n)} \in \mathbb{R}^D$ to a dense $\boldsymbol{f}^{(n)} \in \mathbb{R}^B$ and cluster based on $\boldsymbol{f}^{(n)}$'s
- Cost function for mapping function:

$$\arg\min_{\boldsymbol{F},\boldsymbol{W},\boldsymbol{b}} \|\boldsymbol{X}\boldsymbol{W} + \boldsymbol{1}_N \boldsymbol{b}^\top - \boldsymbol{F}\|^2 + \lambda \|\boldsymbol{S}(\boldsymbol{F} - \boldsymbol{P})\|^2$$

  - $\boldsymbol{X} \in \mathbb{R}^{N \times D}$, $\boldsymbol{W} \in \mathbb{R}^{D \times B}$, $\boldsymbol{b} \in \mathbb{R}^B$, $\boldsymbol{S} \in \mathbb{R}^{N \times N}$, and $\boldsymbol{F}, \boldsymbol{P} \in \mathbb{R}^{N \times B}$

1	0	0
0	**0**	0
0	0	**1**

$\boldsymbol{S}$: row selector

$\times$

	Animal	Leaves	Nature
$\boldsymbol{x}_1$	0.85	0.78	0.22
$\boldsymbol{x}_2$	0.75	0.91	0.36
$\boldsymbol{x}_3$	0.15	0.94	0.82

$-$

	Animal	Leaves	Nature
$\boldsymbol{x}_1$	1	1	0
$\boldsymbol{x}_2$	0	0	0
$\boldsymbol{x}_3$	0	1	1

$\boldsymbol{P}$: feature-level supervision

Perception-Embedding Clustering

- Perception-embedding clustering [4]: to map every $\boldsymbol{x}^{(n)} \in \mathbb{R}^D$ to a dense $\boldsymbol{f}^{(n)} \in \mathbb{R}^B$ and cluster based on $\boldsymbol{f}^{(n)}$'s
- Cost function for mapping function:

$$\arg\min_{\boldsymbol{F},\boldsymbol{W},\boldsymbol{b}} \|\boldsymbol{X}\boldsymbol{W} + \boldsymbol{1}_N \boldsymbol{b}^\top - \boldsymbol{F}\|^2 + \lambda \|\boldsymbol{S}(\boldsymbol{F} - \boldsymbol{P})\|^2$$

  - $\boldsymbol{X} \in \mathbb{R}^{N\times D}$, $\boldsymbol{W} \in \mathbb{R}^{D\times B}$, $\boldsymbol{b} \in \mathbb{R}^B$, $\boldsymbol{S} \in \mathbb{R}^{N\times N}$, and $\boldsymbol{F}, \boldsymbol{P} \in \mathbb{R}^{N\times B}$

1	0	0
0	**0**	0
0	0	**1**

$\boldsymbol{S}$: row selector

$\times$

	Animal	Leaves	Nature
$\boldsymbol{x}_1$	0.85	0.78	0.22
$\boldsymbol{x}_2$	0.75	0.91	0.36
$\boldsymbol{x}_3$	0.15	0.94	0.82

$-$

	Animal	Leaves	Nature
$\boldsymbol{x}_1$	1	1	0
$\boldsymbol{x}_2$	0	0	0
$\boldsymbol{x}_3$	0	1	1

$\boldsymbol{P}$: feature-level supervision

- The embedding (parametrized by $\boldsymbol{W}$ and $\boldsymbol{b}$) applies to ***all*** points, thereby avoiding sampling bias

Outline

Transfer Learning

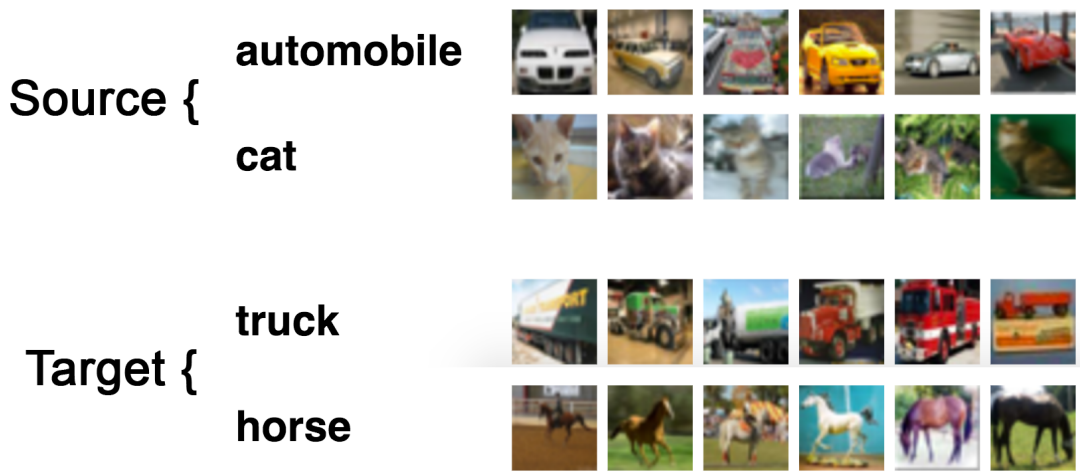
- In practice, we may not have enough data/supervision in $\mathbb{X}$ to generalize well in a task

Transfer Learning

- In practice, we may not have enough data/supervision in $\mathbb{X}$ to generalize well in a task
- Semisupervised learning: to learn from unlabeled data

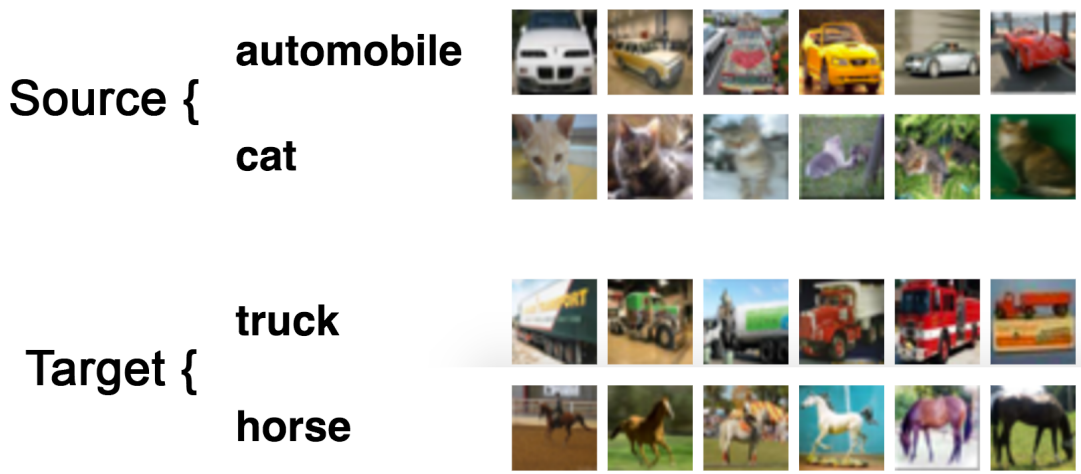
Transfer Learning

- In practice, we may not have enough data/supervision in $\mathbb{X}$ to generalize well in a task
- Semisupervised learning: to learn from unlabeled data
- ***Transfer learning***: to learning from data ***in other domains***



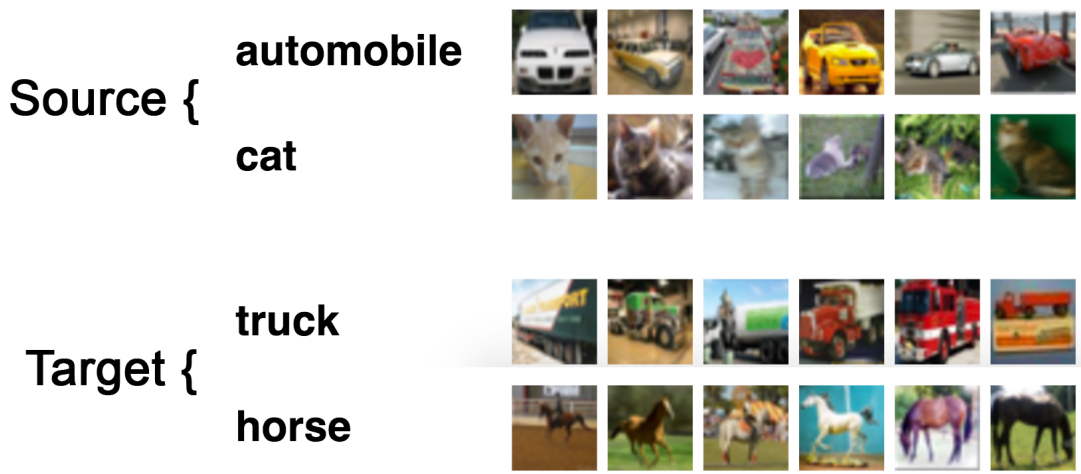
Transfer Learning

- In practice, we may not have enough data/supervision in $\mathbb{X}$ to generalize well in a task
- Semisupervised learning: to learn from unlabeled data
- ***Transfer learning***: to learning from data ***in other domains***
- Define the ***source*** and ***target*** tasks over $\mathbb{X}^{(\text{source})}$ and $\mathbb{X}^{(\text{target})}$



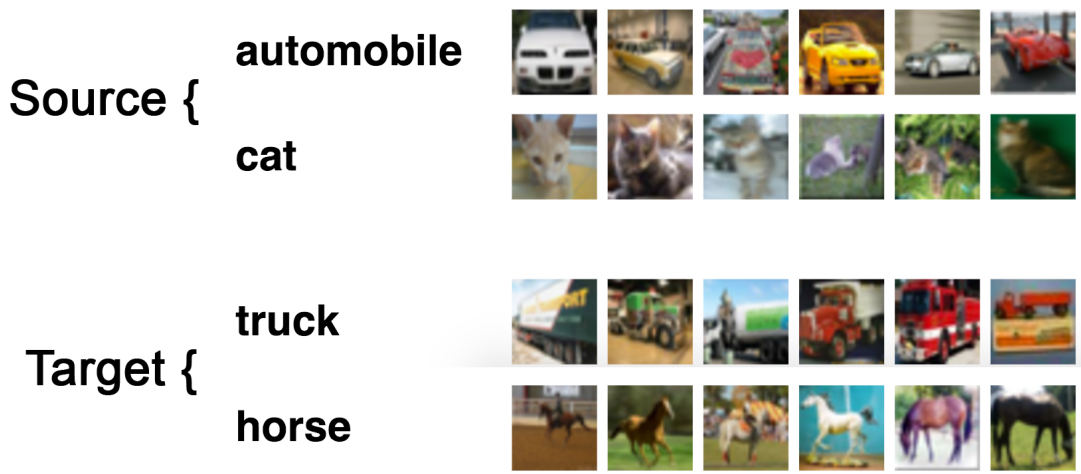
Transfer Learning

- In practice, we may not have enough data/supervision in $\mathbb{X}$ to generalize well in a task
- Semisupervised learning: to learn from unlabeled data
- ***Transfer learning***: to learning from data ***in other domains***
- Define the ***source*** and ***target*** tasks over $\mathbb{X}^{(\text{source})}$ and $\mathbb{X}^{(\text{target})}$
- Goal: use $\mathbb{X}^{(\text{source})}$ to get better results in target task (or vise versa)



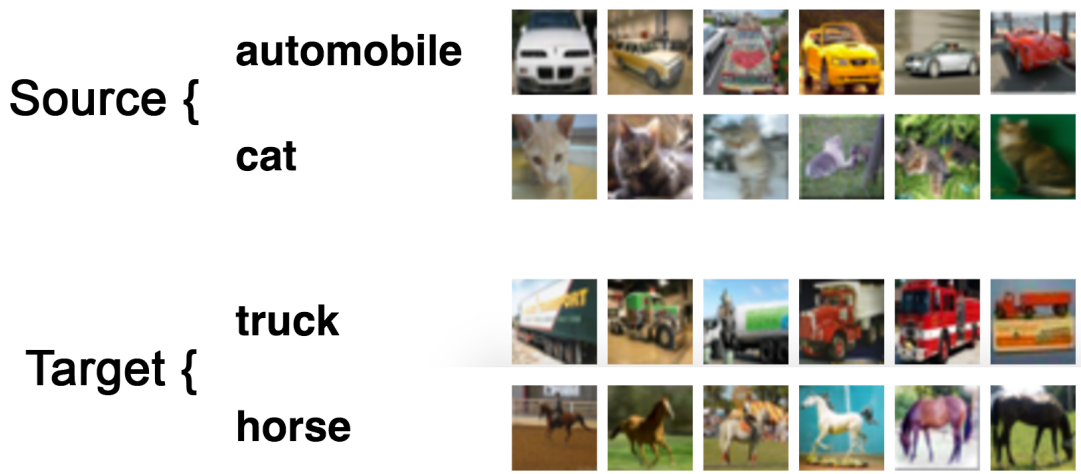
Transfer Learning

- In practice, we may not have enough data/supervision in $\mathbb{X}$ to generalize well in a task
- Semisupervised learning: to learn from unlabeled data
- ***Transfer learning***: to learning from data ***in other domains***
- Define the ***source*** and ***target*** tasks over $\mathbb{X}^{(\text{source})}$ and $\mathbb{X}^{(\text{target})}$
- Goal: use $\mathbb{X}^{(\text{source})}$ to get better results in target task (or vise versa)
- How?

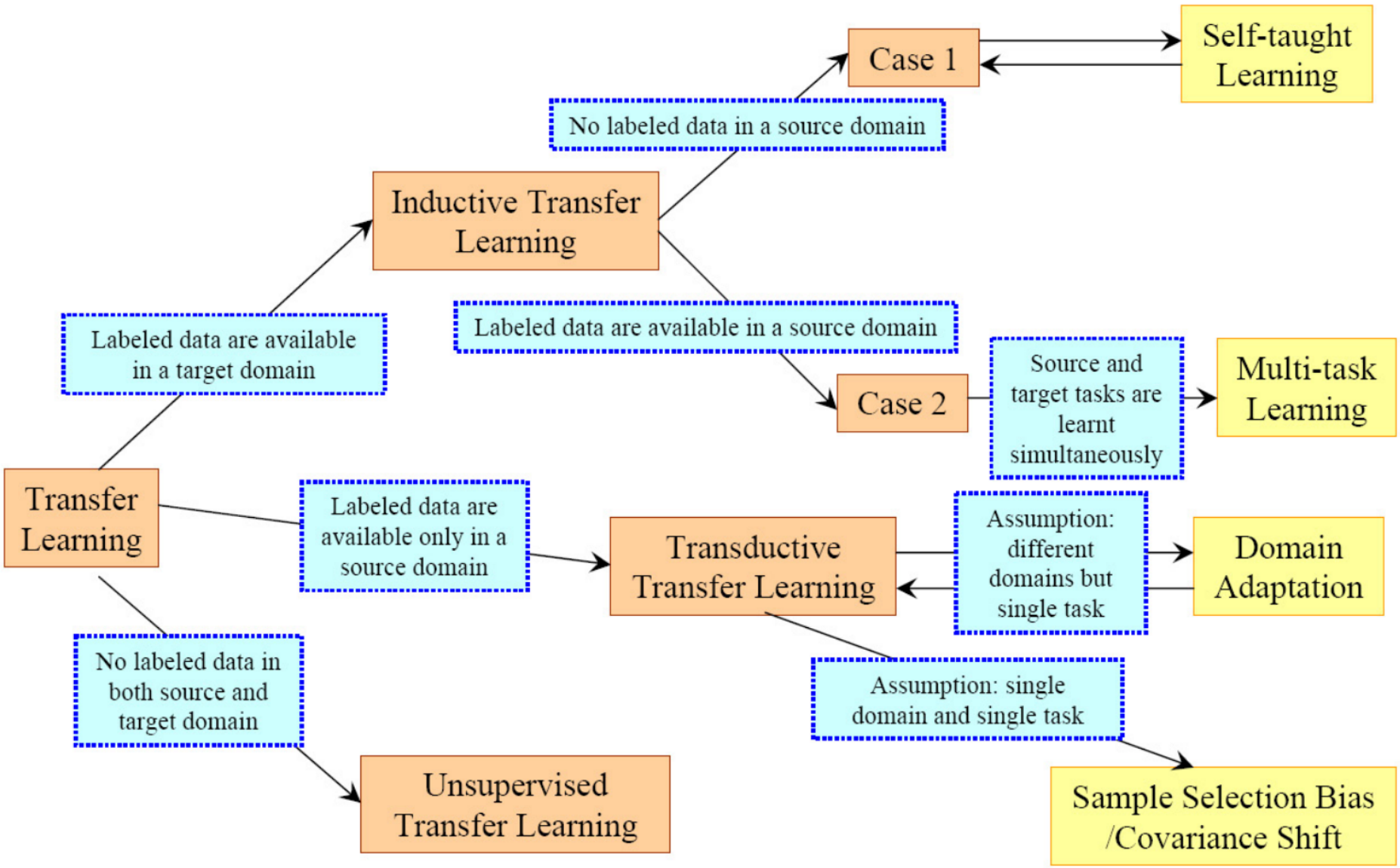


Transfer Learning

- In practice, we may not have enough data/supervision in $\mathbb{X}$ to generalize well in a task
- Semisupervised learning: to learn from unlabeled data
- ***Transfer learning***: to learning from data ***in other domains***
- Define the ***source*** and ***target*** tasks over $\mathbb{X}^{(\text{source})}$ and $\mathbb{X}^{(\text{target})}$
- Goal: use $\mathbb{X}^{(\text{source})}$ to get better results in target task (or vise versa)
- How? To learn "correlations" between $\mathbb{X}^{(\text{source})}$ and $\mathbb{X}^{(\text{target})}$



Branches [10]


Transfer Learning
Labeled data are available in a target domain
Inductive Transfer Learning
No labeled data in a source domain
Case 1
Self-taught Learning
Labeled data are available in a source domain
Case 2
Source and target tasks are learnt simultaneously
Multi-task Learning
Labeled data are available only in a source domain
Transductive Transfer Learning
Assumption: different domains but single task
Domain Adaptation
No labeled data in both source and target domain
Unsupervised Transfer Learning
Assumption: single domain and single task
Sample Selection Bias /Covariance Shift

Few, One, and Zero Shot Learning

- How many data do we need in $\mathbb{X}^{(\text{target})}$ to allow knowledge transfer?

Few, One, and Zero Shot Learning

- How many data do we need in $\mathbb{X}^{(\text{target})}$ to allow knowledge transfer?
- Not many: transfer learning

Few, One, and Zero Shot Learning

- How many data do we need in $\mathbb{X}^{(\text{target})}$ to allow knowledge transfer?
- Not many: transfer learning
- Very few: few shot learning

Few, One, and Zero Shot Learning

- How many data do we need in $\mathbb{X}^{(\text{target})}$ to allow knowledge transfer?
- Not many: transfer learning
- Very few: few shot learning
- Only 1: one shot learning

Few, One, and Zero Shot Learning

- How many data do we need in $\mathbb{X}^{(\text{target})}$ to allow knowledge transfer?
- Not many: transfer learning
- Very few: few shot learning
- Only 1: one shot learning
- None: ***zero shot learning***

Few, One, and Zero Shot Learning

- How many data do we need in $\mathbb{X}^{(\text{target})}$ to allow knowledge transfer?
- Not many: transfer learning
- Very few: few shot learning
- Only 1: one shot learning
- None: ***zero shot learning*** (How is that possible?)

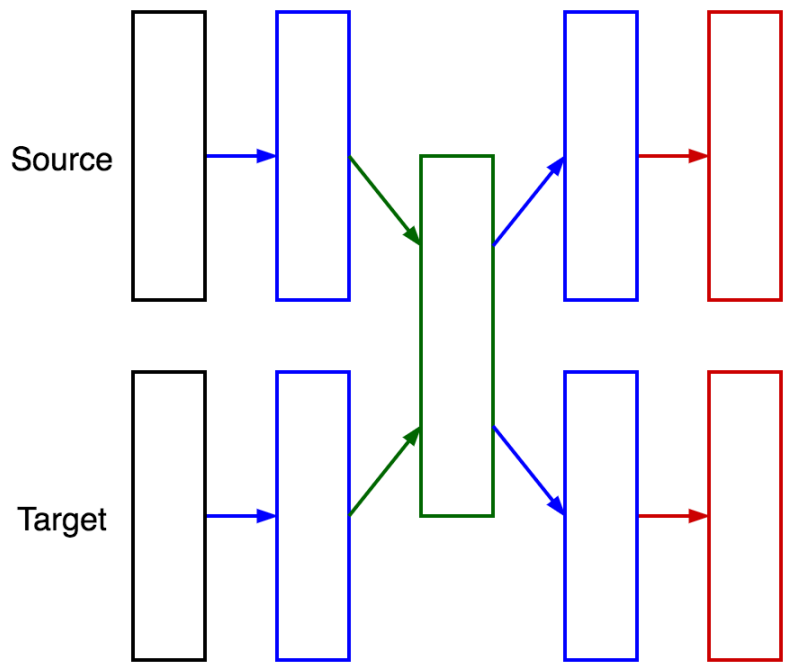
Outline

Multitask Learning

- To jointly learn the source and target models
  - Both $\mathbb{X}^{(\text{source})}$ and $\mathbb{X}^{(\text{target})}$ have labels

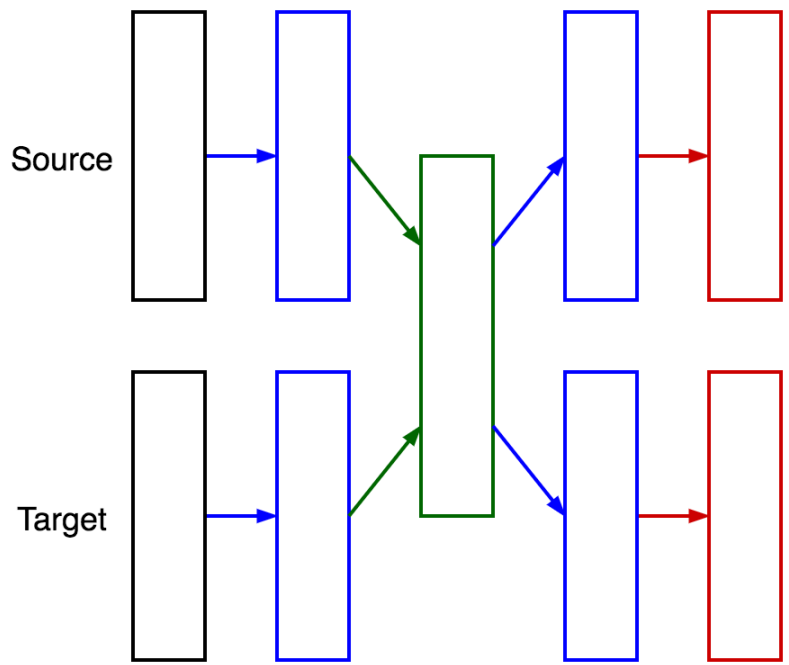
Multitask Learning

- To jointly learn the source and target models
  - Both $\mathbb{X}^{(\text{source})}$ and $\mathbb{X}^{(\text{target})}$ have labels
- Models ***share weights*** that capture the correlation between the data/tasks



Multitask Learning

- To jointly learn the source and target models
  - Both $\mathbb{X}^{(\text{source})}$ and $\mathbb{X}^{(\text{target})}$ have labels
- Models ***share weights*** that capture the correlation between the data/tasks
- Which layers to share in deep NNs?



Weight Sharing

- Application dependent, e.g.,

Weight Sharing

- Application dependent, e.g.,
- Shallow layers in image object recognition
  - To share filters/feature detectors

Weight Sharing

- Application dependent, e.g.,
- Shallow layers in image object recognition
  - To share filters/feature detectors
- Deep layers in speech transcription
  - To share the word map

Weight Initiation

- One simpler way to transfer knowledge is to ***initiate weights*** of target model to those of source model

Weight Initiation

- One simpler way to transfer knowledge is to ***initiate weights*** of target model to those of source model
- Very common in deep learning
  - Training a CNN over ImageNet [5] may take a week

Weight Initiation

- One simpler way to transfer knowledge is to ***initiate weights*** of target model to those of source model
- Very common in deep learning
  - Training a CNN over ImageNet [5] may take a week
  - Many pre-trained NNs on Internet, e.g., ▸ Model Zoo

Weight Initiation

- One simpler way to transfer knowledge is to ***initiate weights*** of target model to those of source model
- Very common in deep learning
  - Training a CNN over ImageNet [5] may take a week
  - Many pre-trained NNs on Internet, e.g., ▸ Model Zoo
- A ***regularization*** technique rather than an optimization technique [3]

Weight Initiation

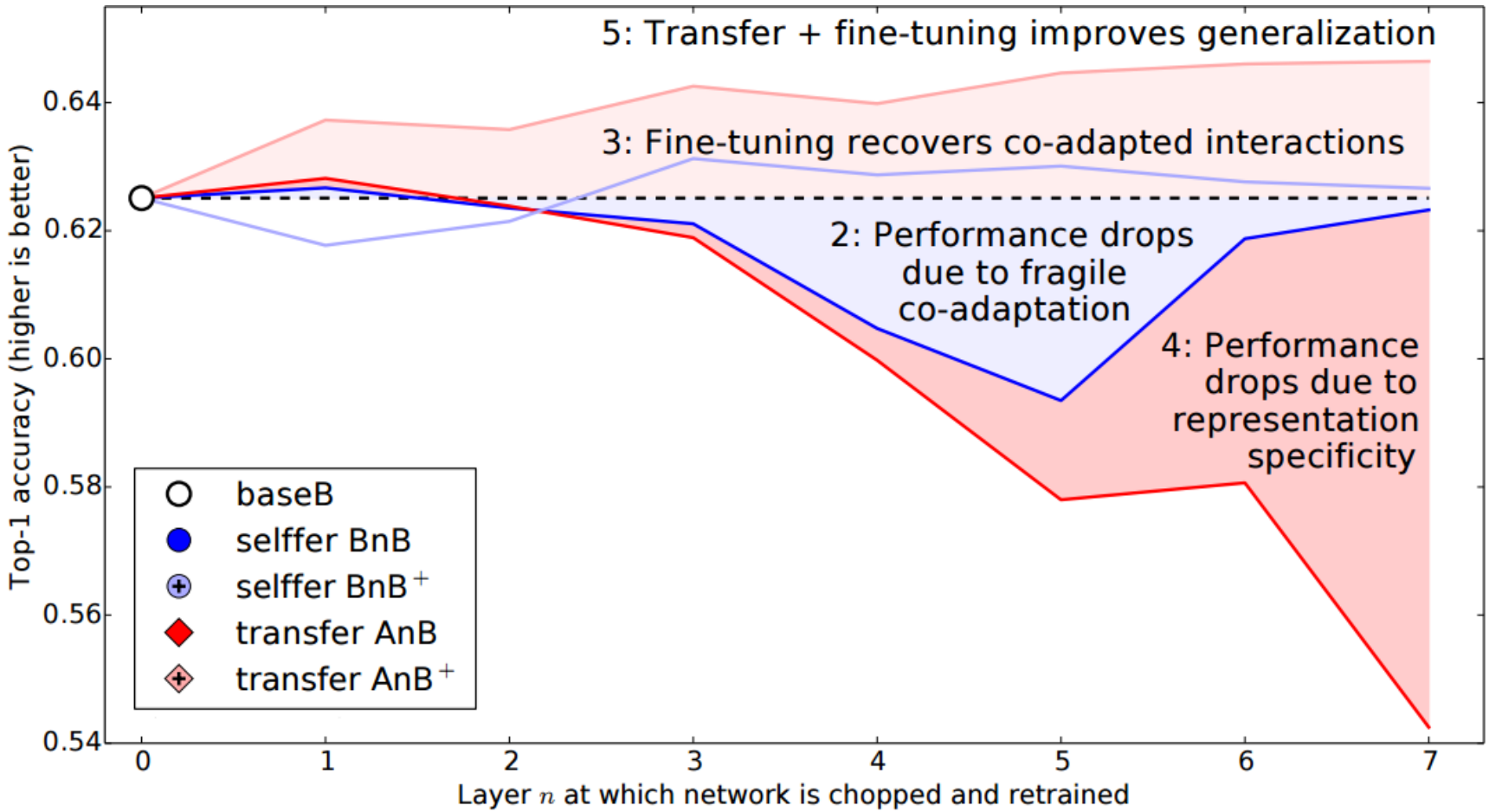
- One simpler way to transfer knowledge is to ***initiate weights*** of target model to those of source model
- Very common in deep learning
  - Training a CNN over ImageNet [5] may take a week
  - Many pre-trained NNs on Internet, e.g., ▸ Model Zoo
- A ***regularization*** technique rather than an optimization technique [3]
- Which weights to borrow from also depends on applications

Fine-Tuning I

- In addition to borrowing weights, we may update (***fine-tune***) the weights when training the target model

Fine-Tuning I

- In addition to borrowing weights, we may update (***fine-tune***) the weights when training the target model
- Results from 2 CNNs (A and B) over ImageNet [14]:



Fine-Tuning II

Caution

Fine tuning does ***not*** always help!

Fine-Tuning II

Caution

Fine tuning does ***not*** always help!

- Fine-tuning or not?

Fine-Tuning II

Caution

Fine tuning does ***not*** always help!

- Fine-tuning or not?
- Large $\mathbb{X}^{(\text{target})}$, similar $\mathbb{X}^{(\text{source})}$:
- Large $\mathbb{X}^{(\text{target})}$, different $\mathbb{X}^{(\text{source})}$:
- Small $\mathbb{X}^{(\text{target})}$, similar $\mathbb{X}^{(\text{source})}$:
- Small $\mathbb{X}^{(\text{target})}$, different $\mathbb{X}^{(\text{source})}$:

Fine-Tuning II

Caution

Fine tuning does ***not*** always help!

- Fine-tuning or not?
- Large $\mathbb{X}^{(\text{target})}$, similar $\mathbb{X}^{(\text{source})}$: Yes
- Large $\mathbb{X}^{(\text{target})}$, different $\mathbb{X}^{(\text{source})}$:
- Small $\mathbb{X}^{(\text{target})}$, similar $\mathbb{X}^{(\text{source})}$:
- Small $\mathbb{X}^{(\text{target})}$, different $\mathbb{X}^{(\text{source})}$:

Fine-Tuning II

Caution

Fine tuning does ***not*** always help!

- Fine-tuning or not?
- Large $\mathbb{X}^{(\text{target})}$, similar $\mathbb{X}^{(\text{source})}$: Yes
- Large $\mathbb{X}^{(\text{target})}$, different $\mathbb{X}^{(\text{source})}$: Yes (often still beneficial in practice)
- Small $\mathbb{X}^{(\text{target})}$, similar $\mathbb{X}^{(\text{source})}$:
- Small $\mathbb{X}^{(\text{target})}$, different $\mathbb{X}^{(\text{source})}$:

Fine-Tuning II

Caution

Fine tuning does ***not*** always help!

- Fine-tuning or not?
- Large $\mathbb{X}^{(\text{target})}$, similar $\mathbb{X}^{(\text{source})}$: Yes
- Large $\mathbb{X}^{(\text{target})}$, different $\mathbb{X}^{(\text{source})}$: Yes (often still beneficial in practice)
- Small $\mathbb{X}^{(\text{target})}$, similar $\mathbb{X}^{(\text{source})}$: No (to avoid overfitting)
- Small $\mathbb{X}^{(\text{target})}$, different $\mathbb{X}^{(\text{source})}$:

Fine-Tuning II

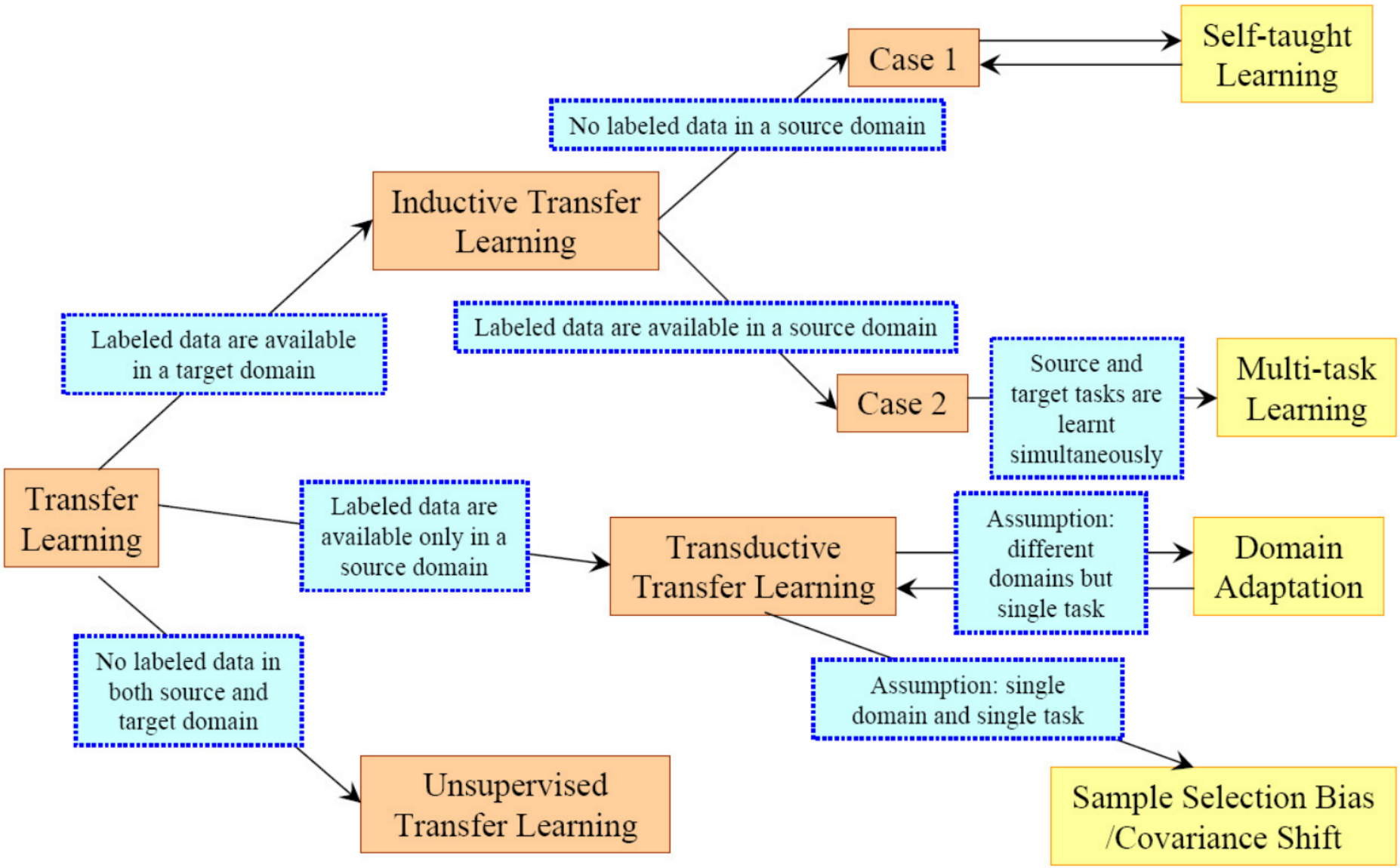
Caution

Fine tuning does ***not*** always help!

- Fine-tuning or not?
- Large $\mathbb{X}^{(\text{target})}$, similar $\mathbb{X}^{(\text{source})}$: Yes
- Large $\mathbb{X}^{(\text{target})}$, different $\mathbb{X}^{(\text{source})}$: Yes (often still beneficial in practice)
- Small $\mathbb{X}^{(\text{target})}$, similar $\mathbb{X}^{(\text{source})}$: No (to avoid overfitting)
- Small $\mathbb{X}^{(\text{target})}$, different $\mathbb{X}^{(\text{source})}$: No
  - Instead prepend/append ***simple weight rewriter*** (e.g., linear SVM)

Outline

Domain Adaptation

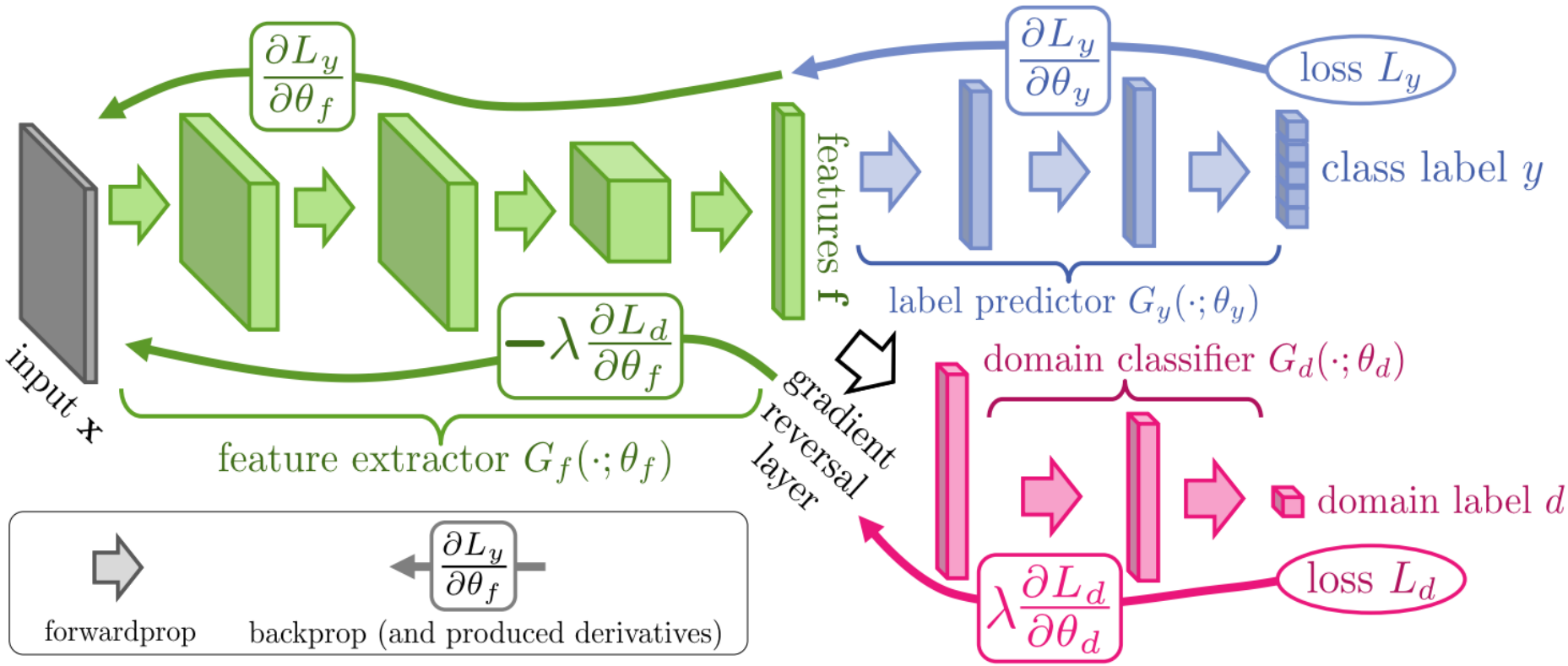


Domain Adversarial Networks

- Goal: to learn ***domain-invariant features*** that help source model adapt to target task

Domain Adversarial Networks

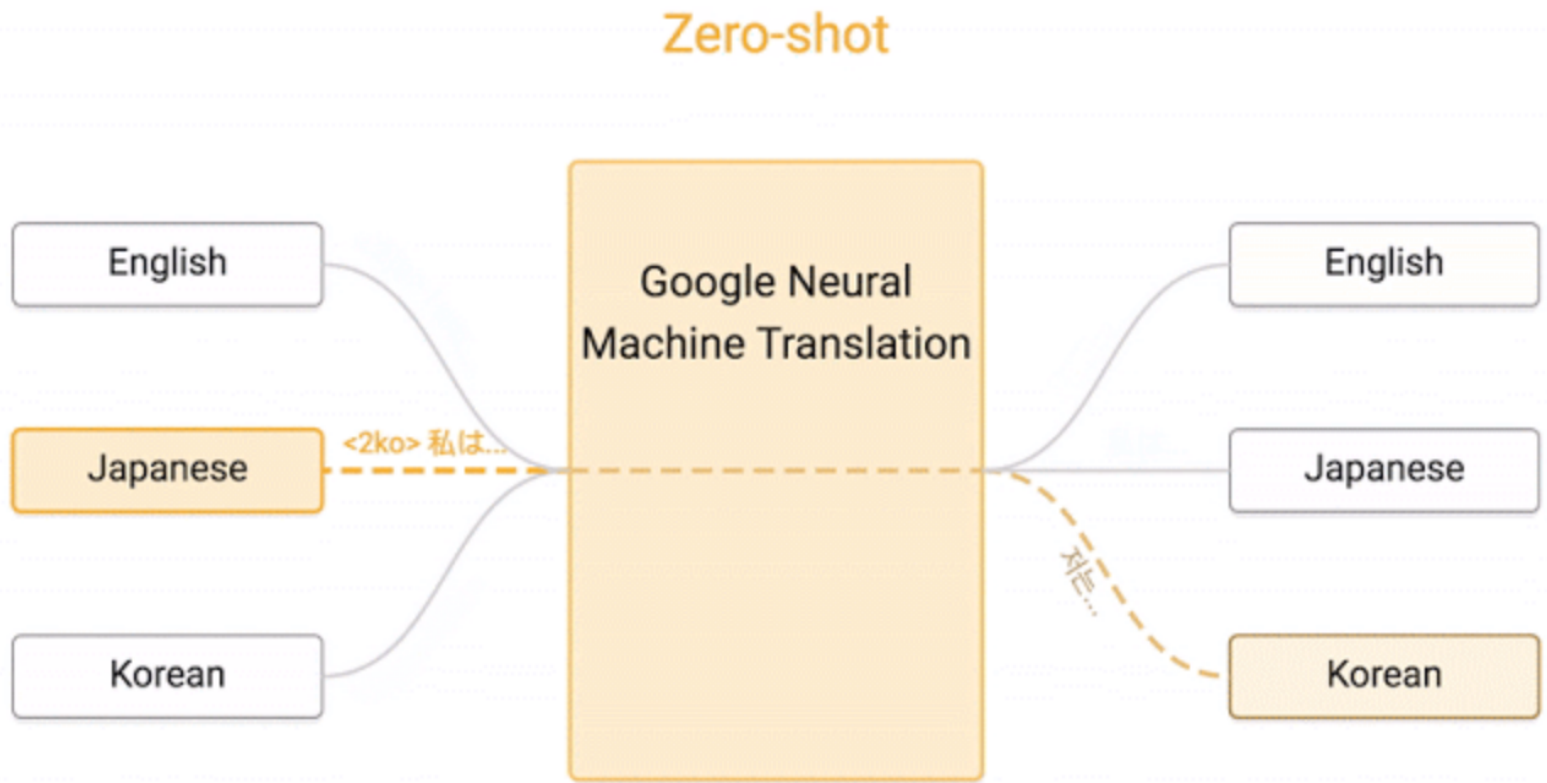
- Goal: to learn ***domain-invariant features*** that help source model adapt to target task
- Domain classifier + gradient reversal layer [7]



Outline

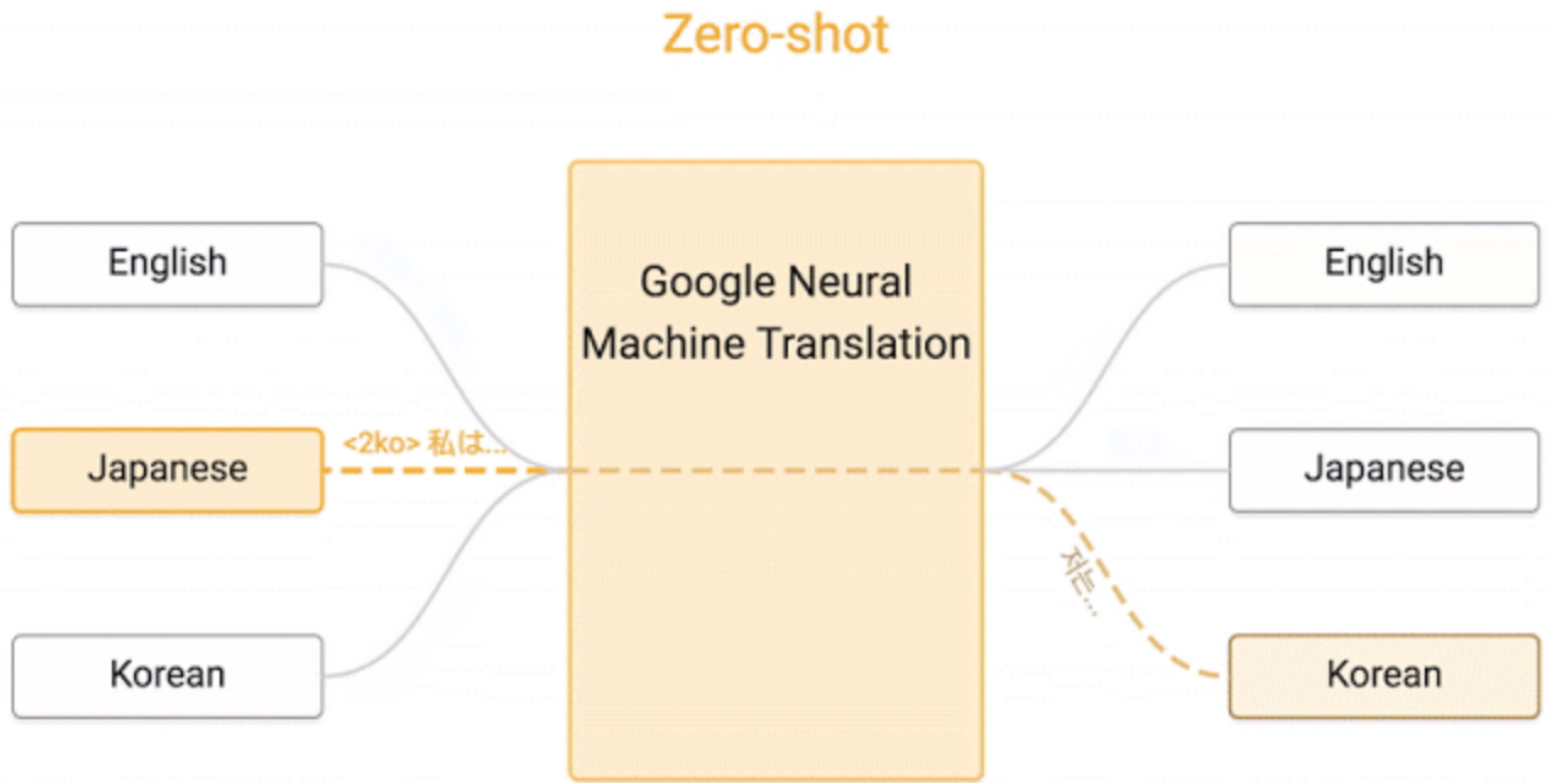
Zero Shot Learning

- Zero shot learning: transfer learning with $\mathbb{X}^{(\text{source})}$ and ***empty*** $\mathbb{X}^{(\text{target})}$



Zero Shot Learning

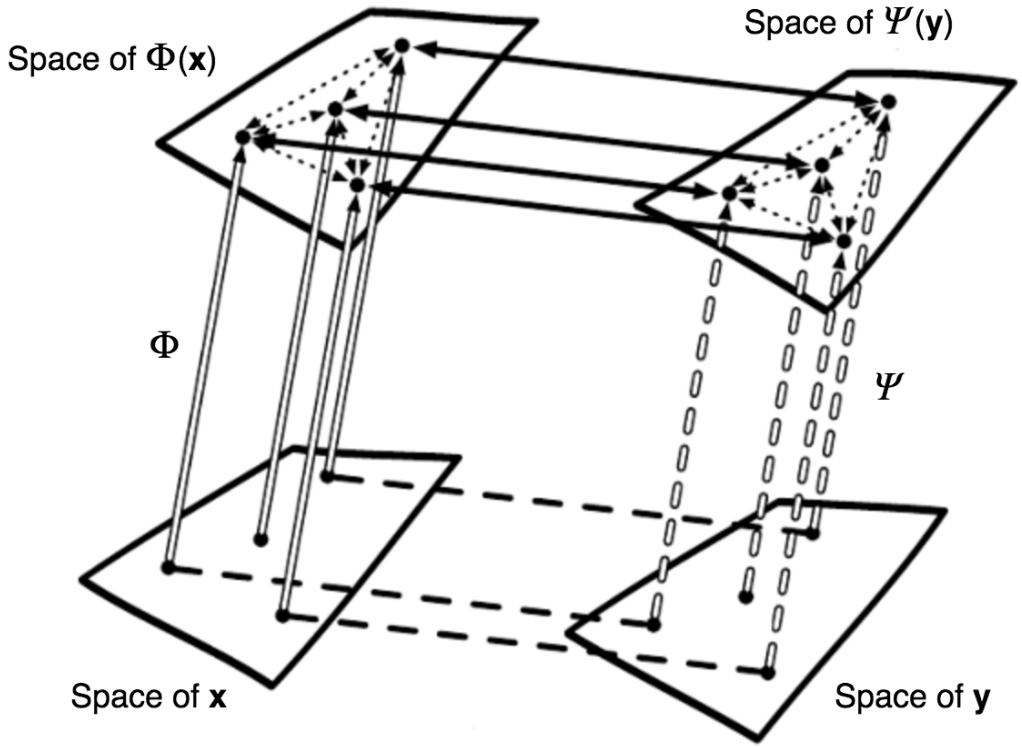
- Zero shot learning: transfer learning with $\mathbb{X}^{(\text{source})}$ and ***empty*** $\mathbb{X}^{(\text{target})}$



- How is that possible?

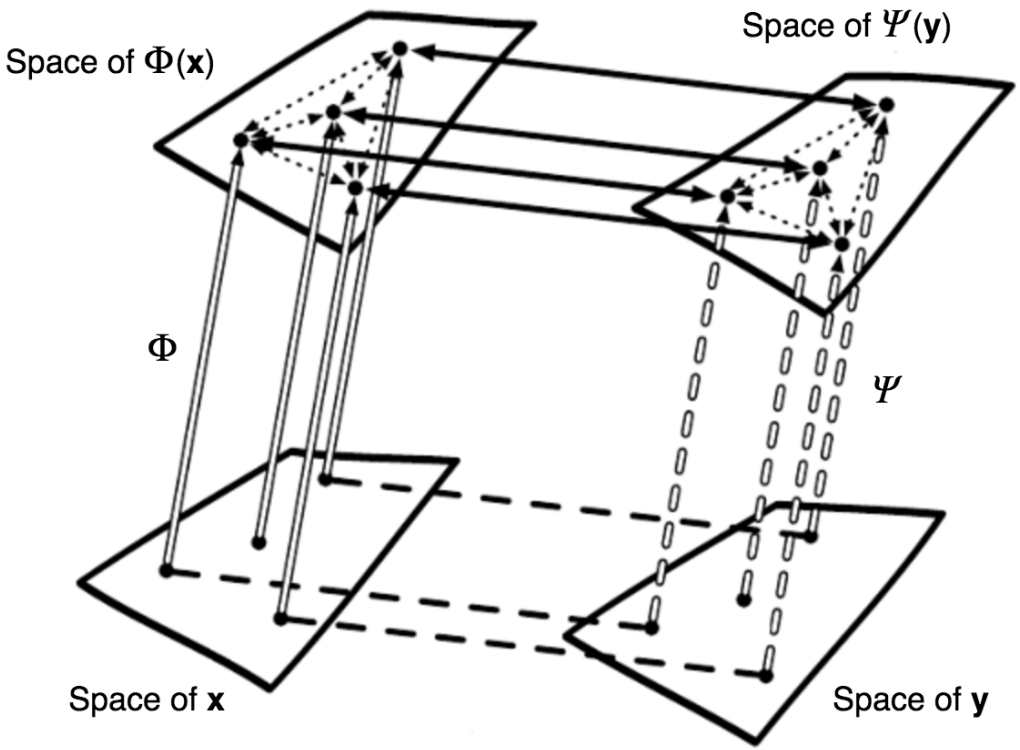
Label Representations

- Side information: the ***semantic representations*** $\Psi(\mathbf{y})$ of labels
  - E.g., "has paws," "has stripes," or "is black" for the "animal" class



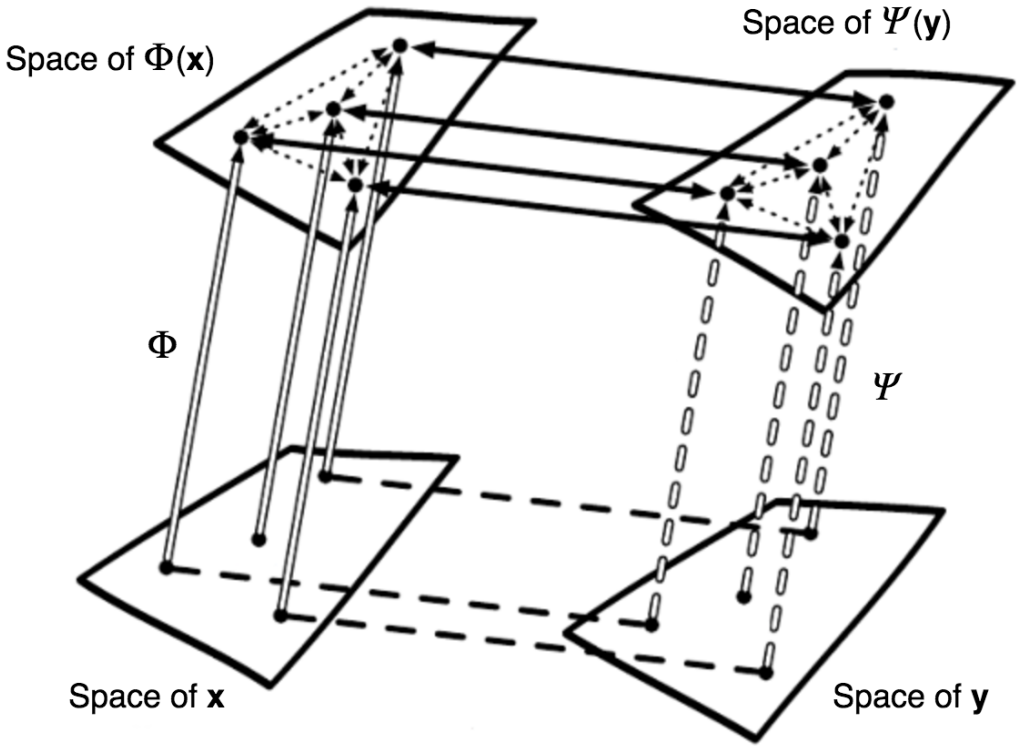
Label Representations

- Side information: the ***semantic representations*** $\Psi(\mathbf{y})$ of labels
  - E.g., "has paws," "has stripes," or "is black" for the "animal" class
- Assume that labels in different domains share the same semantic space



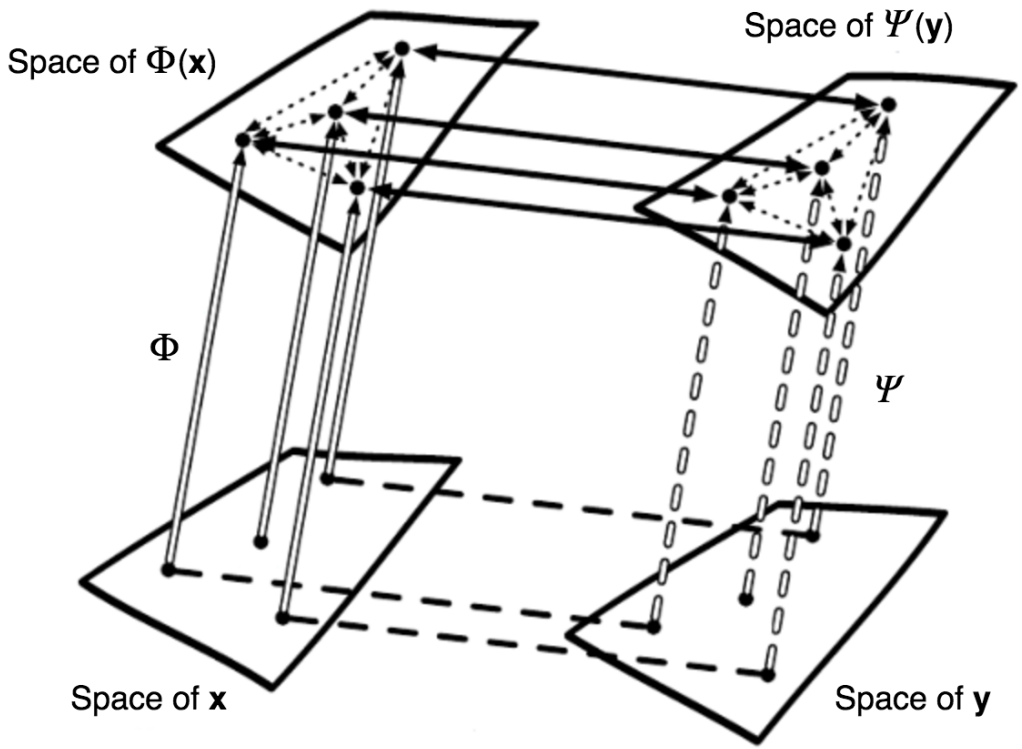
Label Representations

- Side information: the ***semantic representations*** $\Psi(\mathbf{y})$ of labels
  - E.g., "has paws," "has stripes," or "is black" for the "animal" class
- Assume that labels in different domains share the same semantic space
- Embedding function Ψ can be learned
  - jointedly with the model (e.g., in Google Neural Machine Translation)
  - or separately (e.g., in [1])



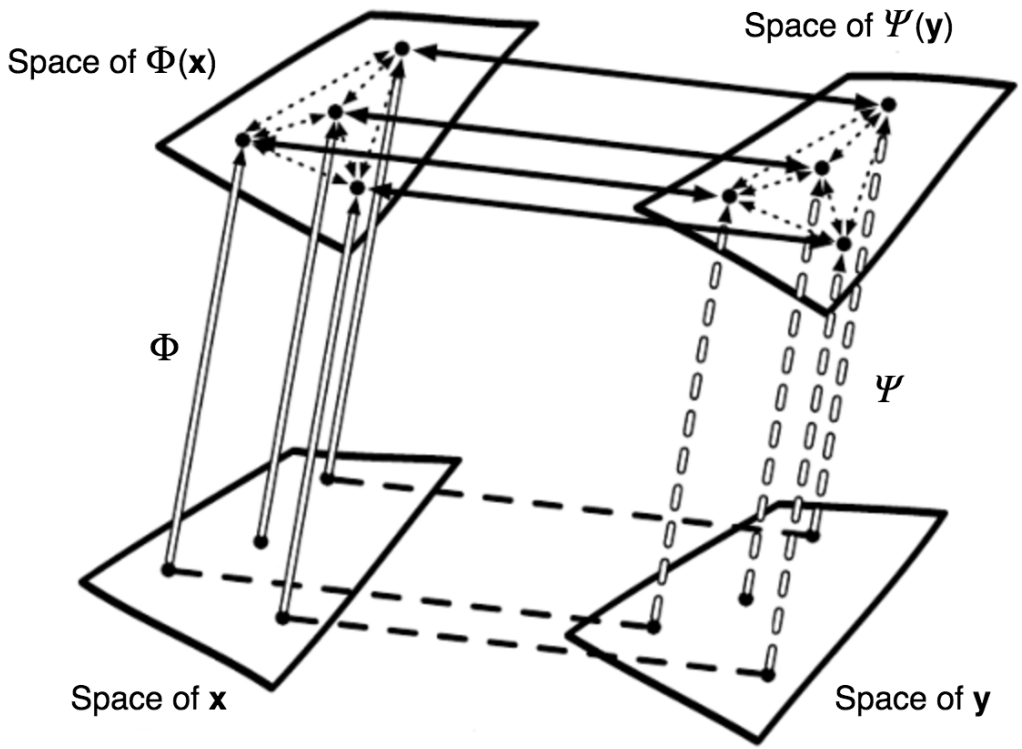
Why Does Zero Shot Learning Work?

- In task A, a model uses labeled pairs $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$'s to learn the map between spaces of $\Phi(\mathbf{x})$ and $\Psi(\mathbf{y})$



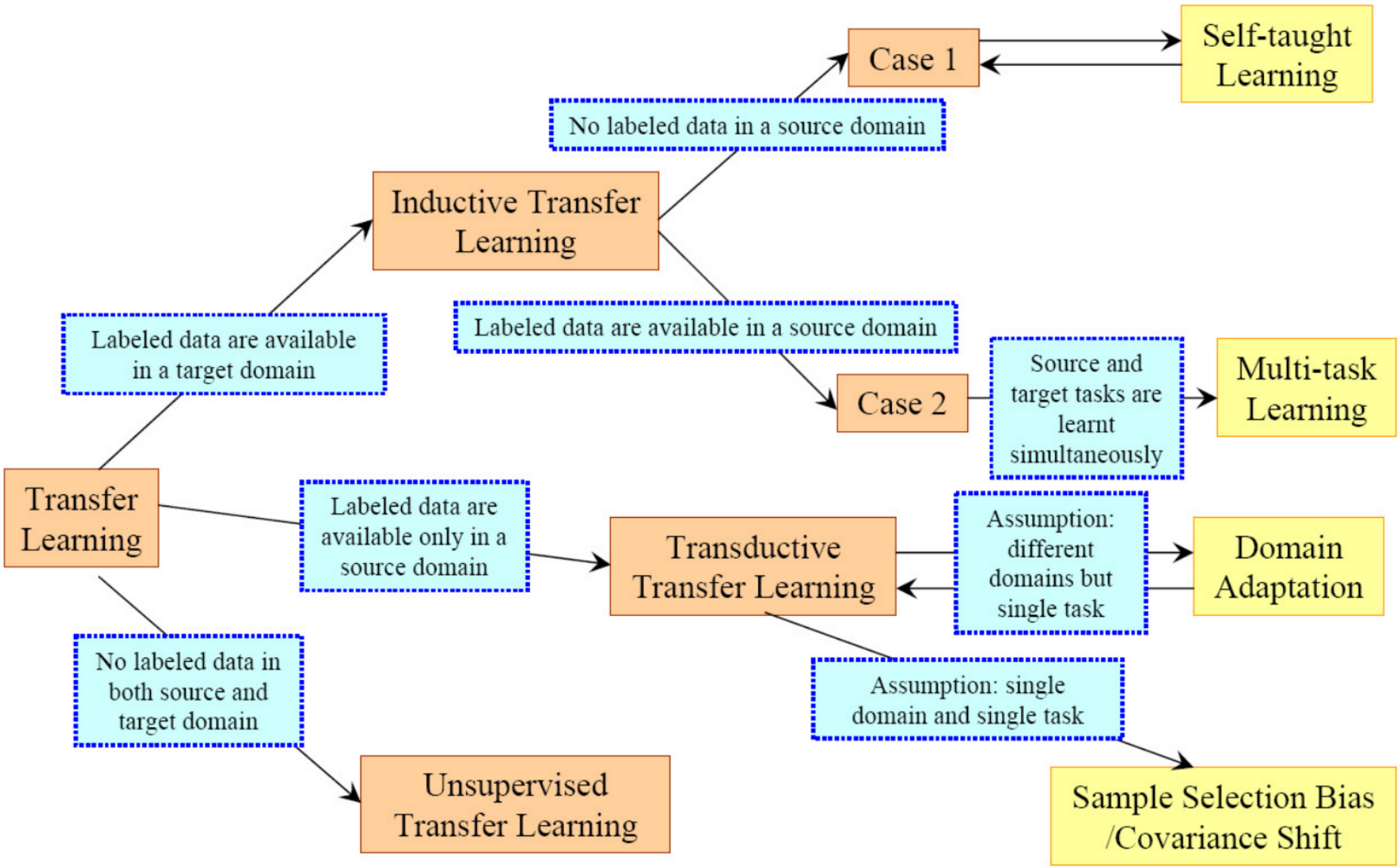
Why Does Zero Shot Learning Work?

- In task A, a model uses labeled pairs $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$'s to learn the map between spaces of $\Phi(\mathbf{x})$ and $\Psi(\mathbf{y})$
- In task B (with zero shot), the model predicts label of point $\boldsymbol{x}'$ by
  1. First obtaining $\Phi(\boldsymbol{x}')$
  2. Then following the map to find out $\Psi(\boldsymbol{y}')$



Outline

Unsupervised TL



Cross-Domain Recommendation

- Data (rating matrix) in a domain may be too sparse
  - Cannot be factorized well

Cross-Domain Recommendation

- Data (rating matrix) in a domain may be too sparse
  - Cannot be factorized well
- Can we exploit rating matrices from other domains?

$X^{(source)}$ (Movies)

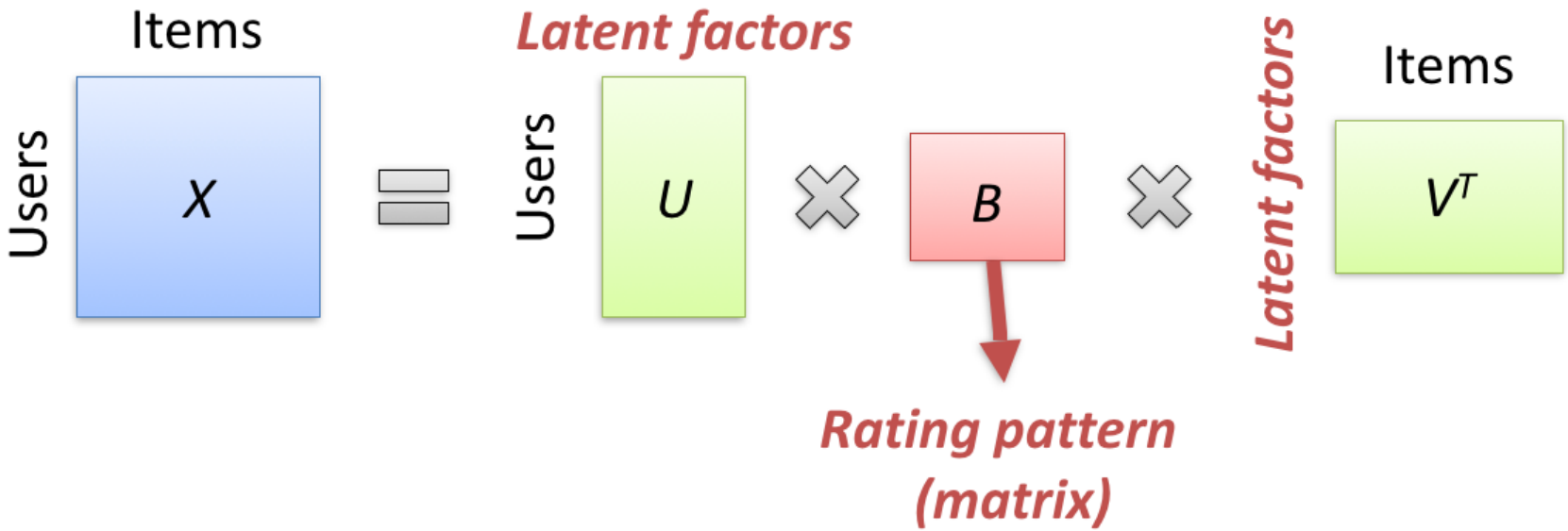
?	?	?	3	?
?	5	?	1	?
?	2	?	?	?
?	?	?	1	?

$X^{(target)}$ (Books)

?	?	?	?	?
?	?	2	?	?
5	?	?	?	3
?	?	1	?	?

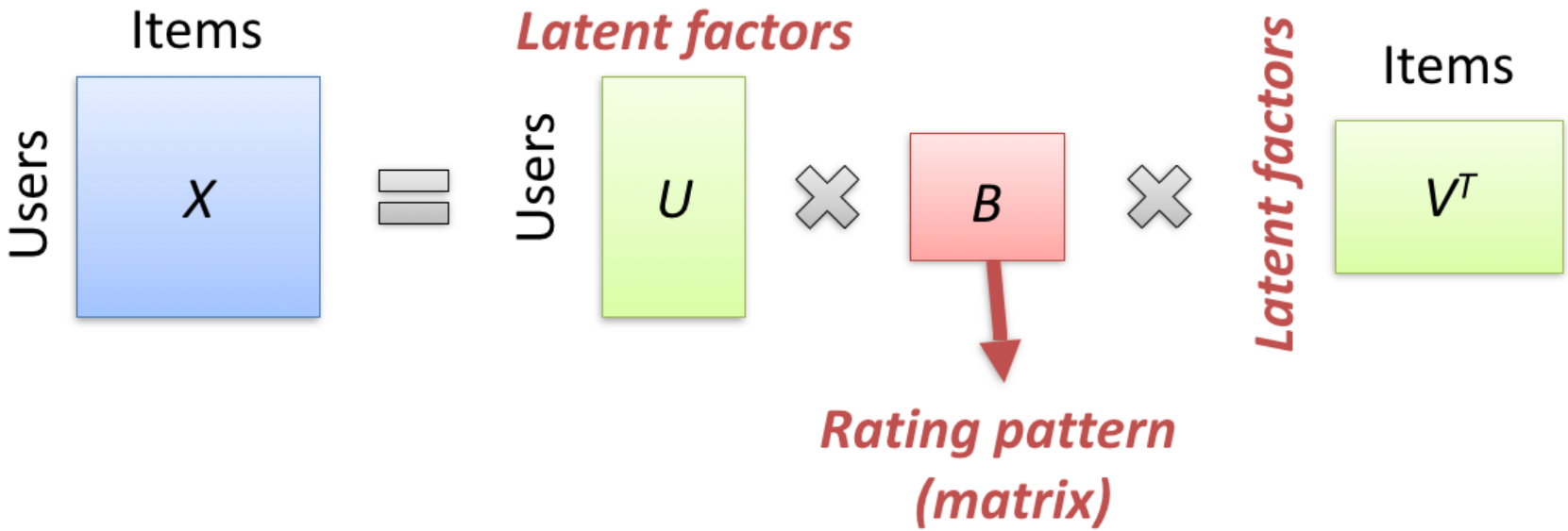
Nonnegative Matrix Tri-Factorization

$$\arg\min_{\boldsymbol{U},\boldsymbol{B},\boldsymbol{V}>\boldsymbol{O}} \|\boldsymbol{X}-\boldsymbol{U}\boldsymbol{B}\boldsymbol{V}^\top\|_F^2$$



Nonnegative Matrix Tri-Factorization

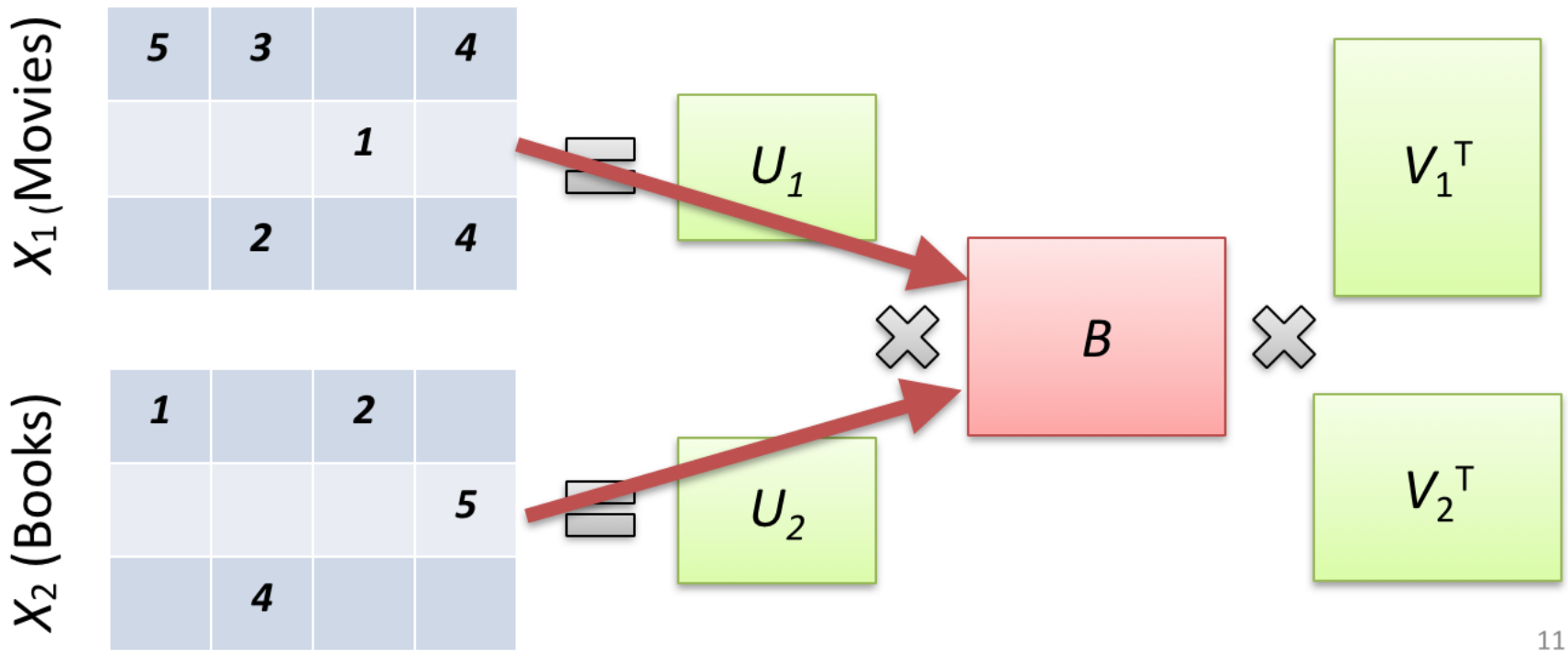
$$\arg \min_{\boldsymbol{U},\boldsymbol{B},\boldsymbol{V}>\boldsymbol{O}} \|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{B}\boldsymbol{V}^\top\|_F^2$$



- Has a clustering interpretation [6]

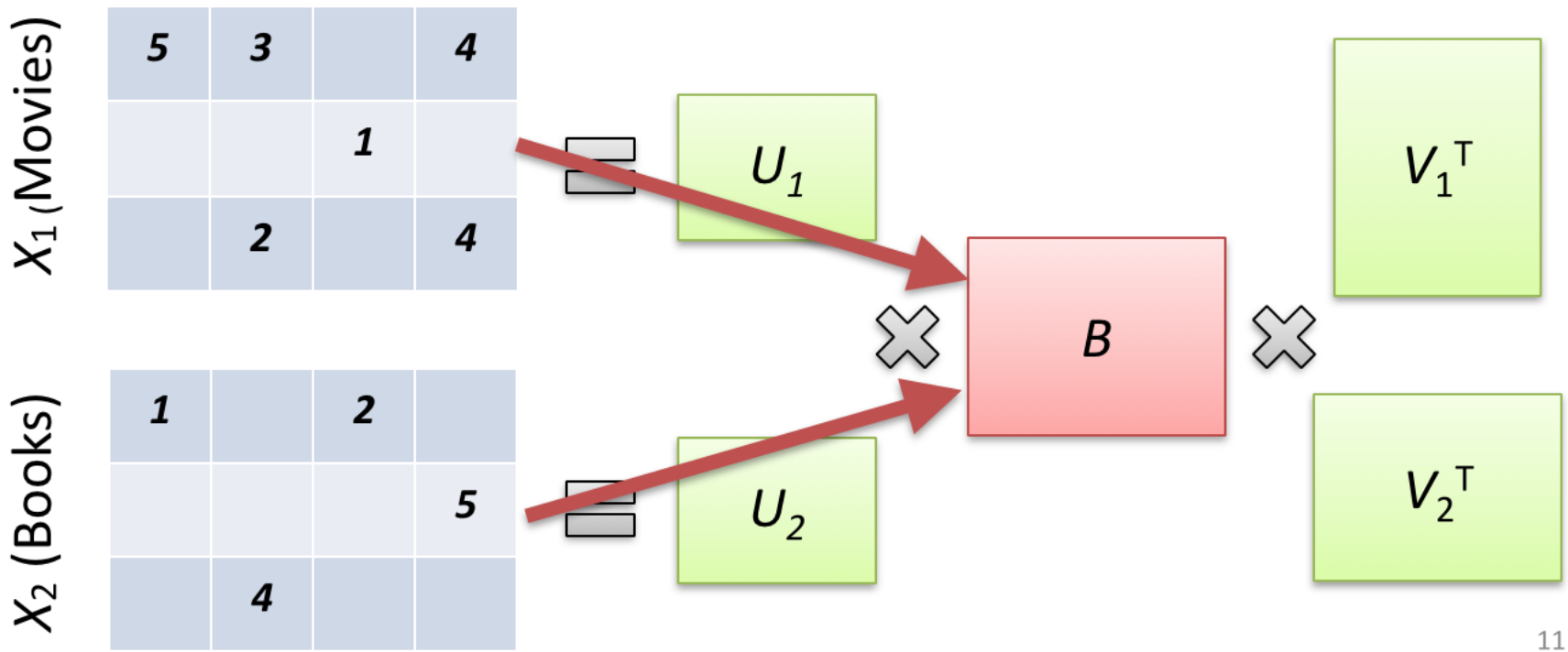
Collective NMTF [9]

$$\arg \min_{\boldsymbol{U}^{(k)},\boldsymbol{B},\boldsymbol{V}^{(k)}>\boldsymbol{O}} \sum_k \|\boldsymbol{X}^{(k)} - \boldsymbol{U}^{(k)}\boldsymbol{B}\boldsymbol{V}^{(k)\top}\|_F^2$$



Collective NMTF [9]

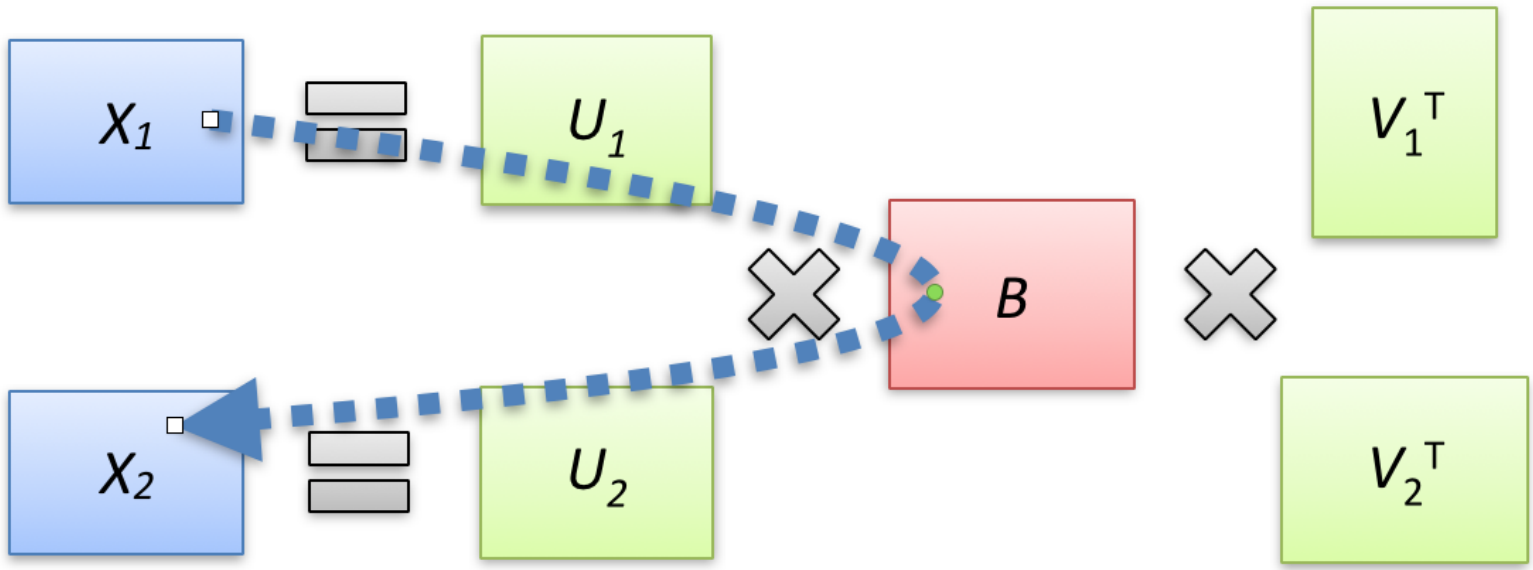
$$\arg \min_{\boldsymbol{U}^{(k)}, \boldsymbol{B}, \boldsymbol{V}^{(k)} > \boldsymbol{O}} \sum_k \|\boldsymbol{X}^{(k)} - \boldsymbol{U}^{(k)} \boldsymbol{B} \boldsymbol{V}^{(k)\top}\|_F^2$$



- More ratings help find better $\boldsymbol{B}$ (and recommendations)

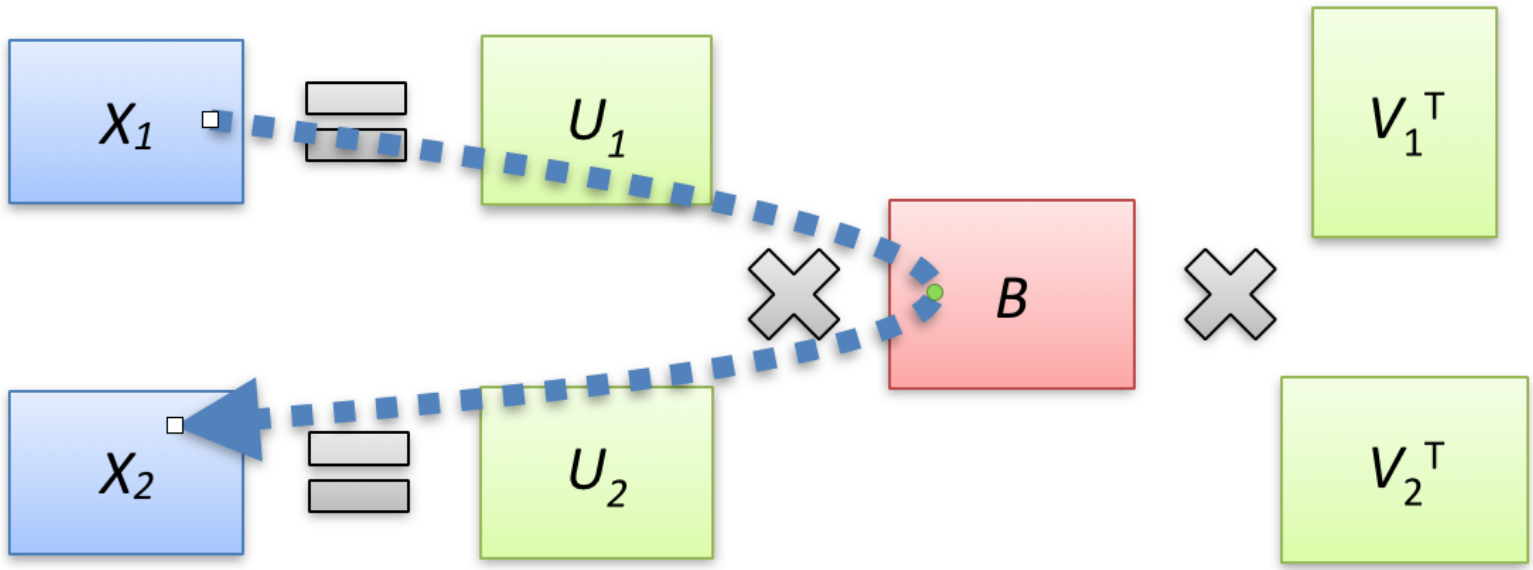
Limitation

- Can only transfer ***linearly*** correlated knowledge



Limitation

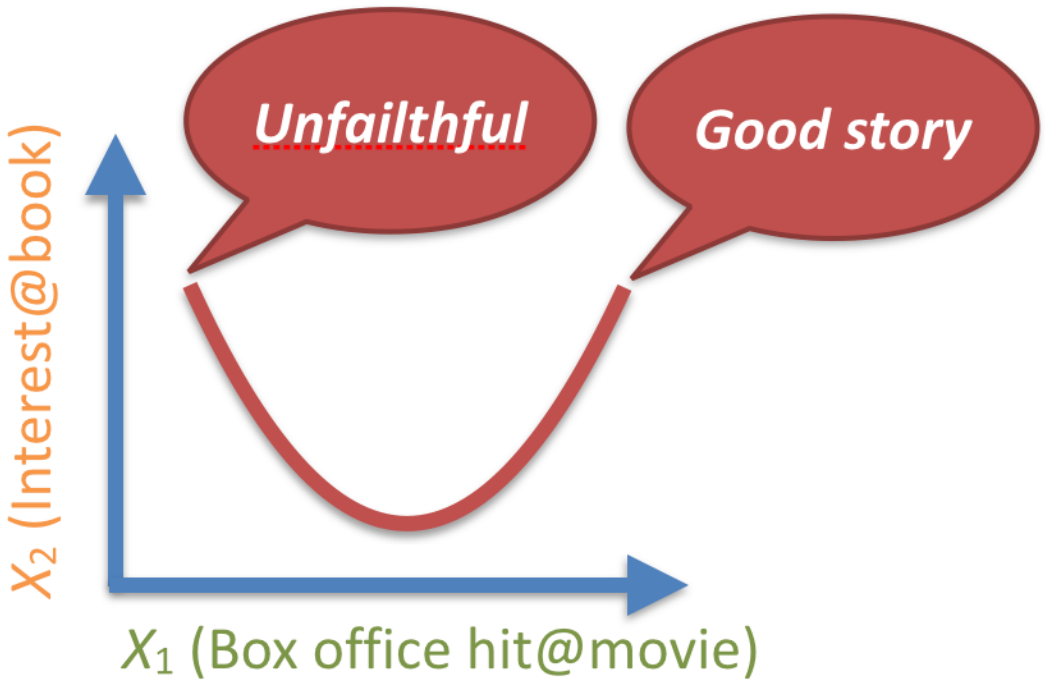
- Can only transfer ***linearly*** correlated knowledge



- In many case, $\boldsymbol{X}^{(\text{source})}$ and $\boldsymbol{X}^{(\text{target})}$ are not linearly correlated

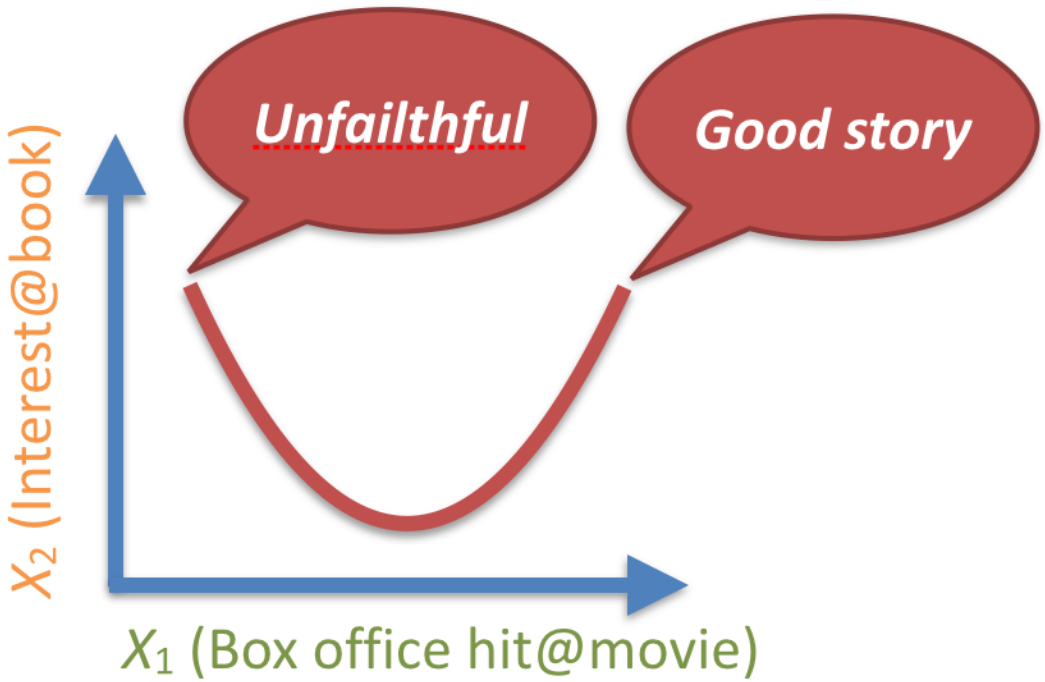
Nonlinearly Correlated Domains

- E.g., suppose we have latent factors:
  - Source (movie): box office hit
  - Target (book): user interests



Nonlinearly Correlated Domains

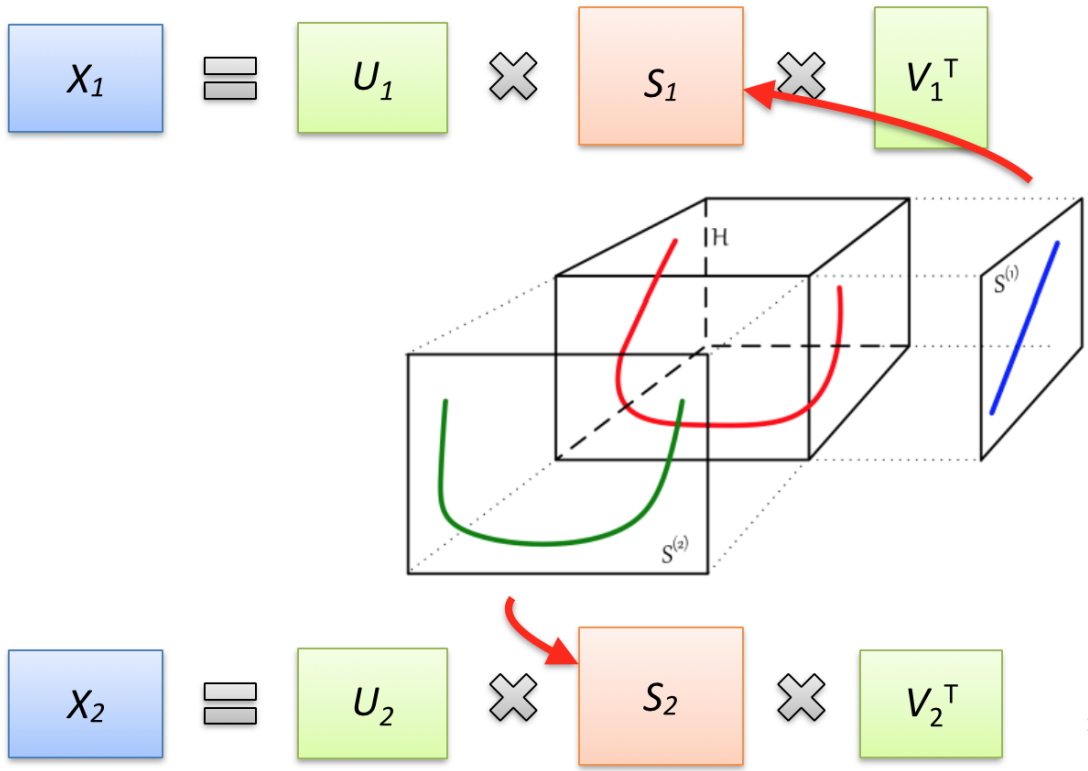
- E.g., suppose we have latent factors:
  - Source (movie): box office hit
  - Target (book): user interests
- How to transfer nonlinearly correlated knowledge?



Hyper Structure Transfer

- Idea: to let $S^{(k)}$'s be *projections of a shared tensor* H [8]:

$$\arg \min_{U^{(k)},\mathbf{H},V^{(k)}>O} \sum_k \|X^{(k)} - U^{(k)} \text{proj}^{(k)}(\mathbf{H}) V^{(k)\top}\|_F^2$$

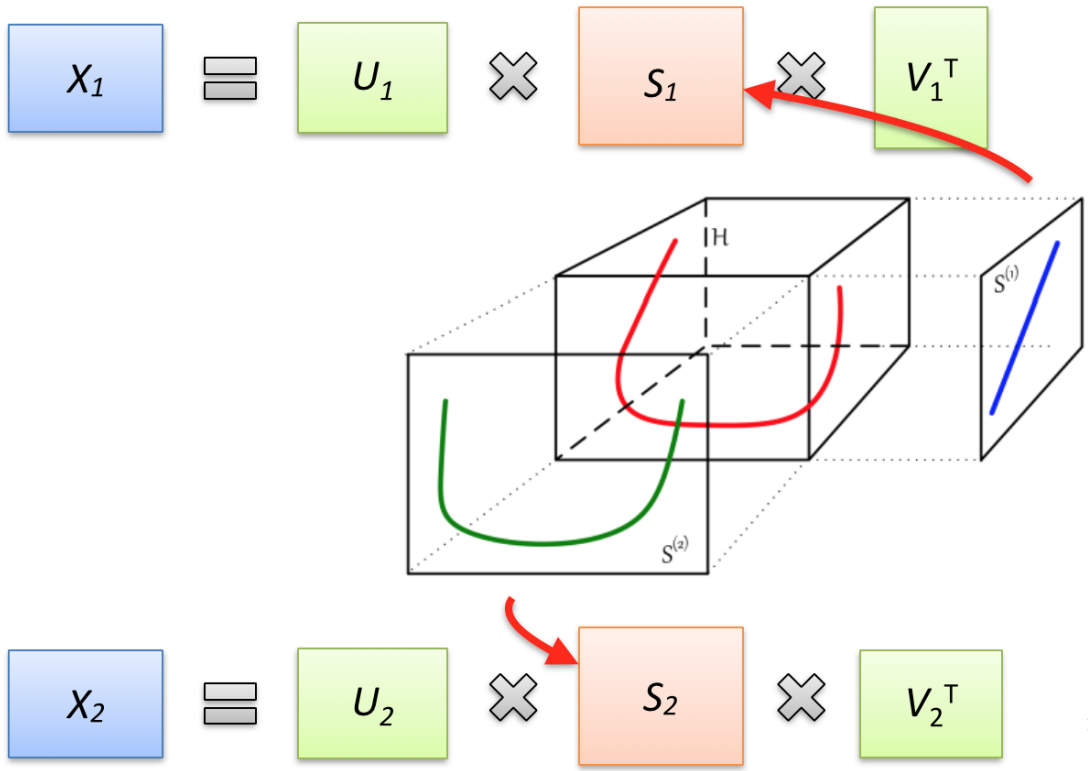


Hyper Structure Transfer

- Idea: to let $\boldsymbol{S}^{(k)}$'s be ***projections of a shared tensor*** $\mathbf{H}$ [8]:

$$\arg \min_{\boldsymbol{U}^{(k)}, \mathbf{H}, \boldsymbol{V}^{(k)} > \boldsymbol{O}} \sum_k \|\boldsymbol{X}^{(k)} - \boldsymbol{U}^{(k)} \text{proj}^{(k)}(\mathbf{H}) \boldsymbol{V}^{(k)\top}\|_F^2$$

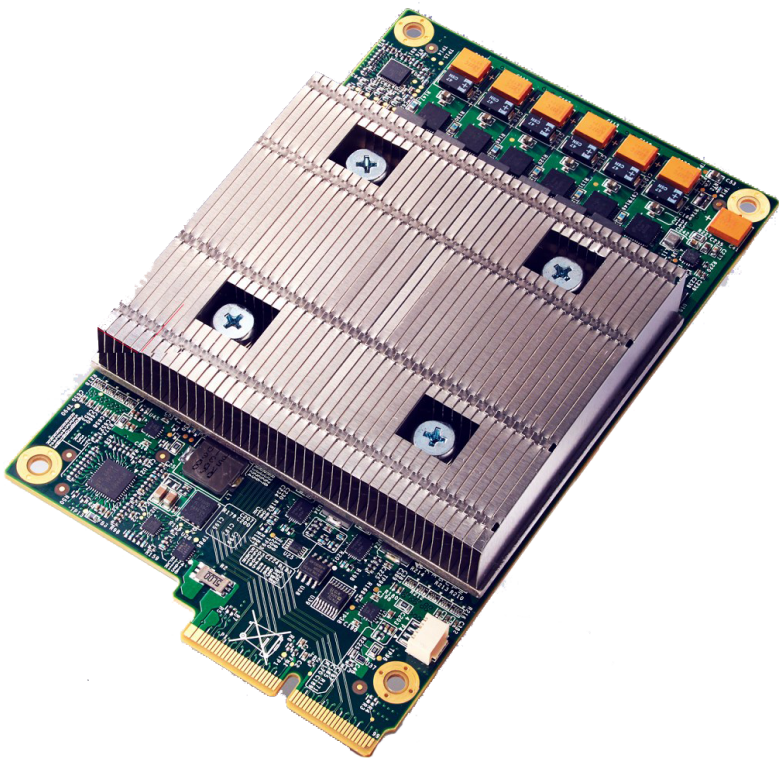
- $\boldsymbol{S}^{(k)} = \text{proj}^{(k)}(\mathbf{H})$'s can be nonlinearly correlated



Outline

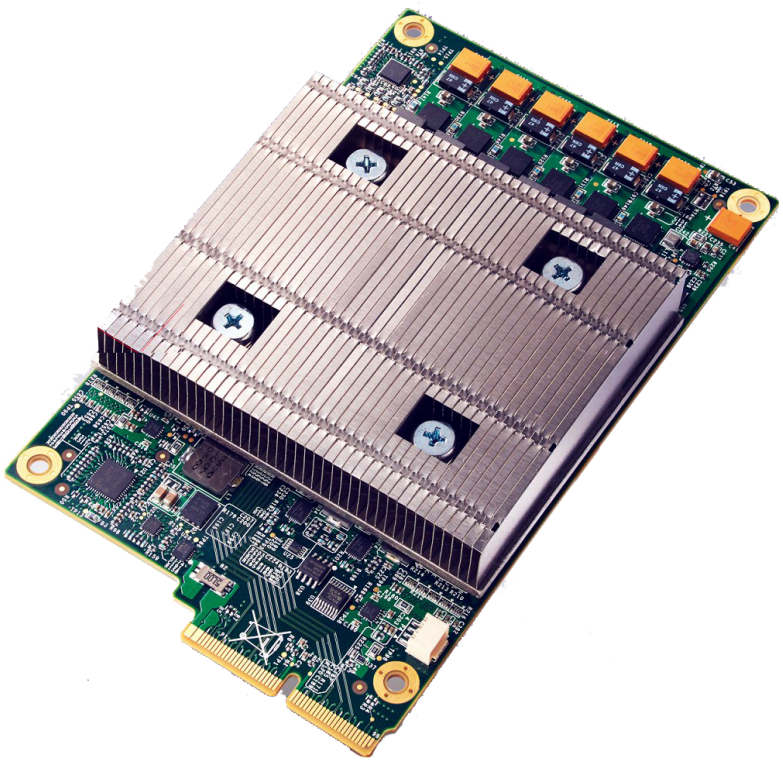
ML-Driven Computers

- Tensor processing units (TPUs) designed by Google:
  - Reduced precision (16- or 8-bit floats)
  - Support TensorFlow
- In Google Photos, each TPU can process 100+ million photos a day



ML-Driven Computers

- Tensor processing units (TPUs) designed by Google:
  - Reduced precision (16- or 8-bit floats)
  - Support TensorFlow
- In Google Photos, each TPU can process 100+ million photos a day



"... performance roughly equivalent to fast-forwarding 7 years into the future (3 gens of Moore's Law)..."

Single "Brain" behind a Corporate?

- Currently, different models for different tasks
  - Inefficient in data collection, computation, and human resource

[1] Google Brain, https://openreview.net/pdf?id=B1ckMDqlg

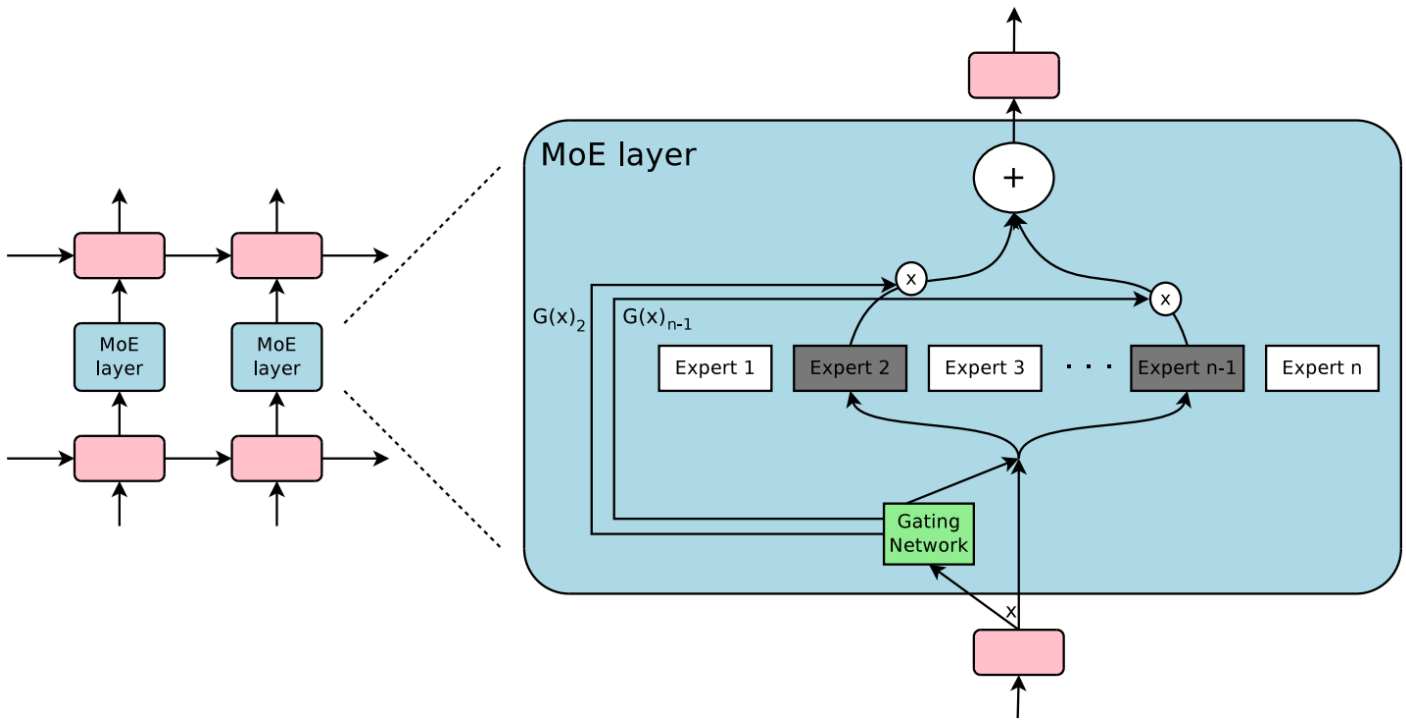
Single "Brain" behind a Corporate?

- Currently, different models for different tasks
  - Inefficient in data collection, computation, and human resource
- Idea: bigger and unified model, but sparsely activated

[1] Google Brain, https://openreview.net/pdf?id=B1ckMDqlg

Single "Brain" behind a Corporate?

- Currently, different models for different tasks
  - Inefficient in data collection, computation, and human resource
- Idea: bigger and unified model, but sparsely activated
- E.g, ***mixture of experts layer***[1] embedded within language model
  - Sparse gating function selects two experts to perform computations
  - Outputs are modulated by the outputs of the gating network



[1] Google Brain, https://openreview.net/pdf?id=B1ckMDqlg

Automated ML

- Currently, solution = ML expert + data + computation

Automated ML

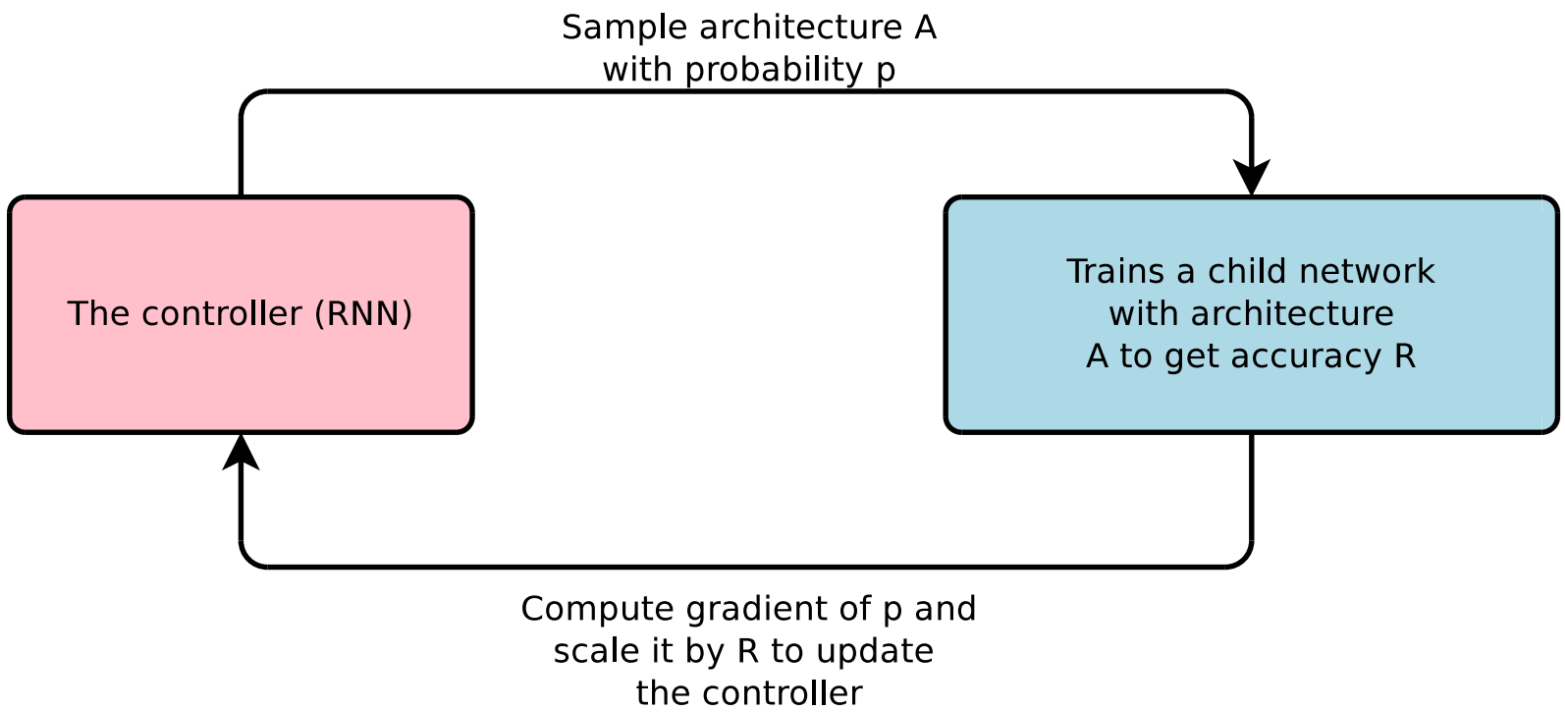
- Currently, solution = ML expert + data + computation
- Can we turn this into: solution = data + 100X computation?

Automated ML

- Currently, solution = ML expert + data + computation
- Can we turn this into: solution = data + 100X computation?
- ***Learning to learn***

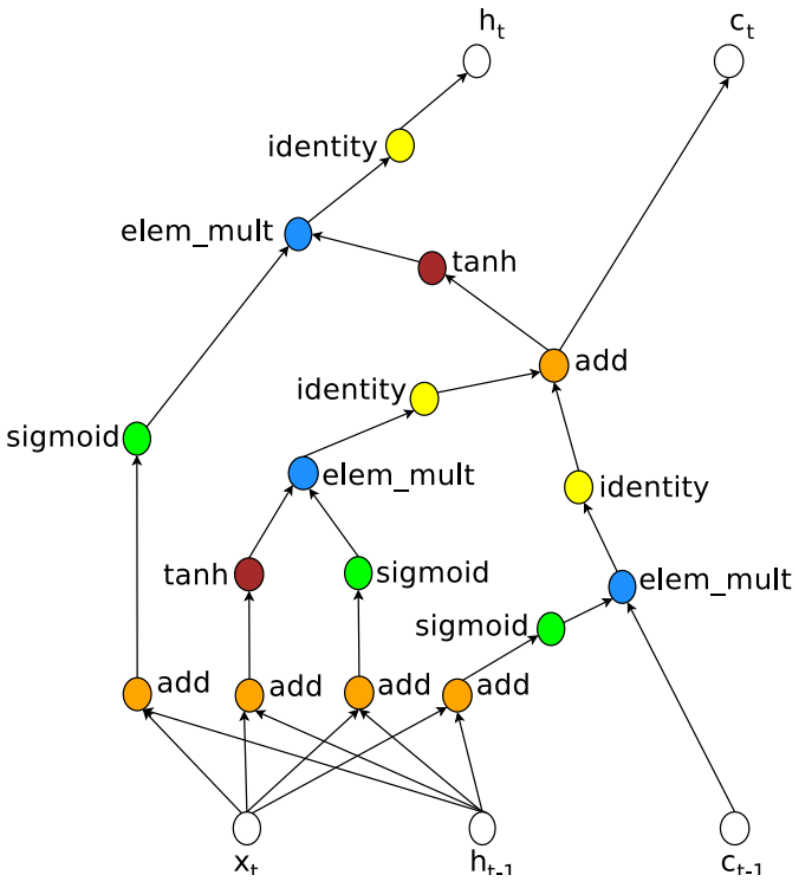
Automated ML

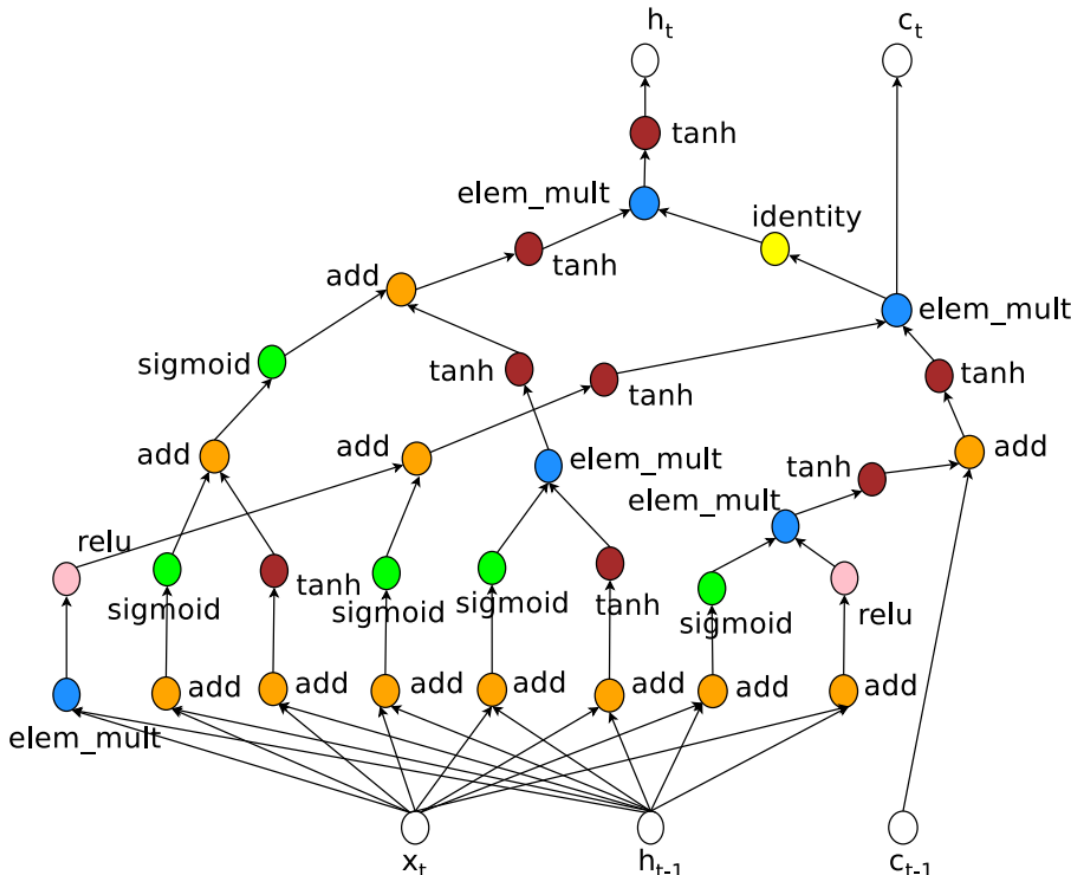
- Currently, solution = ML expert + data + computation
- Can we turn this into: solution = data + 100X computation?
- ***Learning to learn***
- E.g., use ***reinforcement learning*** to search for the best architecture [15]



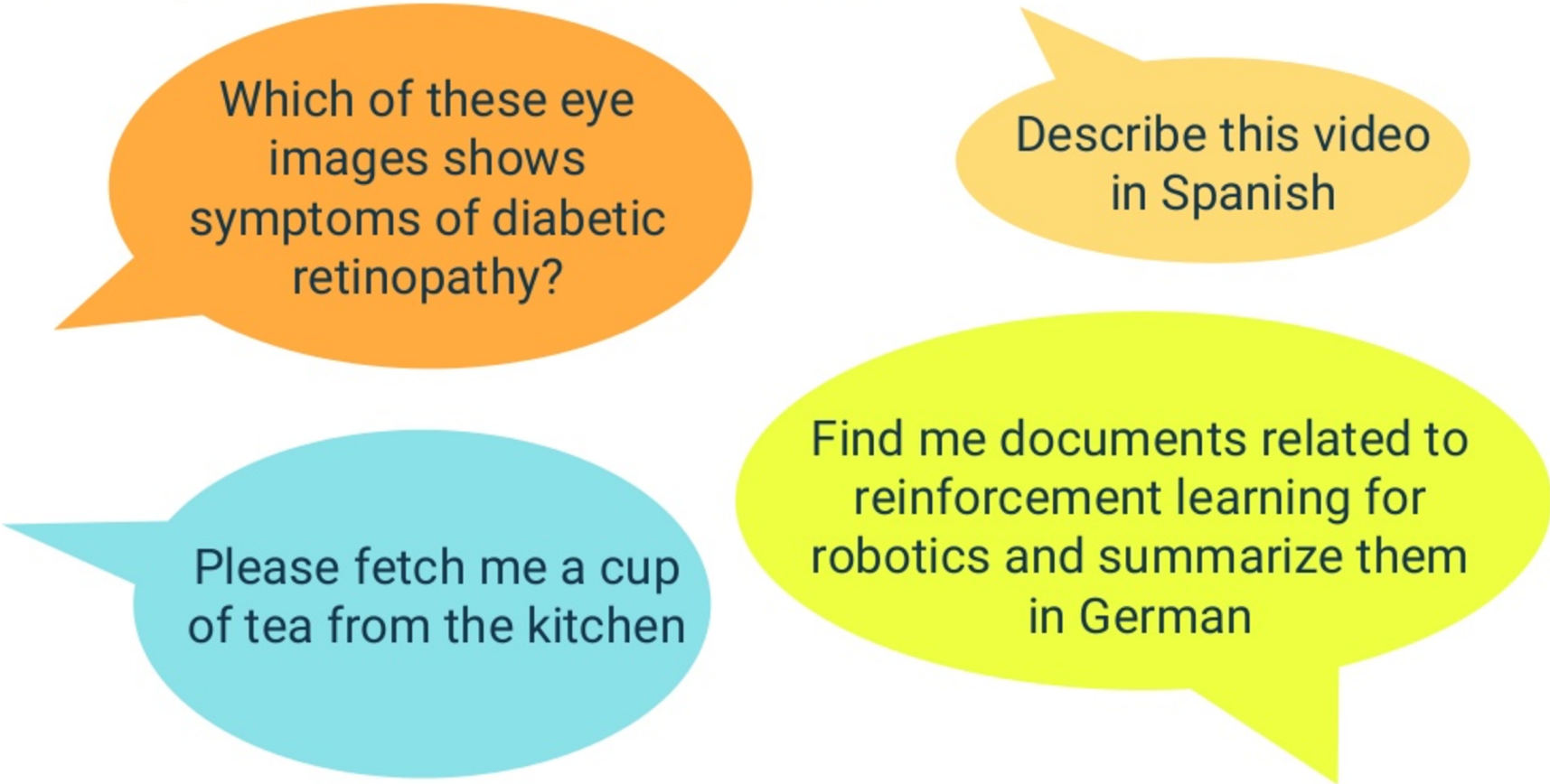
LSTM vs Learned Unit

- Computation graphs



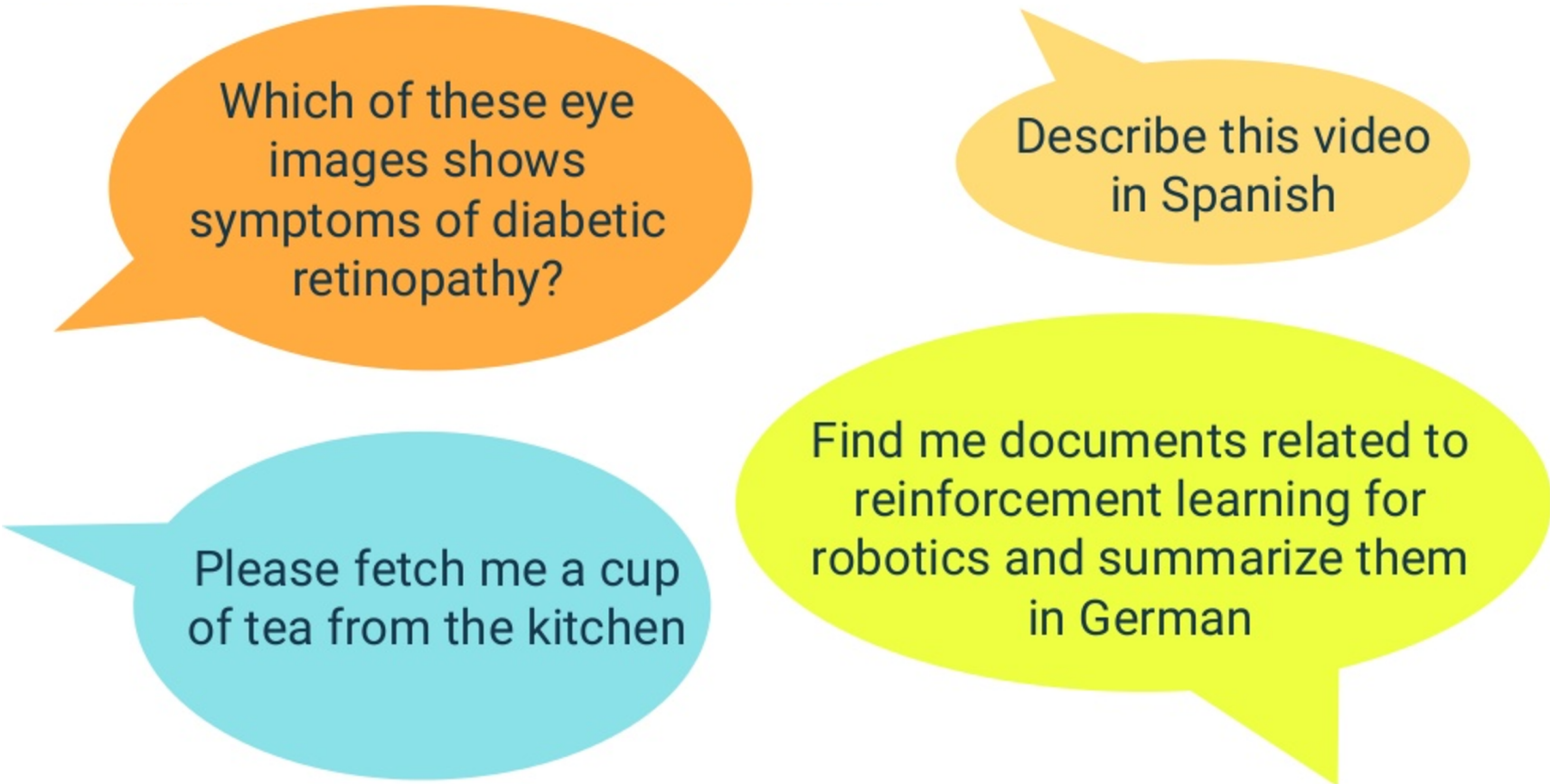


Human-Computer Interaction in the Future



Human-Computer Interaction in the Future

- Or at least how you can query Google in the future...



Reference I

[1] Zeynep Akata, Florent Perronnin, Zaid Harchaoui, and Cordelia Schmid.
Label-embedding for attribute-based classification.
In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 819–826, 2013.

[2] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani.
Manifold regularization: A geometric framework for learning from labeled and unlabeled examples.
Journal of machine learning research, 7(Nov):2399–2434, 2006.

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent.
Representation learning: A review and new perspectives.
IEEE transactions on pattern analysis and machine intelligence, 35(8):1798–1828, 2013.

Reference II

[4] Ting-Yu Cheng, Guiguan Lin, Kang-Jun Liu, Shan-Hung Wu, et al.
Learning user perceived clusters with feature-level supervision.
In *Advances In Neural Information Processing Systems*, pages 532–540, 2016.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei.
Imagenet: A large-scale hierarchical image database.
In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[6] Chris Ding, Tao Li, Wei Peng, and Haesun Park.
Orthogonal nonnegative matrix t-factorizations for clustering.
In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 126–135. ACM, 2006.

Reference III

[7] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky.
Domain-adversarial training of neural networks.
Journal of Machine Learning Research, 17(59):1–35, 2016.

[8] Yan-Fu Liu, Cheng-Yu Hsu, and Shan-Hung Wu.
Non-linear cross-domain collaborative filtering via hyper-structure transfer.
In *ICML*, pages 1190–1198, 2015.

[9] Mingsheng Long, Jianmin Wang, Guiguang Ding, Dou Shen, and Qiang Yang.
Transfer learning with graph co-regularization.
IEEE Transactions on Knowledge and Data Engineering, 26(7):1805–1818, 2014.

Reference IV

[10] Sinno Jialin Pan and Qiang Yang.
A survey on transfer learning.
IEEE Transactions on knowledge and data engineering, 22(10):1345–1359, 2010.

[11] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen.
Improved techniques for training gans.
In *Advances in Neural Information Processing Systems*, pages 2226–2234, 2016.

[12] Patrice Simard, Bernard Victorri, Yann LeCun, and John S Denker.
Tangent prop-a formalism for specifying selected invariances in an adaptive network.
In *NIPS*, volume 91, pages 895–903, 1991.

Reference V

[13] Kiri Wagstaff, Claire Cardie, Seth Rogers, Stefan Schrödl, et al.
Constrained k-means clustering with background knowledge.
In *ICML*, volume 1, pages 577–584, 2001.

[14] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson.
How transferable are features in deep neural networks?
In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[15] Barret Zoph and Quoc V Le.
Neural architecture search with reinforcement learning.
arXiv preprint arXiv:1611.01578, 2016.